به نام خدا

تمرین دوم درس هوش مصنوعی

استاد: دکتر رهبان

نویسنده: سید علیرضا میررکنی

شماره دانشجویی: ۴۰۱۱۰۶۶۱۷

دانشکده مهندسی کامپیوتر دانشگاه صنعتی شریف – بهار ۱۴۰۳

سوال ۱:

الف) نادرست. در حالت کلی، جست و جو ممکن است مجبور به بررسی کردن (تقریبا) تمامی assignment های ممکن شود که تعداد آن ها برابر d^n می باشد. heuristic های MRV و LCV معمولا به بهبود زمان جست و جو و کاهش تعداد assignment های بررسی شده کمک می کنند، اما تضمینی وجود ندارد که اگر از آنها استفاده کنیم تعداد backtracking ها در بدترین حالت (worst case) کاهش یابد. پس حداکثر تعداد دفعات backtracking در صورت استفاده از arc consistency و heuristic های MRV و LCV، $O(d^n)$ می باشد.

ب) نادرست. پیچیدگی زمانی حل کننده کارا در این حالت بر حسب n و d (تعداد مقادیر مجاز برای متغیر ها)، $O(nd^2)$ می باشد (بر حسب n به تنهایی، $O(n)$ است). روش حل به این صورت است که در ابتدا با انتخاب یک متغیر به عنوان ریشه، الگوریتم topological sort را روی رئوس اجرا می کنیم. سپس با شروع از آخرین راس (برگ)، در هر مرحله یال بین راس فعلی و راس قبلی (در ترتیب حاصل از topological sort) را consistent می کنیم و به راس قبلی می رویم. در نهایت با شروع از ریشه و مقدار دهی دلخواه به آن از میان مقادیر مجاز، در هر مرحله مقداری را (از میان مقادیر مجاز) برای راس فعلی انتخاب می کنیم که با راس قبلی (در ترتیب حاصل از topological sort) consistent باشد.

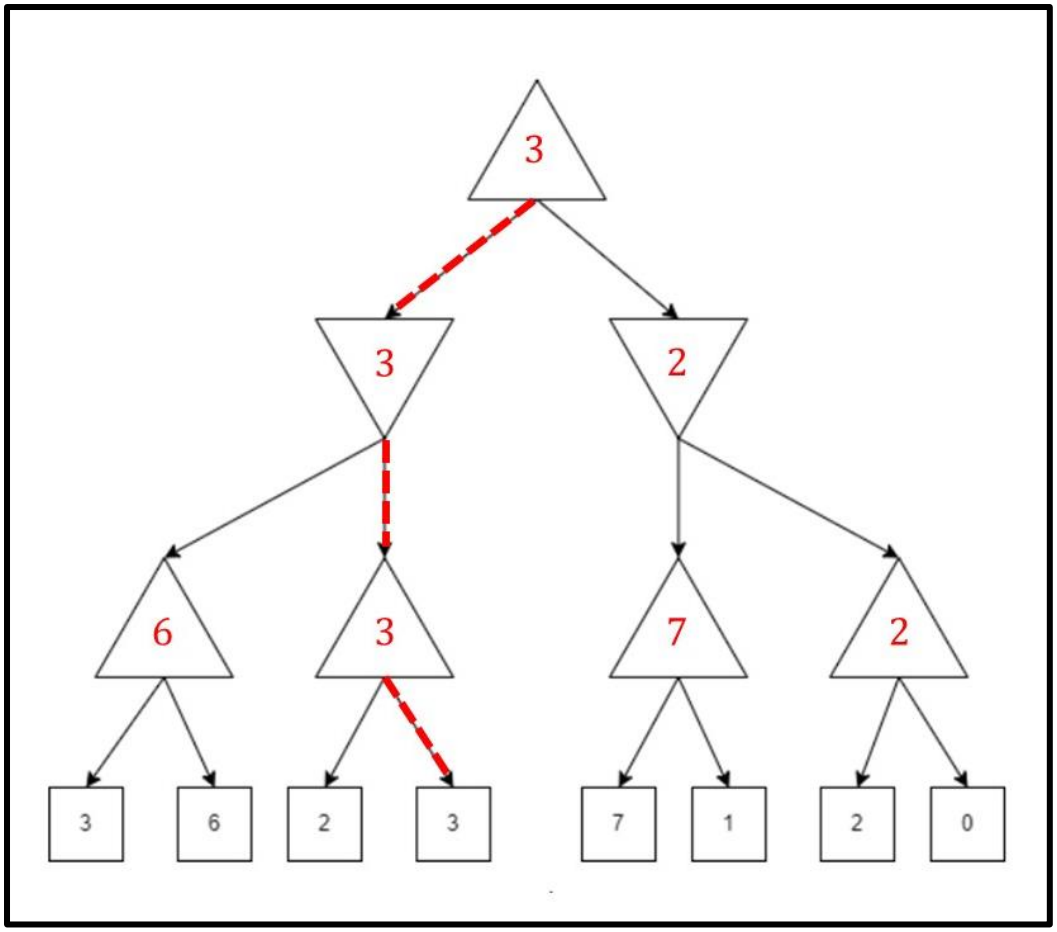
ج) نادرست. هرس آلفابتا هیچ گونه تاثیری بر روی جواب به دست آمده برای ریشه درخت با minimax ندارد. چرا که هم در صورت استفاده از این الگوریتم و هم در صورت عدم استفاده از آن، مقداری که برای ریشه درخت به دست می آید برابر با جواب بهینه می باشد که مقدار ثابتی دارد و تغییر نمی کند (به عبارت دیگر استفاده از الگوریتم هرس آلفابتا بهینگی minimax را خراب نمی کند).

د) درست. اگر یک تابع اکیدا صعودی روی برگ های درخت اعمال کنیم، ترتیب برگ ها (از نظر مقدار) تغییری نمی کند (چرا که طبق تعریف اکیدا صعودی بودن برای تابع F، داریم: $F(x) \leq F(y) \Leftrightarrow x \leq y$). در نتیجه، در فرایند محاسبه مقادیر رئوس درخت minimax، نحوه انتخاب مقدار یک راس از بین مقادیر فرزندانش تغییر نمی کند (یعنی اگر قبل از اعمال این تابع بر روی برگ ها، مقدار راس x از فرزند y آن گرفته می شد، پس از اعمال تابع نیز مقدار راس x از y گرفته خواهد شد، چرا که اعمال این تابع ترتیب فرزندان راس x را تغییر نخواهد داد) و در نتیجه پاسخ بهینه آن تغییر نمی کند (دقت کنید عددی که در نهایت روی ریشه نوشته می شود به وضوح ممکن است با اعمال این تابع تغییر کند، اما مسیری که به عنوان جواب بهینه در درخت بازی یافت می شود، تغییر نخواهد کرد).

