زهرا توکلی 40120063 «بسمه تعالی» تکلیف دوم هوش مصنوعی

مدل سازی

1.1: مربع لاتین متعامد: می‌توانیم هر درایه‌ی مربع لاتین را یک متغیر درنظر بگیریم. بنابراین:

$$A=\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix}, \quad B=\begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} & b_{15} \\ b_{21} & b_{22} & b_{23} & b_{24} & b_{25} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{51} & b_{52} & b_{53} & b_{54} & b_{55} \end{bmatrix}$$

دامنه‌ی هر متغیر از 1 تا n (اینجا 1 تا 5) است. قیود شامل قیدهای alldifferent روی سطرها و ستون های هر مربع و یک قید به صورت $a_{ij} \neq a_{rs}$ or $b_{ij} \neq b_{rs}$ برای این است که همه‌ی زوج های صورت (a_{ij}, b_{ij}) یکتا باشند.

متغیرها: $X=\{a_{11}, a_{12}, \dots, a_{55}, b_{11}, b_{12}, \dots, b_{55}\}$

دامنه ها: $D=\{\{1..5\}, \{1..5\}, \dots, \{1..5\}\}$

قیود: for $i=1..5$ for $j=1..5$ alldifferent(a_{ij}) → در تمام سطرهای A هر عدد از 1 تا 5 یکبار تکرار بشود

for $i=1..5$ for $j=1..5$ alldifferent(b_{ij}) → در تمام سطرهای B هر عدد از 1 تا 5 یکبار تکرار بشود

for $j=1..5$ for $i=1..5$ alldifferent(a_{ij}) → در تمام ستون های A هر عدد از 1 تا 5 یکبار تکرار بشود

for $j=1..5$ for $i=1..5$ alldifferent(b_{ij}) → در تمام ستون های B هر عدد از 1 تا 5 یکبار تکرار بشود

for $i=1..5$ for $j=1..5$ for $r=i..5$ for $s=j..5$ $a_{ij} \neq a_{rs}$ or $b_{ij} \neq b_{rs}$ → اگر $i \neq j$ or $r \neq s$ باید این شرط برقرار باشد

قیود لازم برای اینکه دو مربع لاتین متعامد باشند.

1.3: جمعیت شهرها: متغیرها را جمعیت شهرها درنظر می‌گیریم. جمعیت شهر iام را با x_i نمایش می‌دهیم. به ازای هر دو شهر i و j که با جاده به هم متصل شده اند، قیدی به صورت $|x_i - x_j| \geq 2000$ داریم که جمعیت هر دو شهر مجاور حداقل 2000 نفر اختلاف داشته باشد. به ازای هر دو شهر i و j یک قید به صورت $x_i \leq 3x_j$ داریم که در فرآیند حل CSP اگر هر شهری پرجمعیت ترین شهر شد جمعیت آن از سه برابر کم جمعیت ترین شهر بیشتر نباشد. (به عنوان مثال فرض کنیم x_n پرجمعیت ترین شهر و x_1 کم جمعیت ترین شهر شود پس علاوه بر قید $x_n \leq 3x_1$، قیودی به شکل $x_n \leq 3x_i$ و ... هم برقرار اند چون اگر x_n از 3 برابر x_1 که کم جمعیت ترین شهر است کم تر باشد از 3 برابر جمعیت شهرهای دیگر هم کمتر است و وقتی x_n (پرجمعیت ترین شهر) از 3 برابر کم جمعیت ترین کمتر باشد قیودی به شکل $x_i \leq 3x_1$ هم مشکلی در ارضا شدن CSP ایجاد نمی‌کند. علاوه براین قیودی به شکل $x_1 \leq 3x_n$ یا $x_r \leq 3x_s$ که $x_r \leq x_s$ هم مشکلی ایجاد نمی‌کند.)

