دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

# هوش مصنوعی

**پاییز ۱۴۰۰**

استاد: محمدحسین رهبان
گردآورندگان: محمدرضا مفیضی - علیرضا ایلامی - سروش جهان‌زاد

تمرین سوم — Adversarial Search and CSP — مهلت ارسال: ۱ اردیبهشت

- مهلت ارسال پاسخ تا ساعت ۲۳:۵۹ روز مشخص‌شده است.
- در طول ترم امکان ارسال با تاخیر پاسخ همه‌ی تمارین تا سقف ۷ روز و در مجموع ۲۰ روز، وجود دارد. پس از گذشت این مدت، پاسخ‌های ارسال‌شده پذیرفته نخواهند بود. همچنین، به ازای هر روز تأخیر غیر مجاز ۱۰ درصد از نمره تمرین به صورت ساعتی کسر خواهد شد.
- هم‌کاری و هم‌فکری شما در انجام تمرین مانعی ندارد اما پاسخ ارسالی هر کس حتما باید توسط خود او نوشته شده باشد.
- در صورت هم‌فکری و یا استفاده از هر منابع خارج درسی، نام هم‌فکران و آدرس منابع مورد استفاده برای حل سوال مورد نظر را ذکر کنید.
- لطفا تصویری واضح از پاسخ سوالات نظری بارگذاری کنید. در غیر این صورت پاسخ شما تصحیح نخواهد شد.

## سوالات نظری (۷۰ نمره)

۱. (۱۵ نمره) درستی یا نادرستی گزاره‌های زیر را با ذکر دلیلی کوتاه مشخص کنید.

(آ) زمان اجرای یک روش کارآمد برای مسائل CSP با ساختار درختی، رابطه‌ای خطی با تعداد متغیرها دارد.

(ب) مجموعه مقادیری که پس از اجرای الگوریتم arc consistency باقی می‌مانند، به ترتیب برداشتن arc ها از صف بستگی ندارد.

(ج) هنگامی که arc consistency به عنوان یک مرحله‌ی پیش‌پردازش اعمال می‌شود، می‌توان از forward checking در طول backtracking برای حفظ arc consistency برای همه متغیرها استفاده کرد.

(د) هرس آلفا بتا [۱] پس از اعمال یک تابع اکیدا صعودی روی مقدار برگ‌ها همان شاخه‌های قبلی را هرس می‌کند.

(ه) پس از اعمال یک تابع اکیدا صعودی بر روی مقدار برگ‌ها، ترتیب expectimax همان action های قبلی را انتخاب می‌کند.

(و) در حالت کلی می‌توان درخت‌های expectimax و expectimin را هرس کرد.

۲. (۱۵ نمره) به سوال‌های زیر پاسخ کوتاه بدهید.

(آ) چرا ترجیح می‌دهیم CSP های دارای ساختار درختی را حل کنیم؟

(ب) یک روش استاندارد برای تبدیل مسائلی با ساختار تقریباً درختی به مسائلی با ساختار درختی را توضیح دهید.

(ج) حداکثر تعداد دفعاتی که الگوریتم backtracking ممکن است مجبور به backtrack شود، اگر از arc consistency و هیوریستیک‌های MRV و LCV استفاده کند، چقدر است؟

۳. (۱۵ نمره) شما مسئول برگزاری و هماهنگی مراسم اسکار ۲۰۲۳ هستید! در این مراسم قرار است ۵ فیلم مختلف در سالن‌های مجزا معرفی شود و از بازیگران آن‌ها برای گفت‌وگو دعوت شود. متاسفانه فقط ۳ بازیگر دعوت شما را قبول کرده‌اند و قرار است در مراسم معرفی فیلم خود شرکت کنند. فیلم‌ها و زمان معرفی آن‌ها به شرح زیر است:

[۱] Alpha-beta pruning

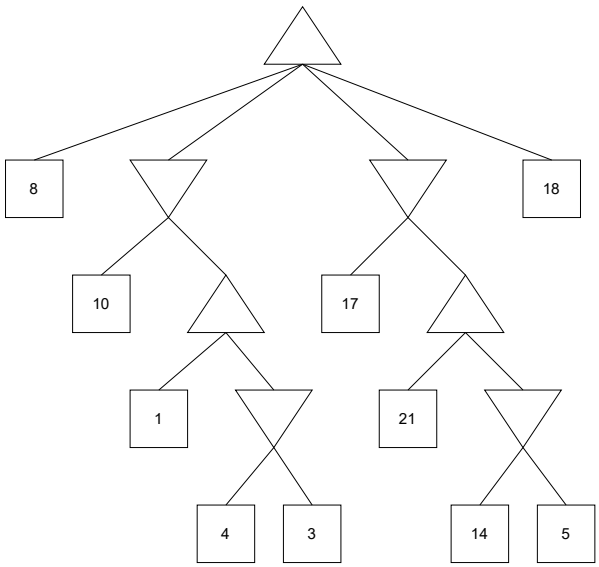
- فیلم اول: از ساعت ۸ تا ۹ صبح
- فیلم دوم: از ساعت ۸:۳۰ تا ۹:۳۰ صبح
- فیلم سوم: از ساعت ۹ تا ۱۰ صبح
- فیلم چهارم: از ساعت ۹ تا ۱۰ صبح
- فیلم پنجم: از ساعت ۹:۳۰ تا ۱۰:۳۰ صبح

