تمرین سری سوم: مسائل ارضای محدودیت

لطفاً به نکات زیر توجه کنید:

- مهلت ارسال این تمرین تا ۳ اردیبهشت است.
- در صورتی که به اطلاعات بیشتری نیاز دارید می‌توانید به صفحه‌ی تمرین در وب‌سایت درس مراجعه کنید.

- این تمرین شامل سوال‌های برنامه‌نویسی می‌باشد، بنابراین توجه کنید که حتماً موارد خواسته‌شده در سوال را رعایت کنید. در صورتی که به هر دلیلی سامانه‌ی داوری نتواند آن را اجرا کند مسئولیت آن تنها به عهده‌ی شماست.
- ما همواره هم‌فکری و هم‌کاری را برای حلِ تمرین‌ها به دانشجویان توصیه می‌کنیم. اما هر فرد باید تمامی سوالات را به تنهایی تمام کند و پاسخ ارسالی حتماً باید توسط خود دانش‌جو نوشته‌شده باشد. لطفاً اگر با کسی هم‌فکری کردید نام او را ذکر کنید. در صورتی که سامانه‌ی تطبیق، تقلبی را تشخیص دهد متأسفانه هیچ مسئولیتی بر عهده‌ی گروه تمرین نخواهد بود.
- لطفاً برای ارسال پاسخ‌های خود از راهنمای موجود در صفحه‌ی تمرین استفاده کنید.
- هر سوالی درباره‌ی این تمرین را می‌توانید از دستیاران حلِ تمرین بپرسید.

- آدرس گروه درس: https://groups.google.com/forum/#!forum/ai972
- صفحه تمرین: https://quera.ir/course/assignments/8862/problems

موفق باشید

مقدمه: معرفی فریم‌ورک

همان‌طور که می‌دانید برای حل مسائل ارضای محدودیت راه‌های مختلفی وجود دارد. برای اینکه بتوانیم یک مسئله‌ی ارضای محدودیت را حل کنیم ابتدا باید آن را مدل سازی کنیم. در فایل csp.py یک مسئله ارضای محدودیت در کلاس CSP پیاده‌سازی شده است. توابع و متغیرهای مهم این کلاس عبارتند از:

توابع و متغیرهای کلاس CSP	
variables	شامل لیستی از متغیرهای مسئله است.
domains	یک دیکشنری که مقادیر کلیدهای آن، متغیر ها و مقادیر هر متغیر، لیستی از مقدار های قابل اخذ توسط آن متغیر است.
neighbors	یک دیکشنری که کلید های آن متغیر ها و مقادیر آن‌ها لیستی از متغیر های همسایه آن متغیر است.
constraints	تابعی است که وجود محدودیت بین دو متغیر با مقادیر مختلف را بررسی می‌کند.
assign(var,val,a)	این تابع مقدار val را برای متغیر var در نظر می‌گیرد.
unassign(var, a)	مقدار قبلی متغیر var را پاک میکند.
nconflicts(var,val,a)	تعداد conflict هایی که در صورت اختصاص val به var به وجود می‌آید را برمی‌گرداند.
curr_domains	یک دیکشنری مانند domains است که می‌توان مقادیر دامنه‌ی یک متغیر را از آن برای حل مسئله پاک کرد.
display	این تابع حالت کنونی مسئله را چاپ می‌کند.
support_pruning	این تابع قابلیت حذف مقادیر دامنه از متغیرها را فعال می‌کند.
prune	مقدار ورودی را از دامنه متغیر داده شده حذف می‌کند.
infer_assignment	وضعیت کنونی دامنه ها را به صورت قابل نمایش برای تابع display به عنوان خروجی بر می‌گرداند.

(باقی توابع در کد به صورت مفصل توضیح داده شده است.)

در این سری تمرین، مسئله سودوکو برای شما انتخاب شده است. فایل sudoku.py حاوی کلاس Sudoko است که از کلاس CSP ارث برده است و مسئله‌ی سودوکو رو مدل‌سازی کرده است. برای این بخش از تمرین، ابتدا سعی کنید کدهای مربوط به مسئله و CSP را خوب متوجه شوید!

