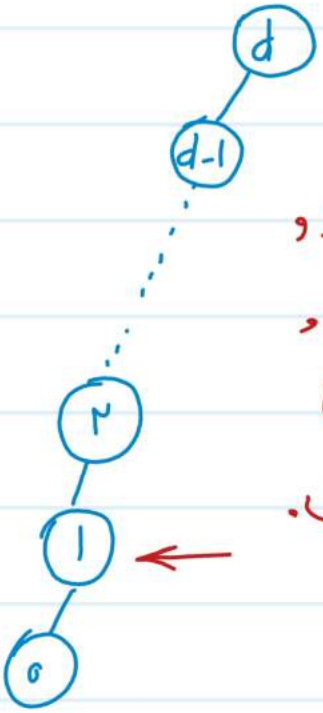
۱. (۱۵ نمره، درجه سختی ۷) درستی یا نادرستیِ عبارت‌های زیر را با ذکر دلیل مشخص کنید:

الف) حداکثر تعداد دفعاتی که الگوریتم backtracking ممکن است مجبور به backtrack شود، اگر از arc consistency و هیوریستیک‌های MRV و LCV استفاده کند $O(dn^2)$ است.(n تعداد متغیرها و d تعداد مقادیر مجاز برای هر متغیر است.)

خیر، در حالت worst-case، تعداد backtrack‌ها بیشتر از $O(dn^2)$ است، برای مثال گراف کامل d+1 راسی را در نظر بگیرید، در این صورت هیچ رأسی بر دیگری در هیچ مرحله‌ای اولویت نخواهد داشت و همچنین چون d رنگ داریم نمی‌توان همه‌ی رئوس را رنگ کرد، موقع backtrack زمانی که تا عمق d-2 پیش برویم، سه راس باقی می‌ماند که هرکدام دو رنگ مجاز دارند و به این صورت هیچ کدام در arc-consistency حذف نخواهند شد، در سایر عمق‌ها هم همین طور، به این صورت داریم:



تعداد داخل راس‌ها، تعداد رنگ مجاز باقی مانده برای هر راس‌اند، در نتیجه در عمق فلش زده شده، backtrack اتفاق می‌افتد، سپس در نود پدرش رنگ دیگر انتخاب می‌شود و دوباره به نود فرزند بازمی‌گردیم و دوباره نود فرزند رنگ می‌شود و به نودی می‌رسیم که نمی‌تواند رنگی داشته باشد، در کل $O(d!)$ تا عملیات داریم که این از در بیشتر از $O(n^2 d)$ است.

d^3

ب) اگر گراف محدودیت یک مسئله CSP با محدودیت های دودویی به صورت درخت با n رأس باشد، پیچیدگی محاسباتی حل کننده کارا برحسب n ، $O(n^2)$ است.

ابتدا به یک topological_sort نیاز داریم که $O(V+E)$ است
$O(n+n-1) = O(n)$ ، به $O(n)$ گام هم نیاز داریم که در هرگام $O(d^2)$ ← عملیات لازم است ← درکل $O(nd^2)$ زمان لازم است، همچنین اگر قرار دهیم $d = \frac{n}{2}$ ، آنگاه اردمان برابر است با $O(n^3)$ که بیشتر از $O(n^2)$ است، پس این گزینه غلط است.

ج) الگوریتم هرس آلفابتا علاوه بر آنکه زمان را کاهش میدهد در جواب به دست آمده برای ریشه درخت با minimax نیز تأثیرگذار می‌باشد.

الگوریتم هرس آلفا بتا، زمان را کاهش می دهد ولی جواب بست آمده برای ریشه درخت را تغییر نمی دهند بلکه فقط حالت هایی که آنقدر بد هستند که نیازی به بررسی آنها نیست را حذف میکند

برای دو مورد بعدی تابع اکیدا صعودی F و یک بازی zero-sum با دو بازیکن را درنظر بگیرید:

د) اعمال تابع F روی برگ‌های یک درخت minimax برای این بازی پاسخ بهینه آن را تغییر نخواهد داد.