1.2: پوشاندن مستطیل‌ها: برای هر مستطیل کوچک i، دو متغیر x_i و y_i را داریم که مختصات شروع آن مستطیل کوچک (نقطهٔ پایین سمت چپ آن) نسبت به مختصات شروع مستطیل بزرگ است. دامنهٔ هر متغیر x_i را می‌توانیم به صورت $\{0, 1, 2, \ldots, W-w_i\}$ در نظر بگیریم. یعنی x_i حداکثر $W-w_i$ می‌تواند باشد تا وقتی در مستطیل بزرگ قرار می‌گیرد، از آن بیرون نزند. دامنهٔ هر متغیر y_i هم می‌توانیم $\{0, 1, 2, \ldots, H-h_i\}$ در نظر بگیریم تا مستطیل کوچک از مستطیل بزرگ بیرون نزند. (می‌توانیم هم دامنه‌های x_iها را از $\{0, \ldots, W\}$ و دامنهٔ y_iها را $\{0, \ldots, H\}$ در نظر بگیریم و قیودی به صورت $x_i + w_i \le W$ و $y_i + h_i \le H$ داشته باشیم تا تمام مستطیل‌های کوچک حتماً در مستطیل بزرگ قرار گیرند.) از آنجایی که نمی‌دانیم در چینش نهایی کدام مستطیل‌ها با هم همسایه می‌شوند باید هر دو مستطیل کوچک با هم overlap نداشته باشند. برای هر دو مستطیل i و j اگر آنها overlap نداشته باشند یا یکی بالای دیگری است یا پایین آن یا سمت چپ آن یا سمت راست آن و یا ترکیبی از آنها مثلاً بالا و راست آن یا و... . پس یک سری قید به ازای هر دو مستطیل i و j به صورت زیر داریم:

$$\underbrace{x_i + w_i \le x_j}_{i \text{ سمت چپ } j} \text{ or } \underbrace{x_j + w_j \le x_i}_{i \text{ سمت راست } j} \text{ or } \underbrace{y_i + h_i \le y_j}_{i \text{ پایین } j} \text{ or } \underbrace{y_j + h_j \le y_i}_{i \text{ بالای } j}$$

1.4: وزرای گیج شده

برای هر ستون یک متغیر x_i در نظر می‌گیریم که نشان دهندهٔ شمارهٔ سطری است که وزیر ستون iام در آن قرار می‌گیرد. پس دامنهٔ هر x_i برابر مجموعهٔ $\{1, 2, \ldots, n\}$ است. برای اینکه هر دو وزیر i و j یکدیگر را تهدید سطری یا قطری نکنند (چون تهدید ستونی نمی‌توانیم داشته باشیم) باید نظیر هر i و j از 1 تا n قیودی به صورت زیر داشته باشیم:

$$\underbrace{x_i = x_j}_{\text{تهدید سطری}} \text{ or } \underbrace{x_i - i = x_j - j \text{ or } x_i + i = x_j + j}_{\text{تهدید قطری}}$$

با استفاده از CSP Solver ها و مدل کردن این مسئله به صورت بالا همهٔ جواب‌های ممکن مسئله هم می‌توانیم به دست بیاوریم. کافی است با رسیدن به یک جواب الگوریتم backtrack را متوقف نکنیم و همهٔ حالات را بررسی کنیم.

2 اثبات

2.1: خیر. از برقراری k+1 consistency همیشه نمی‌توان k_consistency را نتیجه گرفت.

مثال نقض: یک CSP با دو متغیر X و Y و هر یک با دامنهٔ {1و2و3} و قیود زیر را در نظر می‌گیریم: unary

$X<3$, $Y<3$, $X\neq Y \rightarrow$ binary constraint

این شبکه arc-consistent هست چون به ازای هر مقداری که به X یا Y بدهیم مقداری برای متغیر دیگر از دامنهٔ اش باقی می‌ماند ولی شبکه node consistent نیست و X و Y می‌توانند مقدار 3 هم بگیرند.