برای حل کردن این مسئله الگوریتم Backtracking پیاده‌سازی شده است. مسئولیت شما برای سوال‌های بعد پیاده‌سازی Orderingها و Filtering های مختلف برای حل مسئله و همچنین مقایسه راه حل های متخلف است.

برای اجرای الگوریتم های مختلف می‌توانید از دستورات زیر استفاده کنید:

```
$ python3 run.py -b <BOARD_NAME> -valo <VALUE_ORDERING_METHOD> -varo <VARIABLE_ORDERING_METHOD>
-inf <FILTERING_METHOD>
```

در دستور فوق می‌توان پارامترهای مربوط به راه‌حل مسئله را تعیین کرد.

پارامتر BOARD_NAME : نام فایل سودوکو موجود در پوشه‌ی boards

پارامتر VALUE_ORDERING_METHOD : نام تابع استراتژی مربوط به Value Ordering (مانند lcv)

پارامتر VARIABLE_ORDERING_METHOD : نام تابع استراتژی مربوط به Variable Ordering (مانند mrv)

پارامتر FILTERING_METHOD : نام تابع مربوط به فیلترینگ (مثل forward_checking و arc_cons)

سوالات عملی

۱- پیاده‌سازی MRV (۱۵ نمره)

همانطور که می‌دانید، یکی از روش های سریع‌تر کردن حل مسئله، مرتب کردن انتخاب های الگوریتم در هر مرحله است. یکی از روش های مرتب کردن Minimum Remaining Value یا MRV است. در این روش متغیرهایی که کمترین مقدار باقی مانده را دارد زودتر انتخاب می‌شوند.

برای پیاده‌سازی این بخش باید تابع mrv در فایل solutions.py را تکمیل کنید. این تابع باید متغیری که کمترین طول دامنه را دارد برگردانده شود. برای اجرای این سوال می‌توانید از دستور شرح داده شده استفاده کنید.

۲- پیاده‌سازی LCV (۱۵ نمره)

یکی از روش‌های دیگر Ordering برای حل مسائل ارضای محدودیت Least Constraining Values یا LCV است. در این روش متغیرهای که با انتخاب آنها کم‌ترین ناسازگاری پیش‌خواهد آمد انتخاب خواهد شد.

برای پیاده‌سازی این بخش باید تابع lcv در فایل solutions.py را تکمیل کنید. در این تابع باید یک لیست مرتب شده بر اساس کمترین ناسازگاری از متغیرهای مسئله برگردانده شود. برای اجرای این سوال می‌توانید از دستور شرح داده شده استفاده کنید.

۳- پیاده‌سازی Forward Checking (۱۵ نمره)

برای سریع‌تر کردن الگوریتم، میتوان از الگوریتم هایی برای فیلتر کردن متغیرها استفاده کرد. یکی از این الگوریتم ها Forward Checking است. در این الگوریتم مقادیر غیر ممکن برای همسایه‌های متغیر از دامنه‌شان حذف خواهد شد.

برای پیاده‌سازی این بخش باید تابع forward_checking در فایل solutions.py را تکمیل کنید. این تابع یک مقدار Boolean برمی‌گرداند. در صورت عدم امکان ادامه‌ی الگوریتم False و در غیر اینصورت True مقدار خروجی خواهد بود. برای اجرای این سوال می‌توانید از دستور شرح داده شده استفاده کنید.

۴- پیاده‌سازی Arc Consistency (۱۵ نمره)

از روش‌های دیگر برای فیلتر کردن متغیرها Arc Consistency است. در این روش تغییرات در کل گراف مسئله منتشر می‌شود و مقادیر غیر ممکن برای تمامی متغیر ها در گراف مسئله اصلاح می‌گردد.

برای پیاده‌سازی این بخش باید تابع arc_cons در فایل solutions.py را تکمیل کنید. این تابع یک مقدار Boolean برمی‌گرداند. در صورت عدم امکان ادامه‌ی الگوریتم False و در غیر اینصورت True مقدار خروجی خواهد بود. برای اجرای این سوال می‌توانید از دستور شرح داده شده استفاده کنید.