ه) اعمال تابع F روی برگ‌های یک درخت minimax برای این بازی برگ‌هایی که توسط alpha-beta pruning هرس می شوند را تغییر نمی دهد.

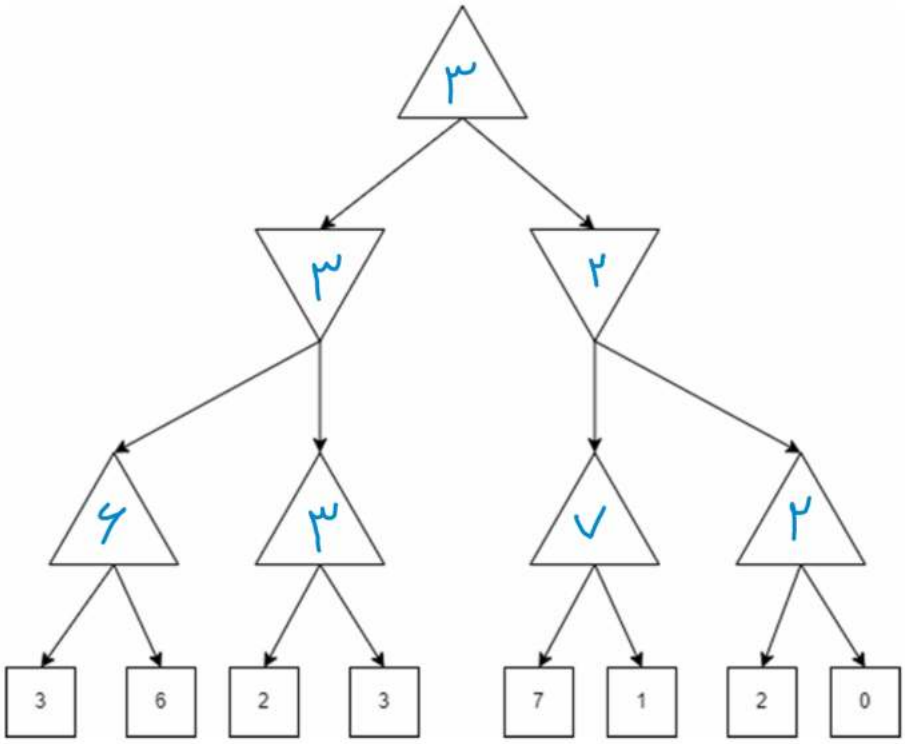
د) چون F اکیداً صعودی است، پس اگر در مرحله‌ای بین a_i و a_j ، a_i انتخاب میشد، الان $F(a_i)$ انتخاب می‌شود چون یا داشته ایم ماکزیمم انتخاب می‌کرده ایم و $a_i > a_j \leftarrow F(a_i) > F(a_j)$ یا اینکه داشته ایم مینیمم انتخاب می‌کرده ایم و $a_i < a_j \leftarrow F(a_i) < F(a_j)$ پس مسیر بهینه تا هر برگی بوده همان باقی خواهد ماند، حالا فرض کنیم a_k برگی باشد که مسیر تا آن در درخت minimax انتخاب می‌شود، حالا هم همان مسیر انتخاب خواهد شد و لی مقدار پاسخ بهینه برابر $F(a_k)$ خواهد شد. جواب بهینه همان قبلی باقی خواهد ماند.

ه) درست است، راس هایی که هرس می‌شدند، به این صورت است که:

در حال انتخاب ماکزیمم بین نودهای مینیمم هستیم و نودی در یکی از مینیمم ها مشاهده می‌شود که از مقدار فعلی رأس ماکزیمم کمتر است، در این صورت F نود حذفی هم کمتر از F نودی است که در ماکزیمم قرار دارد پس Pruning اتفاق می‌افتد، همچنین ممکن است در حال انتخاب مینیمم بین رأس‌های ماکزیمم باشیم در این صورت، برای Pruning ، در یکی از رأس های ماکزیمم مقداری مشاهده شده که بیشتر از مقدار فعلی رأس مینیمم است، در این صورت F نود حذفی هم بیشتر از F نودی است که در رأس مینیمم قرار دارد، پس Pruning اتفاق می‌افتد. (عبارات بالا دو حالت هستند)

← رأس هایی که بعد از اعمال F ، Prune می‌شوند، همان هایی هستند که قبل از F ، Prune می‌شدند.