ه) درست. طبق آنچه که پیش تر در قسمت د گفته شد، با اعمال این تابع اکیدا صعودی بر روی برگ ها، نحوه انتخاب مقدار یک راس از بین مقادیر فرزندانش و همچنین ترتیب کلی رئوس درخت (از نظر مقدار) تغییری نخواهد کرد. در نتیجه رئوسی که در درخت اولیه به خاطر کوچک تر شدن والدشان از α هرس می شدند، در درخت جدید نیز حتما مقدار والدشان کوچک تر از α جدید خواهد شد و هرس می شوند. با استدلالی مشابه، رئوسی که در درخت اولیه والدشان از β بزرگ تر می شد و هرس می شدند، در درخت جدید نیز هرس خواهند شد. همچنین رئوسی که در درخت اولیه هرس نمی شدند، در درخت جدید نیز هرس نمی شوند، چرا که بزرگ تر یا کوچک تر بودن مقدار والد آن ها نسبت به α و β تغییر نمی کند (یعنی مثلا راسی وجود ندارد که در درخت اولیه، والدش از α بزرگ تر باشد و هرس نشود، اما در درخت جدید مقدار والدش از α کوچک تر شده و هرس شود).

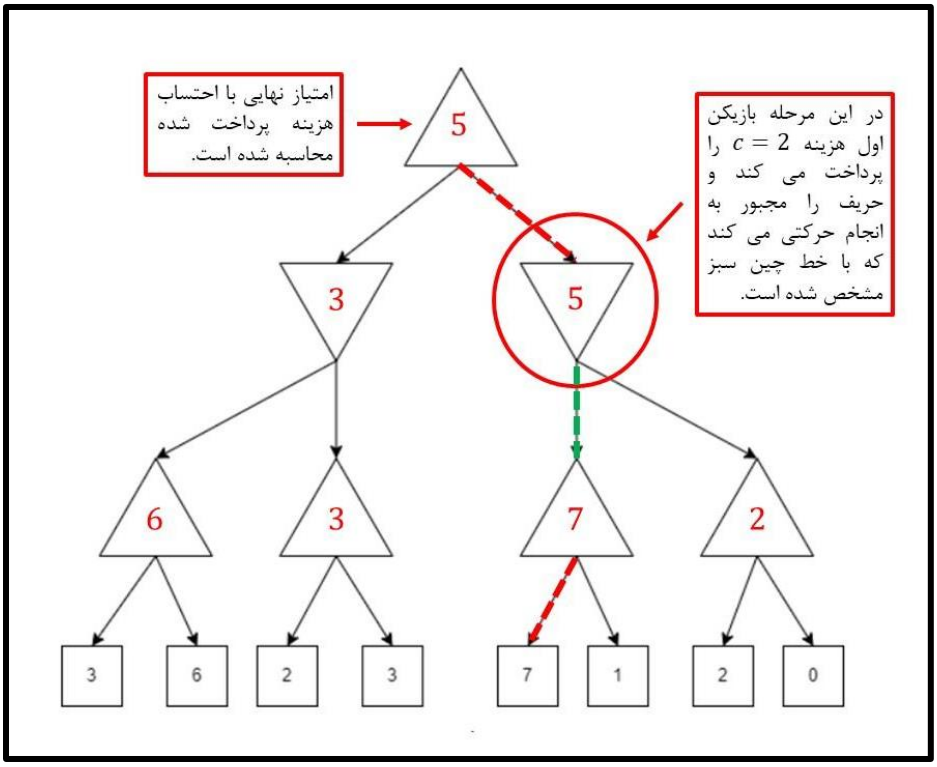
سوال ۲:

الف) درخت کامل شده، به صورت زیر می باشد:

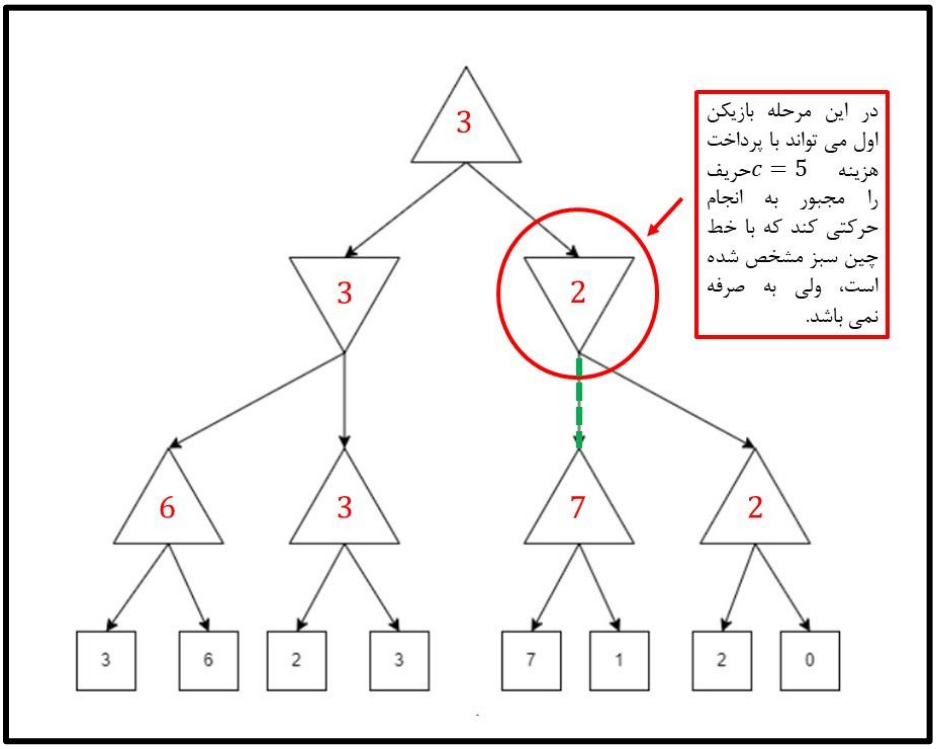


دقت کنید که روئوس به شکل △ رئوس max (مقدار آنها برابر بیشینه مقادیر فرزندانشان می شود) و رئوس به شکل ▽ رئوس min (مقدار آنها برابر کمینه مقادیر فرزندانشان می شود) هستند.

ب و ج) با دریافت این قابلیت ویژه، حداکثر امتیازی که بازیکن اول می تواند به دست بیاورد برابر $\max(3, 7-c)$ خواهد بود، چرا که می تواند مسیری که در شکل های صفحه بعد مشخص شده است را طی کند و به امتیاز ۷ برسد و چون هزینه c را برای به کنترل در آوردن حرکت حریف پرداخت کرده، امتیاز کلی آن $7-c$ می شود؛ یا اینکه می تواند بدون استفاده از این قابلیت ویژه، مسیر نشان داده در درخت بالا (درخت به دست آمده در قسمت الف) را طی کند و به امتیاز ۳ برسد. پس اگر $7-c>3$ باشد، یا معادلا $c<4$ باشد، استفاده از این قابلیت ویژه به صرفه است و در غیر این صورت استفاده از آن به صرفه نخواهد بود. در نتیجه، استفاده از این قابلیت ویژه برای $c=2$ به صرفه است ولی برای $c=5$ به صرفه نمی باشد.



شکل بالا درخت بازی را برای $c = 2$ نمایش می دهد. همانطور که پیش تر گفته شد و در شکل هم مشخص شده است، در این حالت بیشترین امتیازی که بازیکن اول می تواند کسب کند برابر ۵ می باشد و استفاده از قابلیت ویژه به صرفه است. شکل زیر نیز درخت بازی را برای $c = 5$ نشان می دهد که در آن استفاده از قابلیت ویژه به صرفه نمی باشد.