۵- مقایسه الگوریتم ها (۱۰ نمره)

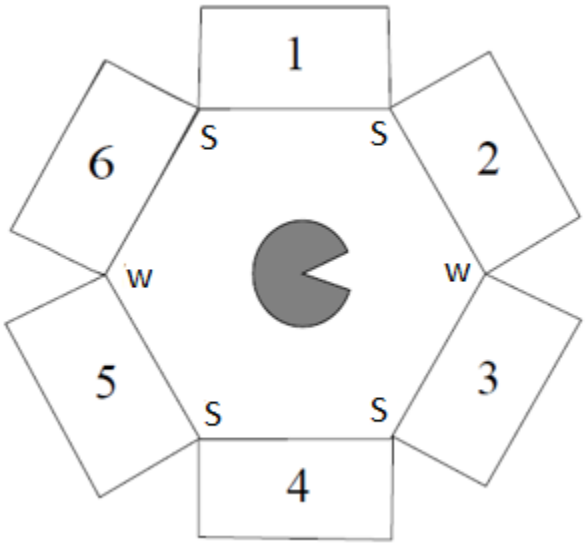
پس از پیاده‌سازی تمام الگوریتم‌های سوالات قبل، به بررسی میزان حافظه اشغال شده و مدت زمان اجرای الگوریتم ها بپردازید. با مقایسه و تحلیل نتایج به دست آمده، تفسیر خود را از تعویض الگوریتم ها با استراتژی های مختلف بنویسید. برای این کار می‌توانید از نمودار و عکس‌های مختلف استفاده کنید.

سوال‌های تئوری

سوال اول

پکمن گیر افتاده است! او با دیوارهایی که پشت آن‌ها غول (P) یا روح (G) یا راه خروج (E) قرار دارد محاصره شده است. برای این که پکمن بتواند فرار کند باید دیواری که به خروج منتهی می‌شود را پیدا کند. یکی از نشانه‌هایی که کمک می کند تا بفهمیم پشت دیوار چیست، بادی ست که می‌وزد. غول باد قوی تولید می‌کند (S) و راه خروج باد ضعیف (W) و روح هیچ بادی تولید نمی‌کند. اما پکمن نمی‌تواند هر جریان هوا رو به صورت جداگانه اندازه‌گیری کند. در عوض در نقطه‌ی به هم رسیدن دو دیوار می‌تواند برایند دو جریان هوا را اندازه گیری کند. مثلا بین دو دیوار که پشت آن‌ها غول وجود دارد ، برایند دو جریان هوا، قوی است و جریان قوی احساس می‌شود. یا بین دو دیوار که پشت یکی غول هست و پشت یکی خروج، جریان قوی احساس می‌شود. بین دو دیوار خروج و روح هم جریان ضعیف احساس می‌شود.

تعداد خروج ها ممکن است صفر، یک و یا بیشتر باشد. پکمن می‌داند که هیچ دو خروجی پشت دیوارهای مجاور هم نیستند.



برای مدل کردن مسئله ی پکمن از X_i برای نشان دادن موجود پشت دیوار i اُم استفاده کنید. دامنه‌ی این متغیر P ، G و یا E است.

الف – به صورت binary و یا unary محدودیت های مستقیم و غیر مستقیم این مدل را بنویسید.

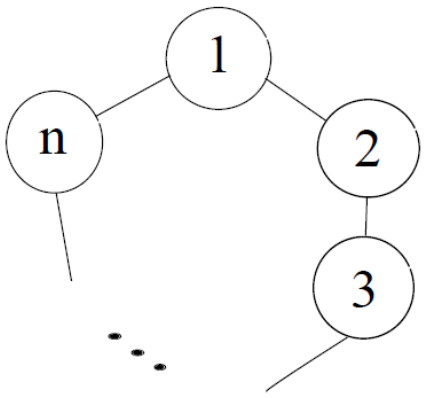
ب- جدول زیر را بعد از اعمال عملیات arc consistency با توجه به دامنه ی مسئه تکمیل کنید.