۲. (۱۵ نمره، درجه سختی ۵) درخت بازی زیر را در نظر بگیرید و به سوالات مربوطه پاسخ دهید:



شکل ۱

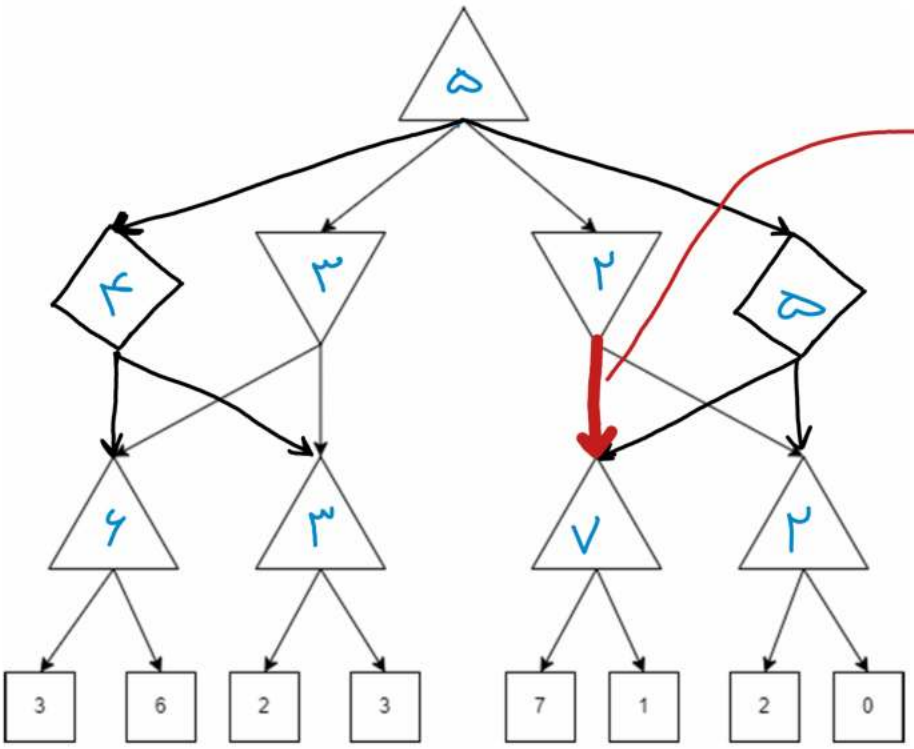
الف) درخت minimax بازی مورد نظر را کامل نمایید.

فرض کنید بازیکن اول یک قابلیت ویژه دریافت کند. این قابلیت ویژه این است که بازیکن اول می تواند با پرداخت کردن هزینه c حرکت انتخابی توسط بازیکن مقابل را تحت کنترل خود دربیاورد.

ب) با در نظر گرفتن فرض $c = ۲$ آیا برای بازیکن اول به صرفه خواهد بود که از این قابلیت ویژه استفاده کند؟ درخت بازی را مجدد رسم کرده و کامل نمایید. اگر پاسخ سوال قبل مثبت بود، نقطه ای در درخت که برای بازیکن اول بهینه است از قابلیت ویژه خود استفاده کند را نیز مشخص نمایید.

ب) بله به صرفه است از این عملیات استفاده کند.