2.2:

تعداد قیود روی n متغیر با اندازهٔ دامنهٔ k برابر 2^{k^n} است: اگر مقادیر این n متغیر را به صورت n تایی مرتب در نظر بگیریم برای مؤلفهٔ اول k حالت، مؤلفهٔ دوم k حالت و ... داریم پس تعداد n تایی های مرتب برابر k^n است. حال اگر قیود را explicit در نظر بگیریم یک قید می‌تواند به صورت هر زیرمجموعه‌ای از این k^n، n تایی مرتب طرح شود و تعداد این زیرمجموعه ها 2^{k^n} است. پس 2^{k^n} قید مختلف می‌توانیم روی این n متغیر با اندازهٔ دامنهٔ k مطرح کنیم.

تعداد شبکه های محدودیت دوتایی روی این n متغیر $2^{k^2n^2}$ است:

تعداد زوج های از این n متغیر به صورت (x_i, x_j) اگر زوج های مثل (x_i, x_i) هم در نظر بگیریم برابر n^2 است.

چون دامنهٔ هر متغیر k عضو دارد پس تعداد زوج های مرتب که می‌توانند مقادیر نظیر هر (x_i, x_j) را نشان دهد k^2 است. پس تعداد کل اینگونه زوج ها k^2n^2 می‌شود و هر کدام از این زوج مرتب ها می‌توانند در شبکهٔ محدودیت باینری باشند یا نباشند. پس تعداد این شبکه های محدودیت باینری برابر $2^{k^2n^2}$ است.

Consistency 3

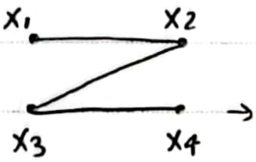
3.2: شبکه محدودیت باینری:

$D_1=\{1,2,3,4\}$ $D_2=\{3,4,5,8,9\}$ $D_3=\{2,3,5,6,7,9\}$

$D_4=\{3,5,7,8,9\}$

$C_1: x_1 \geq x_2$ $C_2: x_2 > x_3$ or $x_3 - x_2 = 2$ $C_3: x_3 \neq x_4$

3.2.1: Constraint graph:



هر دو متغیری که باهم در یک قید شرکت کرده اند را به هم متصل می کنیم.

3.2.2:

خیر. شبکه arc-consistent نیست. زیرا مثلاً اگر به x_1 مقدار 1 را دهیم برای x_2 مقداری از دامنه اش باقی نمی ماند که $x_1 \geq x_2$ باشد.

محاسبه شبکه arc-consistent:

مقادیر 1 و 2 از دامنه x_1 حذف می شوند چون در غیر این صورت برای x_2 مقداری از دامنه اش باقی نمی ماند که قید $x_1 \geq x_2$ برقرار باشد: $D_1=\{3,4\}$ هم چنین مقادیر 5 و 8 و 9 از دامنه x_2 حذف می شوند چون در غیر این صورت برای x_1 مقداری از دامنه اش باقی نمی ماند که $x_1 \geq x_2$: $D_2=\{3,4\}$

مقادیر 7 و 9 از دامنه x_3 حذف می شوند چون در غیر این صورت برای x_2 مقداری از دامنه اش باقی نمی ماند که قید $x_2 > x_3$ or $x_3 - x_2 = 2$ برقرار باشد: $D_3=\{2,3,5,6\}$

قید $x_3 \neq x_4$ دامنه های x_3 و x_4 را تغییر نمی دهد و این قید arc-consistency را دارد.

پس دامنه های نهایی متغیرها به صورت زیر اند:

$D_1=\{3,4\}$ $D_2=\{3,4\}$ $D_3=\{2,3,5,6\}$ $D_4=\{3,5,7,8,9\}$

3.2.3: چون constraint graph مسئله به صورت درخت است می توانیم آن را بدون backtrack حل کنیم: ابتدا درخت constraint graph را با topological sort مرتب می کنیم و سپس از سمت راست حرکت کرده و آن را arc consistent می کنیم و سپس اگر دامنه هیچ متغیری تهی نشد از سمت چپ حرکت کرده و مقادیری را به متغیرها تخصیص کنیم تا یک مقدار دهی ارضاکننده به متغیرها پیدا کنیم.