سوال ۳:

الف) به شکل زیر دقت کنید:

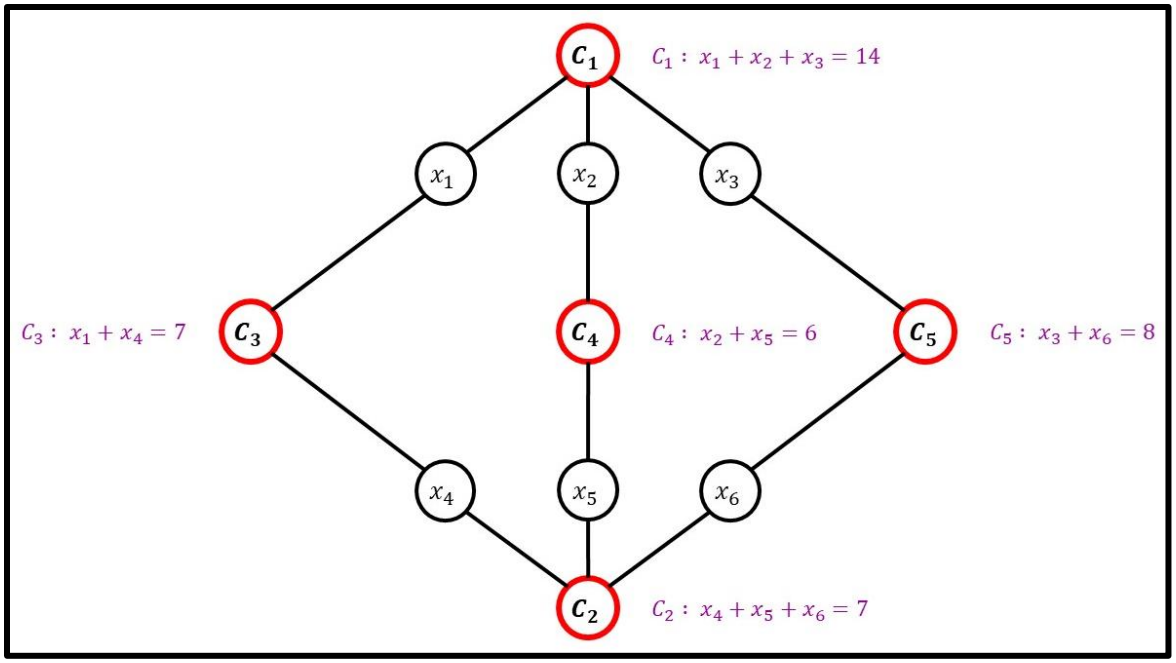
x_1	x_2	x_3	۱۴
x_4	x_5	x_6	۷
۷	۶	۸	

با توجه به شکل بالا، در مسئله CSP معادل، ۶ متغیر (variable) به نام های x_1 ، x_2 ، x_3 ، x_4 ، x_5 و x_6 داریم، به طوری که $x_1 \in \{4,5\}$ ، $x_2 \in \{5,6\}$ ، $x_3 \in \{3,4\}$ ، $x_4 \in \{1,2\}$ ، $x_5 \in \{1,2\}$ و $x_6 \in \{3,4\}$ می باشد (برای هر x_j اگر مقدار اولیه آن در جدول x باشد، داریم: $x_j \in \{[x], [x]+1\}$).

قیود (constraints) مسئله را می توان به شکل تساوی های زیر نمایش داد:

$$C_1 : x_1 + x_2 + x_3 = 14$$
$$C_2 : x_4 + x_5 + x_6 = 7$$
$$C_3 : x_1 + x_4 = 7$$
$$C_4 : x_2 + x_5 = 6$$
$$C_5 : x_3 + x_6 = 8$$

ب) گراف محدودیت های مسئله، به شکل زیر می باشد. در این گراف، محدودیت (قید) ها را با راس های قرمز رنگ و متغیر ها را با راس های سیاه رنگ نمایش داده ایم و هر متغیر به تمامی قیودی که در آنها شرکت دارد، متصل شده است.



ج) مراحل backtrack به شکل زیر می باشد. در هر مرحله، خانه (متغیر) هایی از جدول که تا به آن مرحله پر شده اند، Forward Looking table و همچنین خانه (متغیر) ای که باید در مرحله بعد مقدار دهی شود (با توجه به هیوریستیک های MRV و Degree) مشخص شده اند.

x_1	x_2	x_3	x_4	x_5	x_6
{4,5}	{5,6}	{3,4}	{1,2}	{1,2}	{3,4}

4		

x_1	x_2	x_3	x_4	x_5	x_6
4	{5,6}	{3,4}	–	{1,2}	{3,4}

backtrack

x_1	x_2	x_3	x_4	x_5	x_6
{4,5}	{5,6}	{3,4}	{1,2}	{1,2}	{3,4}

5		

x_1	x_2	x_3	x_4	x_5	x_6
5	{5,6}	{3,4}	2	{1,2}	{3,4}

5		
2		

x_1	x_2	x_3	x_4	x_5	x_6
5	{5,6}	{3,4}	2	{1,2}	{3,4}

5	5	
2		

x_1	x_2	x_3	x_4	x_5	x_6
5	5	4	2	1	{3,4}

5	5	
2	1	

x_1	x_2	x_3	x_4	x_5	x_6
5	5	4	2	1	4

5	5	4
2	1	

x_1	x_2	x_3	x_4	x_5	x_6
5	5	4	2	1	4

5	5	4
2	1	4

x_1	x_2	x_3	x_4	x_5	x_6
5	5	4	2	1	4

جدول نهایی

5	5	4
2	1	4

توضیح: در ابتدا از assignment تهی (جدول خالی) شروع می کنیم. در ابتدا تمامی متغیر ها، تعداد Remaining Value (همگی ۲) و همچنین Degree (همگی ۲) یکسانی دارند، در نتیجه به شکل دلخواه متغیر x_1 را برای مقدار دهی در مرحله اول انتخاب می کنیم و مقدار آن را برابر ۴ قرار می دهیم. در مرحله بعد، با توجه به Forward Checking table متوجه می شویم که بخاطر قید C_3 و مقداری که به متغیر x_1 نسبت دادیم، مقدار مجازی برای متغیر x_4 وجود ندارد و به یک Dead End رسیده ایم، پس Backtrack می کنیم و مقدار متغیر x_1 را برابر ۵ قرار می دهیم. در مرحله بعد با توجه به قید C_3 و مقداری که به متغیر x_1 نسبت دادیم، تنها مقدار مجاز برای متغیر x_4 ۲ می باشد و این متغیر، متغیر MRV است. پس در این مرحله به متغیر x_4 مقدار ۲ را نسبت می دهیم. در ادامه، تمامی متغیر های باقی مانده دارای Remaining Value و Degree یکسان هستند. از همین رو، به شکل دلخواه متغیر x_2 را برای مقدار دهی در مرحله بعد انتخاب می کنیم و مقدار ۵ را درون آن می ریزیم. در مرحله بعد با توجه به قید C_4 و مقداری که به متغیر x_2 نسبت دادیم، تنها مقدار مجاز برای متغیر x_5 ۱ می باشد. همچنین با توجه به قید C_1 و مقادیری که به متغیر های x_1 و x_2 نسبت دادیم، تنها مقدار مجاز برای متغیر x_3 ۴ می باشد. پس هر دوی این متغیر ها MRV هستند و از طرفی هر دو در بین متغیر های باقی مانده، هر دو با ۱ متغیر دیگر (x_6) در قیدی شرکت دارند (یعنی هیوریستیک Degree نیز تساوی شرایط این دو متغیر را به هم نمی زند). در نتیجه به صورت دلخواه متغیر x_5 را برای مقدار دهی انتخاب می کنیم و مقدار ۱ را درون آن می ریزیم. در مرحله بعد، همچنان تنها مقدار مجاز برای متغیر x_3، ۴ می باشد. همچنین با توجه به قید C_2 و مقادیری که به متغیر های x_4 و x_5 نسبت دادیم، تنها مقدار مجاز برای متغیر x_6 نیز ۴ می باشد. پس هر دوی این متغیر ها MRV هستند و از طرفی هر دو در بین متغیر های باقی مانده، هر دو با ۱ متغیر دیگر (با یکدیگر) در قیدی شرکت دارند. در نتیجه به صورت دلخواه متغیر x_3 را برای مقدار دهی انتخاب می کنیم و مقدار ۴ را درون آن قرار می دهیم. در مرحله آخر، فقط متغیر x_6 بدون مقدار باقی مانده و تنها مقدار مجاز برای آن، مقدار ۴ می باشد. پس این مقدار را به متغیر x_6 نسبت می دهیم و جدول نهایی که در آن تمامی قیود ارضا شده اند، به دست می آید.