X_1	
X_2	
X_3	
X_4	
X_5	
X_6	

ج- با توجه به MRV، کدام متغییر یا متغییرها در مرحله‌ی اول به جواب می رسند؟

د- فرض کنید پکمن میداند پشت دیوار شماره ۶ روح قرار دارد. تمام حل های ممکن برای این CSP را بنویسید یا اگر حلی وجود ندارد آن را ذکر کنید.

مسئله‌ی CSP که در بالا دیدید یک ساختار دایروی با ۶ متغیر است. حل یک مسئله ی CSP مانند بالا را در نظر بگیرید که n متغییر دارد (n>2) همچنین در نظر بگیرید دامنه ی مسئله d عضو دارد.

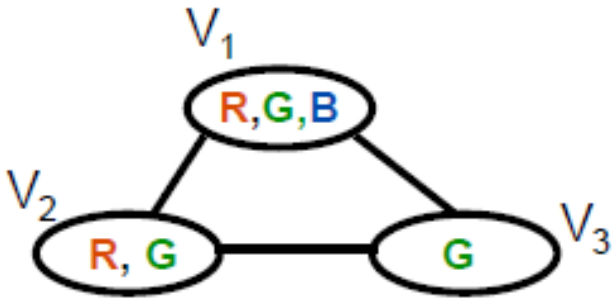


هـ - با جزئیات توضیح دهید این مسئله ی CSP با ساختار دایروی را چگونه میتوان به روش بهینه حل کرد؟ (مثلا در زمان خطی نسبت به تعداد متغیرها). از روش‌های گفته شده در کلاس استفاده کنید.

و – اگر از شیوه‌ی جست‌وجوی backtracking معمولی برای این گراف با ساختار دایروی استفاده کنیم و در هر مرحله arc consistency اجرا بشود، درباره‌ی رفتار backtracking در حالت worst-case چه می‌توان گفت؟ (مثلا تعداد دفعاتی که جست‌وجو عقبگرد می‌کند)

سوال دوم

با توجه به شکل زیر به سوالات پاسخ دهید.



الف – جدول Arc Consistency زیر را کامل کنید.

arc consistency	Value deleted
V1 – V2	
V1 – V3	
V2 – V3	
V1 – V2	
V1 – V3	
V2 – V3	

ب – درخت Backtracking را با شروع از راس V1و سپس V2 به شکل کامل رسم کنید.

ج – درخت Backtracking را با فیلتر کردن Forward Checking به شکل کامل با شروع از راس V1 و سپس V2 رسم کنید.

سوال سوم

شما در حال ساخت یک برنامه‌ی اتوماتیک برای حل جدول crossword هستید چون فکر می‌کنید با آن می‌توان پول خوبی به جیب زد! برای آن‌هایی که با این جدول آشنا نیستند، crossword یا همان جدول کلمات متقاطع (در این سوال برای زبان انگلیسی) یک جدول مربعی است که در آن حروف قرار دارند و باید کلمه های معنی‌دار متقاطع را به صورت چپ به راست یا بالا به پایین پیدا کنید. همچنین یک کلمه در دو جای جدول تکرار نشده است. برای مثال در جدول زیر با شروع از موقعیت های ۱ و ۲ و ۳، کلمات DEN و ARE و MAT ، و با شروع از موقعیت های ۱ و ۴ و ۵ کلمات DAM و ERA و NET دیده می‌شوند.

^{1}D	^{2}A	^{3}M
^{4}E	R	A
^{5}N	E	T

بخشی از کار شما این است که یک جدول کلمات متقاطع را با استفاده از تکنیک حل مسئله‌ی CSP بسازید. برای این کار اول باید بتوانیم مسئله را به صورت درست بازنمایی کنیم. شما یک دیکشنری دارید که میتوانید از کلمات داخل آن در جدول خود استفاده کنید و دیکشنری شامل k کلمه می شود.$\{d_1,d_2,...,d_k\}$. فرض کنید که شما یک جدول با N مربع خالی و M کلمه ی متفاوت میسازید و البته زبان انگلیسی هم ۲۶ حرف دارد. مثلا در مثال بالا N=9 و M=6 میباشد.

