به‌نام خدا

تمرین ۲ – Adversarial Search and CSP

هوش مصنوعی

مهراد میلانلو

۹۹۱۰۵۷۷۵

سوال ۱.

آ) A, C, D, E

در صورت سوال کمی ابهام وجود دارد. زیرا نه دامنه‌ی متغیرها مشخص شده و نه می‌دانیم Constraintها چی هستند. بنابراین به‌طور قطعی نمی‌توان گفت در کدام راس‌ها «باید» Backtrack کرد اما با مثال‌هایی می‌توان نشان داد ممکن است هنگام تغییر مقدار بعضی متغیرها به بن‌بست بخوریم و مجبور به Backtrack باشیم.
همانطور که از گراف CSP مشخص است، متغیرهای B و F تنها با متغیر A در یک Constraints هستند. چون قبل از مقداردهی A عمل arc consistency را انجام می‌دهیم، یا قبل از مقداردهی A مجبور به Backtrack می‌شویم و یا با دادن هر مقداری به A، مقداری در دامنه‌ی B و F وجود دارد که Constraintها را نقض نمی‌کند. از آن‌جایی که B و F از متغیرهای دیگر مستقل هستند، پس در هر صورت در مقداردهی آن‌ها Backtrack رخ نمی‌دهد. به دلیل وجود دور میان متغیرهای A,C,E و همین‌طور A,D,E علی‌رغم اعمال arc consistency ممکن است Backtrack در هر کدام از این متغیرها رخ بدهد. متغیر G نیز چون در آخر مقداردهی می‌شود و قبل از آن arc consistency انجام می‌شود، نیازمند Backtrack نخواهد شد.

ب)

حکم به‌صورت شهودی بدیهی‌ست. زیرا با تغییر گره‌های مینیمم در ارتفاع h به گره‌های شانس، مقدار گره‌های ماکسیمم در ارتفاع h-1 یا بیشتر می‌شود و یا مقدار قبلی باقی می‌ماند. بنابراین مقدار هیچ گرهی در این درخت کمتر نمی‌شود. پس واضح است که مقدار minimax ریشه نیز کمتر نمی‌شود. برای اثبات دقیق‌تر، حکم را به استقرا روی ارتفاع درخت ثابت می‌کنیم.
می‌خواهیم ثابت کنیم با تغییر گره‌های مینیمم درخت بازی که مقدار minimax آن x است، به گره‌های شانس که مقدار expectimax درخت جدید y است، $y \geq x$.
حکم برای پایه‌ی استقرا یعنی ارتفاع ۱ و ۲ بدیهی‌است.(به همان شهودی که بالاتر اشاره شد)
حال فرض می‌کنیم حکم برای درخت‌های بازی به ارتفاع 1, 2, ..., h-1 درست باشد. اکنون درخت بازی به ارتفاع h را که ریشه‌ی آن گره ماکسیمم است در نظر بگیرید. فرزندان ریشه را $m_1, m_2, ..., m_k$

می‌نامیم. سپس برای هر m_i فرزندانش را $t_{i1}, t_{i2}, ..., t_{in_i}$ می‌نامیم. دقت کنید چون m_iها گره شانس هستند، بنابراین t_{ij}ها گره ماکسیمم هستند. طبق فرض استقرا در زیر درخت هایی که t_{ij}ها ریشه هستند، مقدار t_{ij}ها از مقدار آن‌ها در درخت مشابهی که به‌جای گره شانس دارای گره مینیمم است، بیشتر یا مساوی‌است. اگر T_{ij} مقدار گره t_{ij} در درخت جدید، متغیر تصادفی M_i مقدار گره m_i در درخت جدید و M'_i مقدار آن در درخت قدیم (درخت با گره‌های مینیمم) باشد، داریم:

$$E[M_i] \geq min(T_{i1},..., T_{in_i}) \Rightarrow M'_i \leq M_i \Rightarrow x = max(M'_1,..., M'_k) \leq max(M_1,..., M_k) = y$$
$$\Rightarrow x \leq y$$

بنابراین حکم اثبات می‌شود.

سوال ۲.

آ)

مقدار هر متغیر (طبقه‌ای که کارمند iام می‌رود) را با X_i نمایش می‌دهیم. با توجه به شروط Unary که در صورت سوال مطرح شده، برای دامنه‌ی متغیرها خواهیم داشت:

- $D(X_1) = \{5\}$
- $D(X_2) = \{1, 2, 3, 4, 6\}$
- $D(X_3) = \{1, 2, 3\}$
- $D(X_4) = \{1, 2, 3, 4, 5, 6\}$
- $D(X_5) = \{1, 2, 3\}$
- $D(X_6) = \{1, 3, 5\}$
- $D(X_7) = \{1, 2, 3, 4, 5\}$

ب)

در هیوریستیک MRV یا Minimum-Remaining-Values متغیری را ابتدا مقداردهی می‌کنیم که مقادیر کمتری در دامنه‌اش وجود داشته باشد. بنابراین طبق این هیوریستیک ابتدا X_1 را مقداردهی می‌کنیم.