سوال ۴:

الف) طبق تعریف می دانیم تابع t محدب است اگر داشته باشیم:

$$\forall x, y \in D_t \; \forall \theta \in [0,1] \; ; \; t(\theta x + (1-\theta)y) \leq \theta t(x) + (1-\theta)t(y)$$

به طور معادل برای توابع از $\mathbb{R}$ به $\mathbb{R}$ ، تابع t که دارای مشتق دوم می باشد، محدب است اگر و تنها اگر داشته باشیم:

$$\forall x \in D_t \; ; \; t''(x) \geq 0$$

۱- حتما محدب است.

اثبات: چون f و g محدب اند، طبق تعریف اول داریم:

$$\forall x, y \in \mathbb{R} \; \forall \theta \in [0,1] \; ; \; f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y)$$
$$\forall x, y \in \mathbb{R} \; \forall \theta \in [0,1] \; ; \; g(\theta x + (1-\theta)y) \leq \theta g(x) + (1-\theta)g(y)$$

حال دقت کنید که با توجه به نامساوی های بالا، خواهیم داشت:

$$\begin{aligned}\forall x, y \in \mathbb{R} \; \forall \theta \in [0,1] \; ; \; & f(\theta x + (1-\theta)y) + tg(\theta x + (1-\theta)y) \\ & \leq \theta f(x) + (1-\theta)f(y) + t[\theta g(x) + (1-\theta)g(y)] \\ & = [\theta f(x) + t\theta g(x)] + [(1-\theta)f(y) + t(1-\theta)g(y)] \\ & = \theta[f(x) + tg(x)] + (1-\theta)[f(y) + tg(y)]\end{aligned}$$

حال دقت کنید که داریم:

$$h(\theta x + (1-\theta)y) = f(\theta x + (1-\theta)y) + tg(\theta x + (1-\theta)y)$$
$$\theta h(x) + (1-\theta)h(y) = \theta[f(x) + tg(x)] + (1-\theta)[f(y) + tg(y)]$$

از تساوی های بالا و نامساوی که پیش تر ذکر گردید، نتیجه می شود:

$$\forall x, y \in \mathbb{R} \; \forall \theta \in [0,1] \; ; \; h(\theta x + (1-\theta)y) \leq \theta h(x) + (1-\theta)h(y)$$

و در نتیجه طبق تعریف اول، تابع $h(x)$ حتما محدب می باشد.

۲- حتما محدب است.

اثبات: چون f و g محدب اند، طبق تعریف اول می توان نوشت:

$$\forall x,y \in \mathbb{R}\ \forall \theta \in [0,1]\ ;\ f(\theta x + (1-\theta)y) \le \theta f(x) + (1-\theta)f(y)$$
$$\forall x,y \in \mathbb{R}\ \forall \theta \in [0,1]\ ;\ g(\theta x + (1-\theta)y) \le \theta g(x) + (1-\theta)g(y)$$

حال دقت کنید که با توجه به نامساوی های بالا، خواهیم داشت:

$$\begin{aligned}\forall x,y \in \mathbb{R}\ \forall \theta \in [0,1]\ ;& \max\{f(\theta x + (1-\theta)y)\ , g(\theta x + (1-\theta)y)\}\\ &\le\ \max\{\theta f(x) + (1-\theta)f(y)\ , \theta g(x) + (1-\theta)g(y)\}\\ &\le \max\{\theta f(x), \theta g(x)\} + \max\{(1-\theta)f(y), (1-\theta)g(y)\}\\ &= \theta\ \max\{f(x), g(x)\} + (1-\theta)\max\{f(y), g(y)\}\end{aligned}$$

علت برقراری نامساوی اول این است که برای هر عضو از مجموعه سبز رنگ، حداقل یک عضو از مجموعه آبی رنگ وجود دارد که بزرگ تر یا مساوی آن باشد و در نتیجه بیشینه مجموعه سبز رنگ حتما کوچک تر یا مساوی بیشینه مجموعه آبی رنگ خواهد بود. حال دقت کنید که داریم:

$$k(\theta x + (1-\theta)y) = \max\{f(\theta x + (1-\theta)y)\ , g(\theta x + (1-\theta)y)\}$$
$$\theta k(x) + (1-\theta)k(y) = \theta\ \max\{f(x), g(x)\} + (1-\theta)\max\{f(y), g(y)\}$$

از تساوی های بالا و نامساوی که پیش تر ذکر گردید، نتیجه می شود:

$$\forall x,y \in \mathbb{R}\ \forall \theta \in [0,1]\ ;\ k(\theta x + (1-\theta)y) \le \theta k(x) + (1-\theta)k(y)$$

و در نتیجه طبق تعریف اول، تابع $k(x)$ حتما محدب می باشد.

۳- ممکن است محدب نباشد.

مثال نقض: قرار دهید $f(x) = (x+2)^2$ و $g(x) = (x-2)^2$. داریم:

$$f''(x) = g''(x) = 2 \geq 0$$

در نتیجه هر دوی f و g طبق تعریف دوم محدب می باشند. حال نشان می دهیم $r(x)$ محدب نمی باشد.

دقت کنید که اگر قرار دهیم $x = 2$ ، $y = -2$ و $\theta = \frac{1}{2}$، خواهیم داشت:

$$r(\theta x + (1-\theta)y) = r\left(\frac{2-2}{2}\right) = r(0) = \min\{f(0)\,,g(0)\} = \min\{4\,,4\} = 4$$
$$\theta r(x) + (1-\theta)r(y) = \frac{r(2)+r(-2)}{2} = \frac{\min\{f(2)\,,g(2)\} + \min\{f(-2)\,,g(-2)\}}{2}$$
$$\Longrightarrow\ \theta r(x) + (1-\theta)r(y) = \frac{\min\{16\,,\ 0\} + \min\{0\,,16\}}{2} = \frac{0+0}{2} = 0$$

پس در نهایت خواهیم داشت:

$$4 = r(\theta x + (1-\theta)y) \nleq \theta r(x) + (1-\theta)r(y) = 0$$

که این با تعریف اول تحدب برای تابع $r(x)$ در تناقض است. پس تابع $r(x)$ محدب نمی باشد.