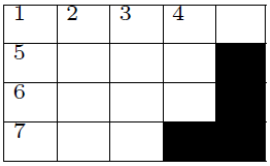
شما در ابتدا تصمیم می گیرید که کلمات، متغییر های شما در CSP باشد. D_1 نمایشی برای کلمه‌ی اول بالا به پایین و D_2 نمایشی برای کلمه ی دوم بالا به پایین می‌باشد. و همچنین A_1 نمایش برای اولین کلمه ی افقی باشد. مثلا در مثال بالا A_1 = DAM, D_1 = DEN, D_2 = ARE ... . همچنین فرض می کنید $D_1[i]$ نمایشی برای حرف i ام در کلمه‌ی اولِ بالا به پایین (D_1) باشد.

الف – اندازه‌ی فضای حالت برای این CSP چقدر می شود؟

ب - دقیقا و با نمایش ریاضی شرح دهید محدودیت‌های CSP وقتی که از کلمات به عنوان متغیر استفاده می کنیم چیست.

بعد از مشخص کردن CSP خود شما تصمیم گرفتید که جلو تر بروید و یک جدول کلمات متقاطع کوچک با استفاده از جدول زیر درست کنید . فرض کنید کلمات دیکشنری شما به شرح زیر است.

Crossword Grid



Dictionary Words

ARCS, BLAM, BEAR, BLOGS, LARD, LARP,
GAME, GAMUT, GRAMS, GPS, MDS, ORCS, WARBLER

جـ - می‌خواهیم بررسی کنیم که arc consistency تا چه حد می‌تواند دامنه را برای این مسئله محدود کند. حتی وقتی هیچ اختصاص دهی ای (assignment) انجام نشده باشد. مثلا فرض کنید هنوز هیچ متغیری اختصاص داده نشده است. محدودیت‌های unary را ابتدا اعمال کنید، و بعد arc consistency را اعمال کنید. (روی جدول پایین)

D_1	ARCS	BLAM	BEAR	BLOGS	LARD	LARP	GPS	MDS	GAME	GAMUT	GRAMS	ORCS	WARBLER
D_2	ARCS	BLAM	BEAR	BLOGS	LARD	LARP	GPS	MDS	GAME	GAMUT	GRAMS	ORCS	WARBLER
D_3	ARCS	BLAM	BEAR	BLOGS	LARD	LARP	GPS	MDS	GAME	GAMUT	GRAMS	ORCS	WARBLER
D_4	ARCS	BLAM	BEAR	BLOGS	LARD	LARP	GPS	MDS	GAME	GAMUT	GRAMS	ORCS	WARBLER
A_1	ARCS	BLAM	BEAR	BLOGS	LARD	LARP	GPS	MDS	GAME	GAMUT	GRAMS	ORCS	WARBLER
A_5	ARCS	BLAM	BEAR	BLOGS	LARD	LARP	GPS	MDS	GAME	GAMUT	GRAMS	ORCS	WARBLER
A_6	ARCS	BLAM	BEAR	BLOGS	LARD	LARP	GPS	MDS	GAME	GAMUT	GRAMS	ORCS	WARBLER
A_7	ARCS	BLAM	BEAR	BLOGS	LARD	LARP	GPS	MDS	GAME	GAMUT	GRAMS	ORCS	WARBLER

دوست شما به شما پیشنهاد می‌کند به جای کلمات، از حروف برای متغییر استفاده نمایید. برای شماره گذاری از سمت بالا چپ شروع کنید و به سمت راست بروید و بعد از آن از بالا به پایین حرکت کنید. یعنی در مثالی که ابتدای این سوال آوردیم: X_1=D, X_2=A و ... .

د – اندازه‌ی فضای حالت برای این CSP چقدر است؟