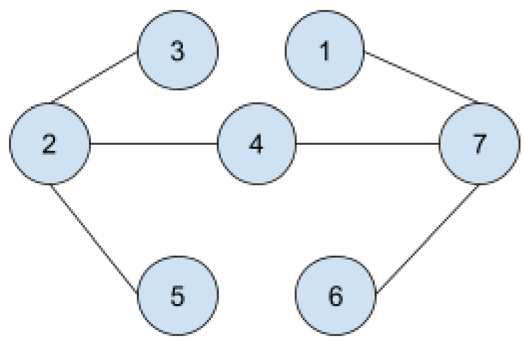
ج)

در هیوریستیک LCV یا Least-Constrainting-Value، به متغیر مورد نظر مقداری را می‌دهیم که شرط‌های کمتری را نقض کند. در واقع در Constraintهای کمتری با متغیرهای دیگر در تضاد باشد و به بقیه‌ی متغیرها

اجازه‌ی انتخاب مقادیر با تعداد بیشتری بدهد. در این‌جا X_6 تنها با X_7 در Constraint است. که طبق این شرط سعی می‌کنیم مقادیر بزرگتر را ابتدا به X_6 بدهیم. بنابراین اولویت مقداردهی ۵ و سپس ۳ است. اولویت آخر نیز ۱ است که چون باید از X_7 بزرگتر باشد، برای X_7 مقدار قابل قبولی نداریم.

د)

بین متغیرهایی که با یکدیگر Constraint دارند یال رسم می‌کنیم. بنابراین گراف محدودیت CSP به این شکل می‌شود:



شرط‌ها را هم طبق صورت سوال ذکر می‌کنیم:

$X_2 \neq X_4 \quad X_7 < X_6 \quad X_5 < X_2 \quad X_4 > X_7 \quad X_1 \neq X_7 \quad X_2 < X_3$

یک جواب قابل قبول نیز برای مسئله این چنین است:

$X_1 = 5,\ X_2 = 2,\ X_3 = 3,\ X_4 = 4,\ X_5 = 1,\ X_6 = 3,\ X_7 = 1$

سوال ۳.

آ)

چون متغیر A با متغیرهای B,C,D همسایه بوده و Constraint دارند، ممکن است دامنه‌ی آن‌ها بعد از forward checking تغییر کند اما الزامی ندارد!

ب)

بعد از مقداردهی متغیر B تنها همسایه‌های آن که هنوز مقداردهی نشده‌اند ممکن است بعد از forward checking دامنه‌شان تغییر کند. از آن‌جایی که A پیش از این مقداردهی شده، پس تنها C,E ممکن است دچار تغییر در دامنه‌شان بشوند.

ج)

بعد از مقداردهی متغیر A، ممکن است دامنه‌ی هر یک از همسایه‌هایش تغییر کند. بنابراین هنگامی که Arc Consistency را اعمال می‌کنیم، هربار راس‌هایی که به A یک مسیر دارند ممکن است دامنه‌شان تغییر کند. زیرا در هر مرحله‌ی Arc Consistency اگر یک مقدار از دامنه‌ی یک متغیر حذف شود، تمام متغیرهای همسایه‌ی آن را به صف اضافه می‌کنیم و Arc Consistency را روی متغیرهای این صف اعمال می‌کنیم. بنابراین متغیرهای B,C,D,E,F ممکن است دچار تغییر در دامنه‌شان بشوند.

د)

بعد از مقداردهی متغیر A و اعمال Arc Consistency، اگر نیاز به Backtrack نباشد، یعنی در دامنه‌ی تمام متغیرها از جمله D حداقل یک مقدار هست که با انتخاب آن، همسایه‌های آن متغیر نیز مقدار مجاز برای انتخاب داشته باشند. بنابراین حتما مقدار مجازی برای D وجود دارد. حال متغیر B را مقداردهی می‌کنیم و طبق قسمت قبلی می‌توان گفت تمام راس‌هایی که به B مسیر دارند ممکن است دامنه‌شان تغییر بکند. اما از آن‌جایی که فرض می‌کنیم Backtrack رخ نداده و مقدار متغیر A انتخاب شده و ثابت شده، تغییری نمی‌کند. چون متغیر D تنها با A در Constraint است، پس دامنه‌ی آن نیز تغییر نمی‌کند. پس متغیرهای C,E,F ممکن است دچار تغییر در دامنه‌شان شود.