۴- ممکن است محدب نباشد.

مثال نقض: قرار دهید $f(x) = x^2$ و $g(x) = -1$. داریم:

$$f''(x) = 2 \geq 0\,, g''(x) = 0 \geq 0$$

در نتیجه هر دوی f و g طبق تعریف دوم محدب می باشند. حال نشان می دهیم $s(x)$ محدب نمی باشد.

دقت کنید که داریم:

$$s(x) = f(x)g(x) = -x^2 \ \Longrightarrow s''(x) = -2 \ngeq 0$$

که این با تعریف دوم تحدب برای تابع $s(x)$ در تناقض است.

همچنین اگر قرار دهیم $x = 2$ ، $y = -2$ و $\theta = \frac{1}{2}$، خواهیم داشت:

$$s(\theta x + (1-\theta)y) = r\left(\frac{2-2}{2}\right) = s(0) = 0$$
$$\theta s(x) + (1-\theta)s(y) = \frac{r(2)+r(-2)}{2} = \frac{-2^2 - (-2)^2}{2} = \frac{-4-4}{2} = -4$$
$$\Longrightarrow 0 = s(\theta x + (1-\theta)y) \nleq \theta s(x) + (1-\theta)s(y) = -4$$

که این نیز با تعریف اول تحدب برای تابع $s(x)$ در تناقض است. پس تابع $s(x)$ محدب نمی باشد.

ب) طبق تعریف می دانیم مجموعه C محدب است اگر داشته باشیم:

$$\forall x, y \in C \; \forall \theta \in [0,1] \; ; \theta x + (1-\theta) y \in C$$

نشان می دهیم مجموعه داده شده یک مجموعه محدب می باشد. فرض کنید $A_1 = (x_1, t_1)$ و همچنین $A_2 = (x_2, t_2)$ دو عضو دلخواه متعلق به مجموعه C هستند. طبق تعریف این مجموعه، داریم:

$$\|x_1\| \leq t_1$$
$$\|x_2\| \leq t_2$$

حال عدد حقیقی دلخواه $\theta \in [0,1]$ را در نظر بگیرید. طبق نامساوی مثلث برای تابع نرم، داریم:

$$\|\theta x_1 + (1-\theta) x_2\| \leq \|\theta x_1\| + \|(1-\theta) x_2\| = \theta \|x_1\| + (1-\theta)\|x_2\|$$

با ترکیب نامساوی بالا و دو نامساوی صفحه قبل، خواهیم داشت:

$$\|\theta x_1 + (1-\theta) x_2\| \leq \theta \|x_1\| + (1-\theta)\|x_2\| \leq \theta t_1 + (1-\theta) t_2$$

حال دقت کنید که داریم:

$$\theta A_1 + (1-\theta) A_2 = \theta (x_1, t_1) + (1-\theta)(x_2, t_2) = (\theta x_1 + (1-\theta) x_2, \theta t_1 + (1-\theta) t_2) = A_3$$

به وضوح با توجه به آنچه که پیش تر نشان دادیم، A_3 متعلق به مجموعه C می باشد، چرا که نرم برداری که در مولفه اول آن قرار گرفته، کوچک تر یا مساوی عدد حقیقی می باشد که در مولفه دوم آن قرار گرفته است.

پس داریم:

$$A_3 = \theta A_1 + (1-\theta) A_2 \in C$$

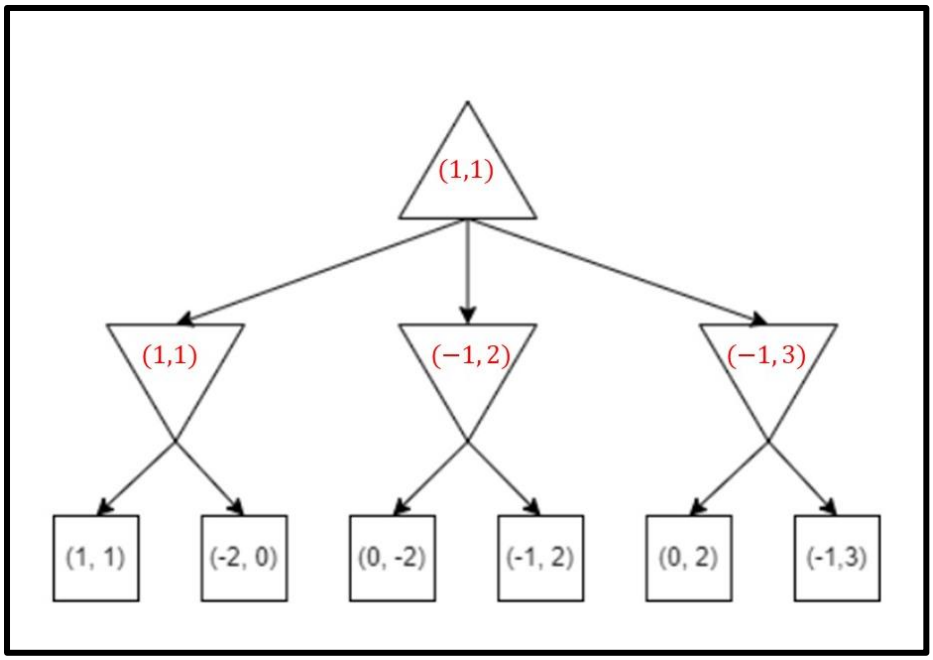
حال دقت کنید که چون A_1 ، A_2 و θ به شکل تصادفی انتخاب شده بودند، می توان نوشت:

$$\forall A_1, A_2 \in C \; \forall \theta \in [0,1] \; ; \theta A_1 + (1-\theta) A_2 \in C$$

که این معادل با محدب بودن مجموعه C می باشد.

سوال ۵:

الف) درخت کامل شده، به صورت زیر می باشد:

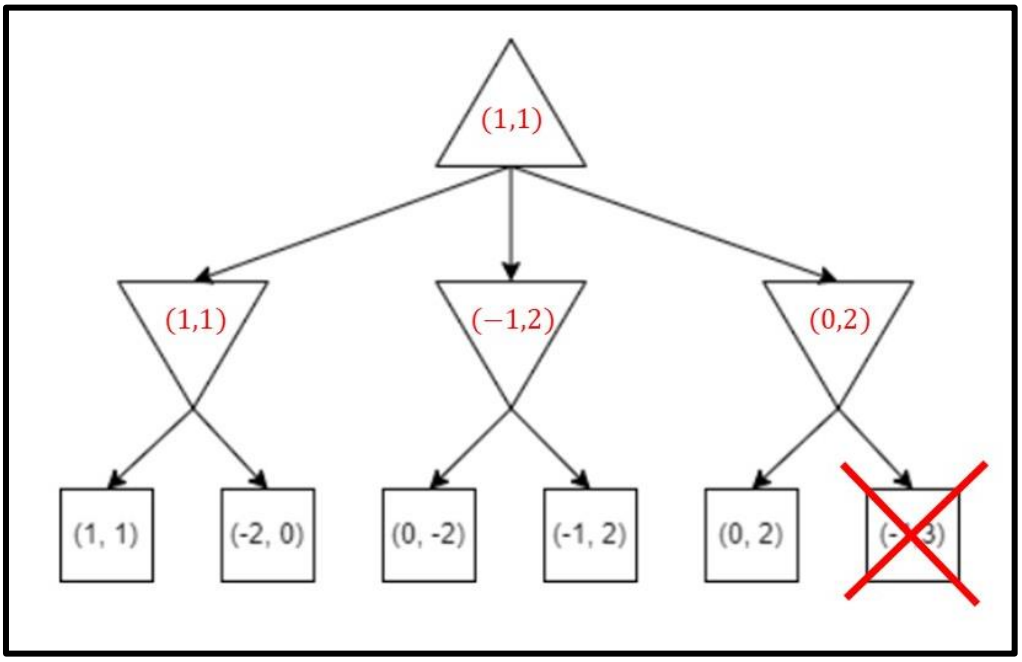


دقت کنید که در درخت مربوط به این بازی، هر بازیکن در هر مرحله به دنبال بیشینه کردن امتیاز خود می باشد. بنابر این بازیکن اول در هر مرحله زوج مرتبی را از بین فرزندانش انتخاب می کند که بزرگ ترین مولفه اول را داشته باشد و بازیکن دوم نیز در هر مرحله زوج مرتبی را از بین فرزندانش انتخاب می کند که دارای بزرگ ترین مولفه دوم باشد.

ب) در بازی های non-zero-sum، مقادیری که بازیکنان اول و دوم در تلاش برای بیشینه کردنشان هستند، کاملا مستقل از یکدیگرند. در نتیجه دیگر موقعیت هایی وجود ندارند که مطمئن هستیم یک بازیکن به هیچ وجه اجازه نمی دهد بازیکن دیگر از یک شاخه (branch) از game tree پایین برود. به طور مثال در حالتی که $U_A = U_B$ باشد، سود هر بازیکن معادل سود بازیکن دیگر خواهد شد و این مسئله (پیدا کردن مسیر بهینه در game tree) به مسئله پیدا کردن برگی که بیشترین مقدار را دارد تبدیل می شود که این برگ می تواند در هر جایی از درخت قرار گرفته باشد.

ج) خیر، مقدار محاسبه شده برای ریشه در یک بازی non-zero-sum لزوما worst case نیست و امتیازی که بازیکن A در نهایت کسب می کند، می تواند کمتر از U_A محاسبه شده باشد. یکی از حالاتی که می تواند منجر به این اتفاق شود، این است که بازیکن دوم (B) به جای اینکه به دنبال بیشینه کردن امتیاز خود باشد بخواهد امتیاز بازیکن اول را کمینه کند. در این حالت، ممکن است امتیاز بازیکن اول در نهایت از مقدار محاسبه شده در ریشه کمتر شود. به طور مثال در بازی مطرح شده در همین سوال، اگر بازیکن B بجای $(1,1)$ در شاخه سمت چپ، $(-2,0)$ را انتخاب کند، امتیاز نهایی بازیکن A از ۱ به ۱- کاهش می یابد.

د) تنها یک راس در طی این فرایند خط می خورد که در شکل زیر مشخص شده است:



در الگوریتم تعمیم یافته هرس آلفابتا، طبق آنچه که در قسمت بعد توضیح خواهیم داد، برای هر راس مربوط به بازیکن B مانند S اگر در حین پیمایش میان فرزندانش شرط زیر برقرار شود، پیمایش سایر فرزندان را متوقف می کنیم و آن فرزندان هرس می شوند:

$$U_B(S) > \epsilon - \alpha$$

در این درخت بازی، در ابتدا مقدار α برابر $-\infty$ می باشد. با پیمایش فرزند چپ ریشه، هنگامی که وارد برگ $(1,1)$ می شویم، مقدار α برابر ۱ می شود. سپس مقدار $(1,1)$ وارد فرزند چپ ریشه می شود (این راس را S_1 می نامیم)، اما چون $U_B(S_1) = 1 \not> 1 = \epsilon - \alpha$ برگ $(-2,0)$ هرس نمی شود و چون مقدار مولفه اول این برگ از α کوچک تر است، مقدار α نیز تغییر نمی کند. در ادامه پیمایش، وارد فرزند وسط ریشه می شویم، سپس وارد برگ $(0,-2)$ می شویم و مقدار $(0,-2)$ وارد فرزند وسط ریشه می شود (این راس را S_2 می نامیم). دقت کنید که چون مولفه اول $(0,-2)$ از مقدار فعلی α کمتر است، مقدار α تغییر نخواهد کرد. چون $U_B(S_2) = -2 \not> 1 = \epsilon - \alpha$ برگ $(-1,2)$ هرس نمی شود و مقدار α نیز دچار تغییر نمی شود. در نهایت، فرزند چپ ریشه را پیمایش می کنیم. در ابتدا وارد برگ $(0,2)$ می شویم و مقدار $(0,2)$ وارد فرزند وسط ریشه می شود (این راس را S_3 می نامیم). در اینجا نیز چون مولفه اول $(0,-2)$ از مقدار فعلی α کمتر است، مقدار α بدون تغییر باقی می ماند. چون $U_B(S_3) = 2 > 1 = \epsilon - \alpha$ و شرط هرس برقرار می باشد، برگ $(-1,3)$ هرس می شود و وارد آن نمی شویم. پس تنها راسی که هرس می شود، برگ $(-1,3)$ می باشد.

۵) برای هر راس مربوط به بازیکن B مانند S اگر در حین پیمایش میان فرزندانش شرط زیر برقرار شود، پیمایش سایر فرزندان را متوقف می کنیم و آن فرزندان هرس خواهند شد:

$$U_B(S) > \epsilon - \alpha$$

دقت کنید که در اینجا منظور از $(U_A(S), U_B(S))$، مقداری است که تاکنون در پیمایش درخت بازی برای راس S پیدا شده است و اگر شرط بالا برقرار شود، همین مقدار درون راس S نوشته خواهد شد.

توضیح: در ابتدا دقت کنید که از شرط nearly-zero-sum بودن بازی، برای هر راس دلخواه مانند N داریم:

$$|U_B(N) + U_A(N)| \leq \epsilon \implies U_B(N) + U_A(N) \leq \epsilon \implies U_B(N) \leq \epsilon - U_A(N)$$

حال توجه کنید که برای هر فرزند دلخواه از راس S مانند S' که پیمایش خواهد شد، در صورتی مقدار آن فرزند درون راس S ریخته می شود که داشته باشیم $U_B(S) < U_B(S')$ (اگر $U_B(S) > U_B(S')$ باشد، قطعا مقدارش در راس S ریخته نمی شود و هرس خواهد شد). پس اگر شرطی که در ابتدا بیان شد برای راس S برقرار باشد، از نامساوی بالا نتیجه می شود:

$$\epsilon - \alpha < U_B(S) < U_B(S') \leq \epsilon - U_A(S') \implies U_A(S') < \alpha$$

اما با توجه به تعریف α، می دانیم که جوابی برای ریشه با $U_A = \alpha$ وجود دارد و در نتیجه می توانیم از پیمایش هر راس که U_A برای آن راس از α کوچک تر می باشد، صرف نظر کنیم (چرا که به علت وجود جوابی با U_A بزرگ تر، در نهایت قطعا مقدار آن راس برای ریشه انتخاب نخواهد شد). پس می توان از پیمایش S' صرف نظر کرد و S' هرس خواهد شد.