Sort: x_1 — x_2 — x_3 — x_4

$D_4=\{3,5,7,8,9\}$

$D_3=\{2,3,5,6,7,9\}$ تغییری نمی کند

$D_2=\{3,4,5,8,9\}$

$D_1=\{3,4\}$ 1 و 2 حذف می شوند

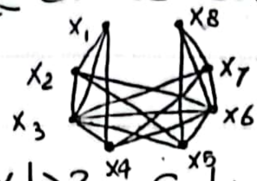
چون دامنه هیچ متغیری تهی نشد پس مسئله ارضاپذیر است. یک مقداردهی ارضاکننده به صورت زیر به دست می آید:

برای x_1 مقدار 3 را انتخاب می کنیم. برای x_2 هم باید مقدار 3 از دامنه اش را انتخاب کنیم تا C_1 برقرار باشد. برای x_3 مقدار 2 یا 5 را می توانیم انتخاب کنیم تا C_2 برقرار باشد. اگر برای x_3 مقدار 5 را انتخاب کنیم برای x_4 مقدار 9 را می توانیم انتخاب کنیم تا C_3 برقرار باشد پس $(x_1,x_2,x_3,x_4)=(3,3,5,9)$

	x_2	x_5	
x_1	x_3	x_6	x_8
	x_4	x_7	

3.3: جدول: هرخانهٔ جدول را یک متغیر درنظر می گیریم: دامنهٔ هر متغیر اعداد 1 تا 8 است.

3.3.1: Constraint graph →



Constraints:

$C_1: |x_1-x_2| \geq 2$ $C_2: |x_1-x_3| \geq 2$ $C_3: |x_1-x_4| \geq 2$ $C_4: |x_2-x_3| \geq 2$ $C_5: |x_2-x_5| \geq 2$

$C_6: |x_2-x_6| \geq 2$ $C_7: |x_3-x_4| \geq 2$ $C_8: |x_3-x_5| \geq 2$ $C_9: |x_3-x_6| \geq 2$ $C_{10}: |x_3-x_7| \geq 2$

$C_{11}: |x_4-x_6| \geq 2$ $C_{12}: |x_4-x_7| \geq 2$ $C_{13}: |x_5-x_6| \geq 2$ $C_{14}: |x_5-x_8| \geq 2$ $C_{15}: |x_6-x_8| \geq 2$

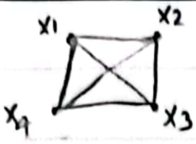
$C_{16}: |x_7-x_8| \geq 2$ $C_{17}: |x_7-x_6| \geq 2$

3.3.2: بله شبکه arc consistent هست چون برای هر دو متغیر، هر مقداری از دامنه شان به آنها بدهیم برای متغیر دیگر مقادیری باقی می ماند تا قید بین آنها برقرار باشد.