و بازیگرانی که در مراسم شرکت می‌کنند به همراه فیلم‌هایی که در آن ایفای نقش کرده‌اند به شرح زیر است:

- بازیگر اول: ایفای نقش در فیلم‌های سوم و چهارم
- بازیگر دوم: ایفای نقش در فیلم‌های دوم، سوم، چهارم و پنجم
- بازیگر سوم: ایفای نقش در فیلم‌های اول، دوم، سوم، چهارم و پنجم

(آ) این کار را با یک مسئله CSP مدل کنید و دامنه‌ها و محدودیت‌ها را به صورت دقیق بیان کنید.
(ب) گراف محدودیت CSP را رسم کنید.
(ج) دامنه‌ی متغیرها را پس از اعمال arc consistency روی گراف اولیه (و بعد از اعمال شروط unary) بنویسید.
(د) یک جواب قابل قبول (درصورت وجود) برای مسئله بیابید.

۴. (۱۵ نمره) درخت minimax زیر را درنظر بگیرید:

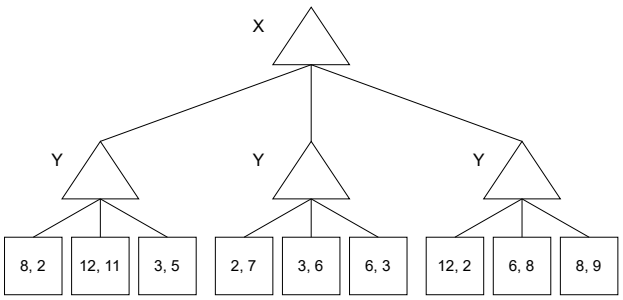



(آ) مقدار minimax ریشه چه خواهد بود؟
(ب) هرس آلفا بتا را اجرا کنید. فرض کنید که گره‌ها از چپ به راست بررسی می‌شوند.
(ج) آیا ترتیب دیگری برای فرزندان ریشه وجود دارد که هرس بیشتری در پی داشته باشد؟ اگر وجود دارد، ترتیب را بنویسید.
(د) یک روش کلی و عملی برای ترتیب دادن به فرزندان گره‌ها پیشنهاد کنید که به افزایش فرصت‌های هرس منجر می‌شود. توضیح دهید در مورد گره‌های کمینه و بیشینه چه باید کرد.

۵. (۱۰ نمره) به بازی non-zero-sum زیر توجه کنید. در این مثال، utility بازیکن X اولین عدد (سمت چپ) و utility بازیکن Y دومین عدد از دو عدد برگ است.

(آ) با فرض اینکه هر بازیکن بهینه عمل می‌کند، درخت بازی را پر کنید.

(ب) کدام گره‌ها را می توان از درخت بازی بالا با روش آلفا بتا هرس کرد؟ اگر هیچ گره‌ای را نمی‌توان هرس کرد، توضیح دهید که چرا نه.




## سوالات عملی (۷۰ نمره)

۱. (۳۰ نمره) در این تمرین برای یادگیری مفاهیم CSP شما باید جدول سودوکو را پیاده‌سازی کنید.

محدودیت زمانی سوال: ۱۵۰ ms

ورودی سوال عبارت است از یک جدول ۹ در ۹ که برخی خانه‌های آن پر شده‌اند. ورودی در ۹ سطر داده می‌شود و خانه‌های هر سطر با space جدا شده‌اند. خانه‌های خالی نیز با . (نقطه) نمایش داده می‌شوند. برای حل این سوال از هر الگوریتم CSP دلخواه می‌توانید استفاده کنید. اما توصیه می‌شود از الگوریتم‌های پیچیده‌تر استفاده کنید تا با محدودیت زمانی برخورد نکنید.

الگوریتم‌های مجاز:

BackTracking, AC3, AC3-LCV, AC3-MRV, AC3-LCV-MRV

خروجی: شما باید به همان فرمت ورودی، جدول سودوکو را کامل کرده و خروجی دهید. یعنی در ۹ سطر که هر سطر حاوی ۹ عدد از بین ۱ تا ۹ است که با space جدا شده‌اند.

نکات: شما باید فقط یک فایل پایتون آپلود کنید که ورودی را از کاربر بخواند و خروجی را چاپ کند.

پاسخ یک جدول سودوکو، لزوما یکتا نیست.

همچنین نمره‌ی نهایی شما، نمره تست‌های کوئرا نیست. پس از تصحیح و بررسی کد شما، نمرات مشخص می‌شوند.

یک مثال از ورودی و خروجی سوال:

input:

. 1 . 4 . . . . 2

3 . . . 5 . . . .

8 . . . . . 6 4 .

. 3 . 2 . 5 . . 6

7 . 4 9 . . . . .

9 . . . . . . . 4

4 . . 1 . 2 . . .

. . . . . . 3 . .