و) حداقل امتیازی که ممکن است توسط بازیکن اول کسب شود (با فرض بهینه بازی کردن این بازیکن)، برابر است با:

$U_A - 2\epsilon$

اثبات: فرض کنید در درخت بازی به دست آمده با فرض اینکه هر دو بازیکن به صورت بهینه عمل می کنند و به دنبال بیشینه کردن امتیاز خود هستند، مقدار نوشته شده در داخل ریشه برابر (U_A, U_B) باشد. همچنین فرض کنید در درخت بازی به دست آمده در حالتی که فقط بازیکن اول به شکل بهینه عمل کرده و بازیکن دوم لزوما به صورت بهینه بازی نمی کند، مقدار نوشته شده در داخل ریشه برابر (U'_A, U'_B) باشد. چون در هر دو حالت بازیکن اول به شکل بهینه عمل کرده ولی بازیکن دوم در حالت اول به صورت بهینه بازی کرده و به دنبال بیشینه کردن امتیاز خود بوده و در حالت دوم به صورت غیربهینه بازی کرده است، به وضوح داریم:

$U'_B \leq U_B$

حال دقت کنید که از شرط nearly-zero-sum بودن بازی، به دست می آید:

$$|U_A + U_B| \leq \epsilon \Longrightarrow U_A + U_B \leq \epsilon \Longrightarrow U_B \leq \epsilon - U_A$$
$$|U'_A + U'_B| \leq \epsilon \Longrightarrow -\epsilon \leq U'_A + U'_B \Longrightarrow -\epsilon - U'_A \leq U'_B$$

از نامساوی های بالا نتیجه می شود:

$$-\epsilon - U'_A \leq U'_B \leq U_B \leq \epsilon - U_A \Longrightarrow -\epsilon - U'_A \leq \epsilon - U_A \Longrightarrow U_A - 2\epsilon \leq U'_A$$

که این همان حکم می باشد و درنتیجه حکم درست است.

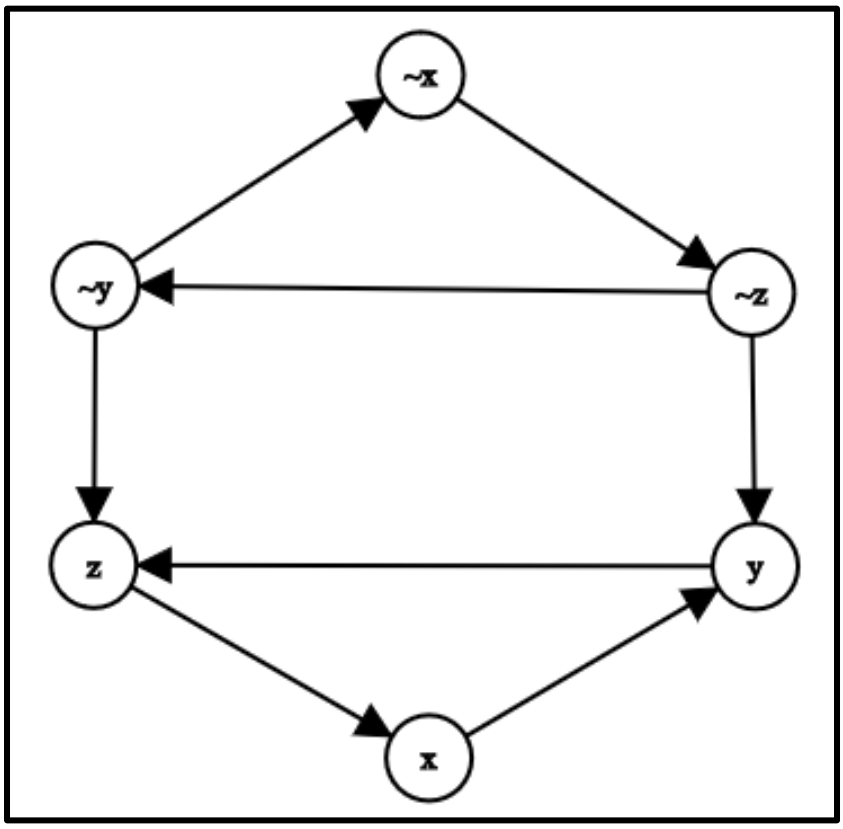
سوال ۶:

الف) برای بیان مسئله به شکل CSP، می توان گفت که n متغیر به نام های $x_1, x_2, \dots, x_n$ داریم به طوری که مقدار هر کدام از مجموعه $D = \{0,1\}$ (دامنه متغیر) انتخاب خواهد شد. همچنین m قید (constraint) به شکل زیر داریم:

$C_1 : x_{i_1} \vee x_{j_1} = 1$
$C_1 : x_{i_2} \vee x_{j_2} = 1$
...
$C_m : x_{i_m} \vee x_{j_m} = 1$

به طوری که $i_1, i_2, \dots, i_m, j_1, j_2, \dots, j_m \in \{1,2, \dots, n\}$ می باشند.

گراف 2-SAT برای مسئله نمونه داده شده به شکل زیر می باشد. در این مسئله سه متغیر به نام های x, y, z و چهار قید به شکل $y \vee z = 1$ ، $x \vee \sim z = 1$ ، $\sim y \vee z = 1$ و $\sim x \vee y = 1$ داریم.



یک پاسخ برای این مسئله، مقداردهی (assignment) $x = y = z = 1\ (True)$ می باشد. به وضوح با این مقداردهی به متغیر ها، تمامی مقادیر $y \vee z$ ، $x \vee \sim z$ ، $\sim y \vee z$ و $\sim x \vee y$ برابر یک می شوند و در نتیجه قید داده شده، ارضا می شود.

ب و ج) باید هر دو طرف قضیه داده شده در قسمت ب را اثبات نماییم.

طرف اول:

- اگر حداقل یک مولفه قویا همبند وجود داشته باشد که شامل یک متغیر و نقیض آن باشد، مسئله 2-SAT پاسخی نخواهد داشت.

اثبات: دقت کنید که اگر مسیر جهت دار $x_1 \longrightarrow x_2 \longrightarrow \cdots \longrightarrow x_n$ از متغیر x_1 به متغیر x_n در گراف متناظر وجود داشته باشد، با توجه به نحوه تعریف یال ها، باید پس از مقدار دهی به متغیر ها داشته باشیم:

$$(x_1 \Longrightarrow x_2) \wedge (x_2 \Longrightarrow x_3) \wedge \ldots \wedge (x_{n-1} \Longrightarrow x_n) \equiv True$$

از عبارت بالا، می توان عبارت زیر را نتیجه گرفت:

$$x_1 \Longrightarrow x_n \equiv True$$

حال دقت کنید که اگر متغیر x و نقیض آن، یعنی $\sim x$ در یک مولفه قویا همبند قرار گرفته باشند، هم مسیری جهت دار از x به $\sim x$ و هم مسیری جهت دار از $\sim x$ به x وجود دارد. پس طبق آنچه که بالا تر نشان دادیم، باید داشته باشیم:

$$(x \Longrightarrow \sim x) \wedge (\sim x \Longrightarrow x) \equiv True \Longrightarrow (\sim x \vee \sim x) \wedge (x \vee x) \equiv \sim x \wedge x \equiv True$$
$$\Longrightarrow False \equiv True$$

تناقض حاصل نشان می دهد که مقداردهی مطلوبی برای متغیر ها (که قید های مسئله را ارضا نماید) در این حالت وجود ندارد و حکم درست می باشد.