3.3.3: بله این شبکه ارضا پذیر است. چون اگر اعداد 8، 7، 6، 5 و 4 و 3 و 2 و 1 را درنظر بگیریم به x_3 باید عدد 1 یا 8 را بدهیم چون اگر مثلاً عدد 2 را بدهیم اعداد 1 و 3 را چون کمتر از دو واحد با آن اختلاف دارند نمی توان به همسایه هایش داد. پس 5 عدد 4 و 5 و 6 و 7 و 8 برای همسایه ها باقی می ماند که چون 6 همسایه دارد کافی نیست. اگر عدد 1 یا 8 مثلاً عدد 1 را به x_3 بدهیم، به همسایه ها الزاماً باید اعداد 3 تا 8 (که 6 تا هستند) را بدهیم تا شرط بین آنها و x_1 برقرار باشد. پس x_6 مقداری بین 3 تا 8 خواهد داشت. اگر مقدار آن بین 3 تا 7 باشد مقادیر مجاز برای همسایه های آن به جز x_3 که مقدارش قبلاً تعیین شده برابر 4 مقدار خواهد بود در صورتی که به جز x_3، 5 همسایهٔ دیگر دارد. پس به x_6 هم مقدار 8 را می دهیم. چون $x_3=1$ و هیچ کدام از همسایه هایش نمی توانند 2 شوند الزاماً باید $x_8=2$ شود و به همین شکل $x_1=7$ می شود. اعداد باقی مانده یعنی 3 و 4 و 5 و 6 هم باید به گونه ای قرار دهیم 3 با 4، 4 با 5 و 5 با 6 همسایه نباشند. پس یک مقداردهی ارضاکننده به صورت زیر به دست می آید:

	4	6	
7	1	8	2
	3	5	

Subject: Year Month Date ( )



3.1: رنگ آمیزی گراف: $D_1=D_2=D_3=D_4=\{r,g,b\}$

به این مسئله arc-consistent هست چون به هر متغیر هر مقداری از دامنه اش بدهیم برای متغیرهای دیگر مقداری از دامنه‌شان باقی می‌ماند تا قیود بین آنها برقرار باشد.
به path-consistent هم هست چون به هر دو متغیر که مقدار دهیم طوری که قید بین آن دو برقرار باشد و همرنگ نباشند برای سومی هم رنگی باقی می‌ماند که با دوتای دیگر همرنگ نباشد (3 رنگ داریم).
خیر. 4-consistent نیست چون به هر سه‌تایی مقدار دهیم طوری که هیچ دوتایی همرنگ نباشند، هر 3 رنگ مصرف می‌شوند و برای 4ای که به هر سه‌تای قبلی با یالی وصل است رنگ دیگری نمی‌ماند تا با آنها غیرهمرنگ باشد. این مسئله را نمی‌توان قویاً 4-consistent کرد، زیرا در آن صورت دیگر بدون نیاز به backtrack مسئله حل می‌شود و به وضوح می‌توانیم ببینیم که این مسئله با 3 رنگ ارضاپذیر نیست. پس چه با قیود باینری یا قیود دیگر نمی‌توان آن را 4-consistent کرد.

	1	2	3	4
1	O			
2		O		
3			O	
4				O

$\to \binom{4}{2}=6$ تهدید

حل 4 با Backtracking یا Local Search

4.1: چهار وزیر: نظیر وزیر ستون i ام متغیر x_i را در نظر می‌گیریم. در حالت اولیه وزیرها روی قطر اصلی اند و هر دو وزیر یکدیگر را تهدید می‌کنند. دامنهٔ متغیرها اعداد 1 تا 4 اند و قیود به این صورت تعریف می‌شوند که هیچ دو وزیری یکدیگر را تهدید سطری یا قطری نکنند. روش MCH از بین متغیرهایی که قیودی را نقض می‌کنند یکی را به تصادف انتخاب می‌کند و مقدار آن را به گونه‌ای تغییر می‌دهد که بیش‌ترین تعداد قید ارضا شوند و این کار را تا زمانی که به ماکزیمم تعداد تکرارها برسد ادامه می‌دهد. مثلاً ما ابتدا متغیر x_1 را انتخاب می‌کنیم:

$x_1=2 \to$ conflicts=4 , $x_1=3 \to$ conflicts=5 , $x_1=4 \to$ conflicts=4 $\to$ $x_1=2$ را تصادفاً انتخاب می‌کنیم

اینبار از بین متغیرهای نقض‌کننده قیود x_4 را به صورت تصادفی انتخاب می‌کنیم:

$x_4=1 \to$ conflicts=2 , $x_4=2 \to$ conflicts=5 , $x_4=3 \to$ conflicts=3 $\to$ $x_4=1$ را انتخاب می‌کنیم.