2 . . 6 . . . . 7

output:

5 1 6 4 9 7 8 3 2

3 4 2 8 5 6 7 9 1

8 7 9 3 2 1 6 4 5

1 3 8 2 4 5 9 7 6

7 2 4 9 6 8 1 5 3

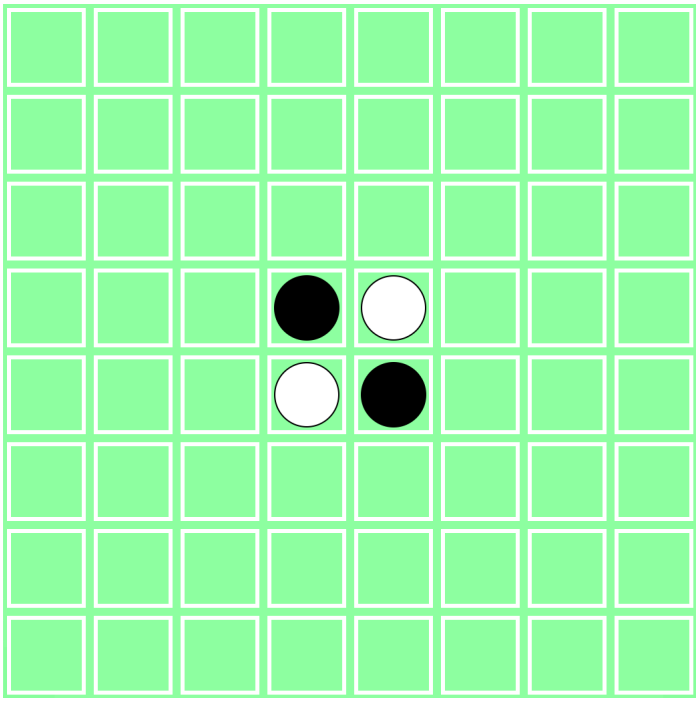
9 6 5 7 1 3 2 8 4

4 9 7 1 3 2 5 6 8

6 8 1 5 7 4 3 2 9

2 5 3 6 8 9 4 1 7

۲. (۴۰ نمره) بازی اتلو مهره‌هایی دارد که یک روی آن‌ها سفید و یک روی آن‌ها سیاه است. در آغاز، ۴ مهره به شکل زیر روی یک تخته‌ی ۸ در ۸ قرار می‌گیرند و هر کدام از دو بازیکن، یک رنگ (سیاه یا سفید) را انتخاب می‌کنند.



سپس با شروع از بازیکن سیاه، هر کس در نوبت خود مهره‌اش را در جایی قرار می‌دهد که تعدادی مهره‌ی غیرهم‌رنگ میان مهره‌ی تازه‌گذاشته‌شده و مهره‌های هم‌رنگ قرار بگیرند. با این کار، این مهره‌های میانی تغییر رنگ می‌دهند. اگر بازیکنی حرکت مجازی نداشته باشد نوبت به بازیکن بعدی می‌رسد و اگر هیچ‌کدام نتوانند حرکتی انجام دهند بازی به پایان می‌رسد. در پایان، بازیکنی برنده است که مهره‌های بیشتری در زمین رنگ او را داشته باشند. (برای آشنایی بیشتر با این بازی می‌توانید درباره‌ی آن در اینترنت جست‌وجو کنید.)

حال برای این سوال، شما باید بازیکنی طراحی کنید که با روش هرس آلفابتا، برنده‌ی این بازی شود. برای شروع، کد این بازی به همراه یک بازیکن ساده به شما داده شده است. تنها کاری که شما باید انجام دهید این است که در فایلی به نام alphabeta.py یک کلاس به نام AlphaBetaPlayer ایجاد کنید که از کلاس Player ارث‌بری می‌کند و تابع get_next_move آن کلاس را به گونه‌ای بنویسید که بهترین حرکت از نظر بازیکنتان را به صورت زوج مرتبی از شماره‌ی سطر و ستون (با شروع از 0 و با حرکت از بالا به پایین و از چپ به راست) برگرداند.

توجه کنید که در کدهای داده شده، بازیکن سیاه با شماره‌ی 0 و بازیکن سفید با شماره‌ی 1 مشخص شده‌اند. اگر در هر نوبتی یک بازیکن حرکت نامعتبری را به عنوان انتخابش برگرداند، فرض می‌شود که حرکتش را واگذار کرده‌است و بدون این‌که اتفاق خاصی رخ بدهد، نوبت به بازیکن بعدی می‌رسد. زمانی که هیچ حرکت معتبری وجود نداشته باشد، آزاد هستید که یک حرکت نامعتبر یا None را به عنوان خروجی تابع get_next_move برگردانید.

در نظر داشته باشید که بازیکن شما در چند بازی با بازیکنی مطابق پاسخ مورد انتظار سنجیده خواهد شد و نمره‌ی سوال با توجه به پیاده‌سازی و کد شما و همین‌طور با در نظر گرفتن عملکرد بازیکنتان در بازی به شما داده خواهد شد. ماژول game.py برای ارزیابی بازیکن در طول حل سوال در اختیارتان قرار گرفته است.