طرف دوم:

- اگر هیچ مولفه قویا همبندی به صورت همزمان شامل یک متغیر و نقیض آن نباشد، مسئله 2-SAT دارای حداقل یک پاسخ خواهد بود.

اثبات: برای اثبات این گزاره الگوریتمی معرفی می کنیم که اگر فرض برقرار باشد، یک مقداردهی معتبر (که قید های مسئله را ارضا می نماید) برای متغیر ها پیدا کند.

دقت کنید که حتی اگر فرض برقرار باشد، ممکن است یک مسیر جهت دار از x به $\sim x$ و یا بالعکس (البته نه به صورت همزمان) وجود داشته باشد. در این صورت انتخاب یکی از مقادیر $True$ یا $False$ برای متغیر x به تناقض منجر می شود، در حالی که انتخاب مقدار دیگر برای این متغیر به تناقض منجر نخواهد شد. به عبارت دیگر، اگر مسیری جهت دار از $\sim x$ به x وجود داشته باشد، باید داشته باشیم:

$$x \Longrightarrow \sim x \equiv x \equiv True$$

و اگر مسیری جهت دار از x به $\sim x$ وجود داشته باشد، باید داشته باشیم:

$$x \Longrightarrow \sim x \equiv \sim x \equiv T \Longrightarrow x \equiv False$$

حال با در نظر گرفتن این نکته، در صفحه بعد به بیان الگوریتم می پردازیم.

در ابتدا دقت کنید که اگر در گراف مسئله، هر مولفه قویا همبند را یک ابر گره در نظر بگیریم و با استفاده از این ابرگره ها یک گراف جدید بسازیم، گراف حاصل یک dag خواهد شد و در نتیجه می توان با استفاده از مرتب سازی توپولوژیک مولفه های قویا همبند گراف متناظر به مسئله را به ترتیب توپولوژیک مرتب نمود. اگر $comp[v]$ برابر شماره مولفه قویا همبندی باشد که راس v در داخل آن قرار گرفته، پس از مرتب سازی توپولوژیک، اگر $comp[v] < comp[u]$ باشد می توان گفت که در ترتیب به دست آمده، راس u در سمت راست راس v قرار گرفته است و می دانیم که مسیری جهت دار از رئوس سمت راست یک راس به آن وجود ندارد؛ بنابراین هیچ مسیر جهت داری از راس u به راس v وجود نخواهد داشت. پس از مرتب سازی توپولوژیک مولفه ها، اگر $comp[x] < comp[\sim x]$ بود، مقدار متغیر x را برابر $False$ و در غیر این صورت برابر $True$ قرار می دهیم.

حال نشان می دهیم که با این مقداردهی به متغیر ها، به تناقض نخواهیم رسید. فرض کنید مقدار $True$ برای متغیر x انتخاب شده است. حالت $x \equiv False$ نیز به طریقی مشابه اثبات می شود.

در ابتدا نشان می دهیم که مسیری جهت دار از x به $\sim x$ وجود ندارد. چون مقدار $True$ به متغیر x منتسب شده است، داریم $comp[\sim x] < comp[x]$ و در نتیجه طبق آنچه که پیش تر بیان کردیم، مسیری جهت دار از x به $\sim x$ وجود نخواهد داشت.

در ادامه، با استفاده از برهان خلف؛ نشان می دهیم که متغیری مانند y وجود ندارد به طوری که هم مسیری جهت دار از x به y و هم مسیری جهت دار از x به $\sim y$ وجود داشته باشد. وجود چنین متغیری به تناقض منجر می شود، چرا که در صورت وجود آن، $True$ بودن x نتیجه می دهد $y \equiv True \wedge \sim y \equiv True$ که تناقض است. فرض کنید چنین متغیری وجود دارد. حال با توجه به خواص گراف، چون مسیری از x به y وجود دارد، مسیری جهت دار از $\sim y$ به $\sim x$ نیز وجود خواهد داشت (اگر $x = p_0 \longrightarrow p_1 \longrightarrow \cdots \longrightarrow p_n = y$ مسیر جهت دار از x به y باشد، برای هر یال $p_i \longrightarrow p_{i+1}$ در این مسیر، یال $\sim p_{i+1} \longrightarrow \sim p_i$ نیز در این گراف وجود دارد (با توجه به نحوه ساخت گراف). پس مسیر $\sim y = \sim p_n \longrightarrow \cdots \sim p_1 \longrightarrow \sim p_0 = \sim x$ که مسیری جهت دار از $\sim y$ به $\sim x$ می باشد نیز در این گراف وجود خواهد داشت). در نتیجه چون مسیری جهت دار از x به $\sim y$ و همچنین مسیری جهت دار از $\sim y$ به $\sim x$ وجود دارد، مسیری جهت دار از x به $\sim x$ وجود خواهد داشت که با آنچه پیش تر ثابت کردیم در تناقض است. بنابراین فرض خلف باطل است و حکم درست می باشد.

همچنین باید نشان دهیم هیچ مسیر جهت داری از x به متغیری مانند y وجود ندارد، به طوری که داشته باشیم $y \equiv False$ (به وضوح وجود چنین مسیری موجب عدم اعتبار مقداردهی ارائه شده خواهد شد). مجددا به برهان خلف فرض کنید چنین مسیری وجود دارد. دقت کنید چون x مقدار $True$ و y مقدار $False$ را اتخاذ کرده است، می توان نوشت:

$$comp[\sim x] < comp[x]$$
$$comp[y] < comp[\sim y]$$

همچنین چون مسیری جهت دار از x به y وجود دارد، داریم:

$$comp[x] \leq comp[y]$$

که نتیجه می دهد:

$$comp[\sim x] < comp[x] \leq comp[y] < comp[\sim y] \Longrightarrow comp[\sim x] < comp[\sim y]$$

از طرفی دقت کنید که چون مسیری جهت دار از از x به y وجود دارد، طبق استدلالی که در قسمت قبل بیان نمودیم، مسیری جهت دار از $\sim y$ به $\sim x$ نیز وجود خواهد داشت که با عبارت بالا در تناقض است. بنابراین فرض خلف باطل بوده و حکم درست می باشد.

پس الگوریتمی ارائه کردیم که اگر فرض برقرار باشد، مقدار دهی معتبری برای متغیر ها پیدا می کند و درستی آن را نیز نشان دادیم. در نتیجه درستی حکم نشان داده شد.

حال دقت کنید که می توانیم مولفه های قویا همبند گراف مسئله را با استفاده از الگوریتم کساراجو در زمان $O(n+m)$ بیابیم. در پیمایش دوم گراف مسئله با استفاده از الگوریتم کساراجو، مولفه های قویا همبند را به ترتیب توپولوژیک می بینیم و در نتیجه می توانیم برای هر متغیر دلخواه مانند x، مقدار $comp[x]$ را محاسبه نماییم. در نهایت مقدار تمامی متغیر ها را با مقایسه $comp[x]$ و $comp[\sim x]$ برای هر متغیر مانند x در زمان $O(n)$ به دست می آوریم. پس پیچیدگی زمانی این الگوریتم، $O(n+m)$ می باشد.

یک پیاده سازی برای این الگوریتم را می توانید در این لینک مشاهده نمایید.