از بین x_2 و x_3 که قیود را نقض کرده‌اند x_3 را تصادفاً انتخاب می‌کنیم:

$x_3=1 \to$ conflicts=3 , $x_3=2 \to$ conflicts=4 $x_3=4 \to$ conflicts=2 $\to$ $x_3=4$ را انتخاب می‌کنیم.

سپس x_1 را انتخاب می‌کنیم: $x_1=1 \to$ conflicts=2 , $x_1=3 \to$ conflicts=1, $x_1=4 \to$ conflicts=2 $\to$ $x_1=3$ را انتخاب می‌کنیم.

سپس x_2 تصادفاً انتخاب می‌شود:

$x_2=1 \to$ conflicts=1 , $x_2=3 \to$ conflicts=3 , $x_2=4 \to$ conflicts=2 $\to$ پس $x_2=1$ انتخاب می‌شود.

سپس از بین x_2 و x_4 که هم را تهدید می‌کنند x_4 تصادفاً انتخاب می‌شود:

$x_4=2 \to$ conflicts=0 $\to$ پس جواب مسئله پیدا شد. $\to$ بورد نهایی:

	O		
			O
O			
		O	

GERALD
+DONALD
ROBERT

4.2: Puzzle Cryptarithmetic: متغیرها را حروف G و E و R و A و L و D و O و N و B و T در نظر می‌گیریم که دامنه‌ی همه به جز D و G و R اعداد 0 تا 9 و دامنه‌ی D و G و R که نمی‌توانند صفر باشند اعداد 1 تا 9 است. حین backtrack، هیوریستیک MRV می‌گوید از بین متغیرهایی که مقداردهی نشده‌اند متغیری را انتخاب کنید که دامنه‌اش کمترین تعداد عضو را دارد. پس ما از بین متغیرهای R و G و D، R را تصادفاً برای سبط دادن انتخاب می‌کنیم. هیوریستیک LCV می‌گوید پس از انتخاب متغیر بعدی ابتدا مقداری از آن را سبط بده که بیش‌ترین تعداد مقدار را برای متغیرهای دیگر بگذارد. مثلاً اگر مقدار 1 برای R را در نظر بگیریم برای G و D مقداری از دامنه‌شان باقی می‌ماند طوری که قید ارضا شوند چون G و D اگر حداقل 1 باشند، R نمی‌تواند 1 شود. پس برای سبط R از مقدار 1 یا 2 یا ... شروع نمی‌کنیم چون با احتمال بیش‌تری مجبور به بازگشت به عقب می‌شویم بلکه بهترین مقدار برای گسترش عدد 9 است. پس از انتخاب 9 برای R، consistency ها را مثلاً arc consistency برقرار می‌کنیم. طی این فرآیند ممکن است دامنه‌ی برخی متغیرها کاهش یابد. سپس دوباره الگوریتم backtrack را روی این شبکه‌ی consistent اجرا می‌کنیم و این الگوریتم دوباره ابتدا متغیری را برای سبط دادن انتخاب می‌کند که دامنه‌اش کمترین تعداد عضو را داشته باشد از MRV و LCV استفاده می‌کند. اگر این backtrack جواب ارضاکننده‌ای پیدا کند پس جواب مسئله با R=9 پیدا می‌شود و اگر پیدا نکند باید از R=9 یک بازگشت به عقب داشته و با LCV مقدار بعدی سبط R را انتخاب کنیم و اگر مسئله ارضاپذیر باشد بالاخره با یکی از مقادیر 1 تا 9 برای R به حل مسئله می‌رسیم.

قیود مسئله:

C_1: alldifferent(A, R, E, G, L, D, O, N, B, T)

$$C_2: D + 10L + 100A + 1000R + 10000E + 100000G + D + 10L + 100A + 1000N + 10000O + 100000D$$
$$= T + 10R + 100E + 1000B + 10000O + 100000R$$