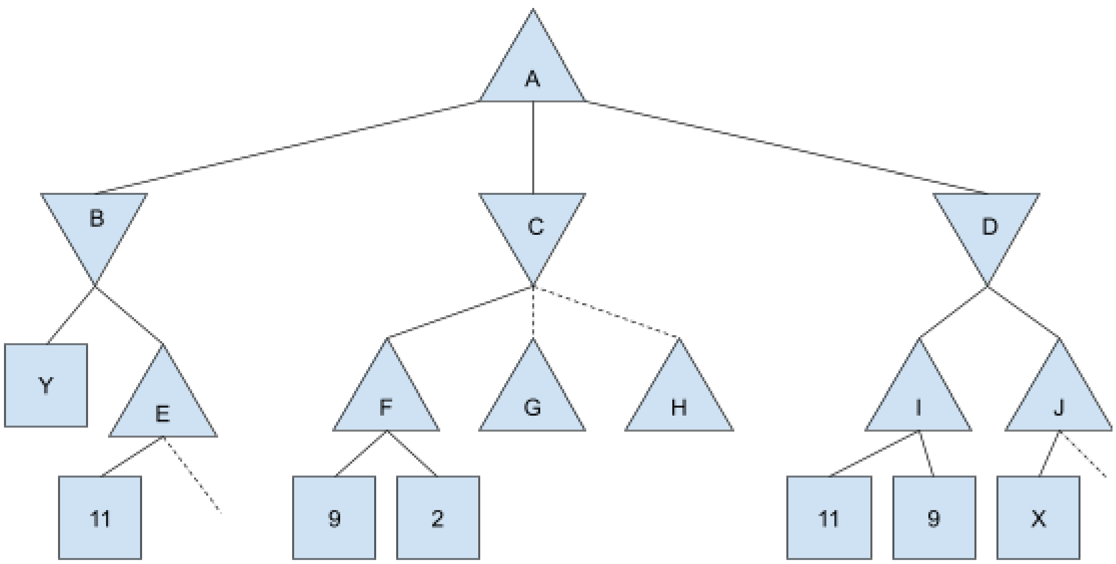
سوال ۴.

دقت کنید که هرس از سمت چپ شروع می‌شود. بنابراین به‌ترتیب از چپ گره‌های B, Y, E, 11 بررسی می‌شوند. چون گره B مینیمم است بنابراین $Y \leq 11$ که شاخه‌ی سمت راست E هرس شده است. در نهایت B=Y.

واضح است که F=9. از آن‌جایی که A گره ماکسیمم است و در شاخه‌ی سمت راست C هرس داریم، به این معناست که $B \geq F$. بنابراین مقدار G و H هرچه باشند هرس می‌شوند. بنابراین $Y \geq 9$.

واضح است که I=11. چون گره D مینیمم است و شاخه سمت راست J هرس شده، می‌توان فهمید که $X \geq 11$. زیرا شاخه‌ی راست J در این‌صورت تفاوتی ندارد چه باشد. مقدار D در این صورت از گره I انتخاب می‌شود. ($J \geq I$)

در نهایت دقت کنید اگر $Y = 11$ باشد، بعد از تعیین مقدار I، گره J هرس می‌شود که بر خلاف فرض صورت سوال است. بنابراین $Y < 11$.



در نهایت داریم:

$Y \in [9,\ 10]\ ,\ X \in [11,\ \infty]$

سوال ۵.

آ)

می‌خواهیم Expected utility را برای در حالتی محاسبه کنیم که با دادن هزینه‌ی c=1، بتوانیم مقدار زیردرخت سمت راست را ببینیم. چون احتمال ظاهر شدن هر یک از مقادیر ذکر شده برابر است، داریم:

$E(Utility|State = 3) = \frac{1}{3}(21 + 12 + 12) - c = 15 - c = 14$

ب)

مسئله به سه حالت تقسیم می‌شود:

۱. همواره سمت راست را انتخاب کنیم. در این حالت مقدار مورد انتظار utility برابر است با:

$E(Utility|State = 1) = \frac{1}{3}(21 - 8 - 4) = 3$

۲. همواره سمت چپ را انتخاب کنیم. در این حالت بدیهتا داریم:

$E(Utility|State = 2) = 12$

۳. در این حالت نیز با پرداخت هزینه‌ی c سمت راست را انتخاب کرده و درخت بازی تبدیل به minimax می‌شود که یعنی مقدار بیشتر بین سمت راست و چپ را انتخاب می‌کنیم. همانطور که در قسمت قبل به‌دست آوردیم، $E(Utility|State = 3) = 15 - c$.

چون می‌خواهیم بیشترین Utility را داشته باشیم، پس در صورتی حالت سوم را انتخاب می‌کنیم که مقدار مورد انتظار برای utility آن بیشتر از حالت‌های دیگر باشد. یعنی:

$15 - c \geq 12 \Rightarrow c \leq 3$

بنابراین تا وقتی $c \leq 3$، منطقی‌است این حالت را انتخاب کنیم.