نود هایی که نفر اول، نفر دوم را فورس کند که کاری را که می‌خواهد انجام دهد را با ◇ نشان می دهیم، قاعدتاً هم نفر دوم را فورس می‌کند که کاری را کند که برای نفر اول بهترین است:

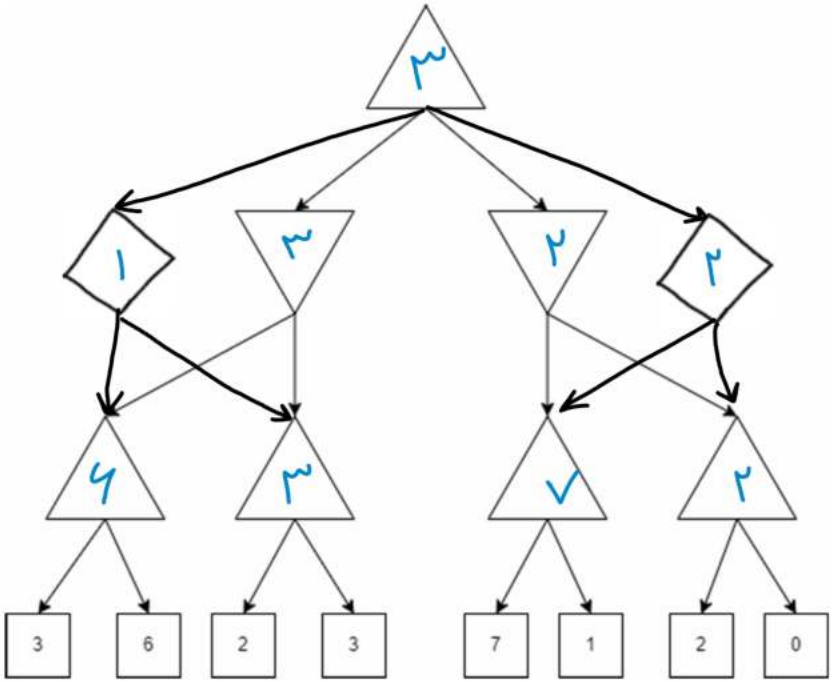


نفر اول، نفر دوم را مجبور به انجام این حرکت کند.

نفر اول حرکت سمت راستی را انجام دهد و سپس با قابلیت ویژه اش نفر دوم را مجبور کند تا حرکتی را انجام دهد که نفر اول می‌تواند ۱ یا ۷ امتیاز بگیرد در این صورت نفر اول هم حرکت ۷ امتیازی را انجام خواهد داد و بنابراین استفاده از قابلیتش ۲ امتیاز از او کم می‌شود ← نفر اول ۵ امتیاز می‌گیرد.

ج) بخش قبل را مجددا اما این بار با فرض $c = ۵$ روی هزینه قابلیت ویژه برای بازیکن اول پاسخ دهید.

خیر به صرفه نیست. همان طور که در شکل پیدا است، اگر c=۵ باشد، نهایتاً ۲ امتیاز می‌گیریم که این از حالت عادی که ۳ امتیاز می‌گرفتیم بدتر است پس با c=۵ به صرفه نیست که نفر اول از قابلیتش استفاده کند.



۳. (۲۰ نمره، درجه سختی ۶) فرض کنید یک جدول به شکل زیر داریم که در هر خانه آن یک عدد تا یک رقم اعشار، در پایین هر ستون و روبه‌روی هر ردیف یک عدد صحیح نوشته شده است. حال می‌خواهیم اعداد درون جدول را به گونه ای به سمت بالا یا پایین گرد کنیم که مجموع اعداد هر ردیف با عدد روبه‌روی آن و مجموع اعداد هر ستون با عدد پایین آن برابر شود.

X_1	X_2	X_3	
۴.۶	۵.۷	۳.۷	۱۴
۱.۵	۱.۶	۳.۷	۷
۷	۶	۸	
X_4	X_5	X_6	

الف) این مسئله را به یک مسئله CSP تبدیل کنید.

ب) گراف محدودیت‌های آن را رسم کنید.

ج) با استفاده از روش forward checking و هیوریستیک‌های MRV و Degree با Backtrack مسئله را حل کنید.

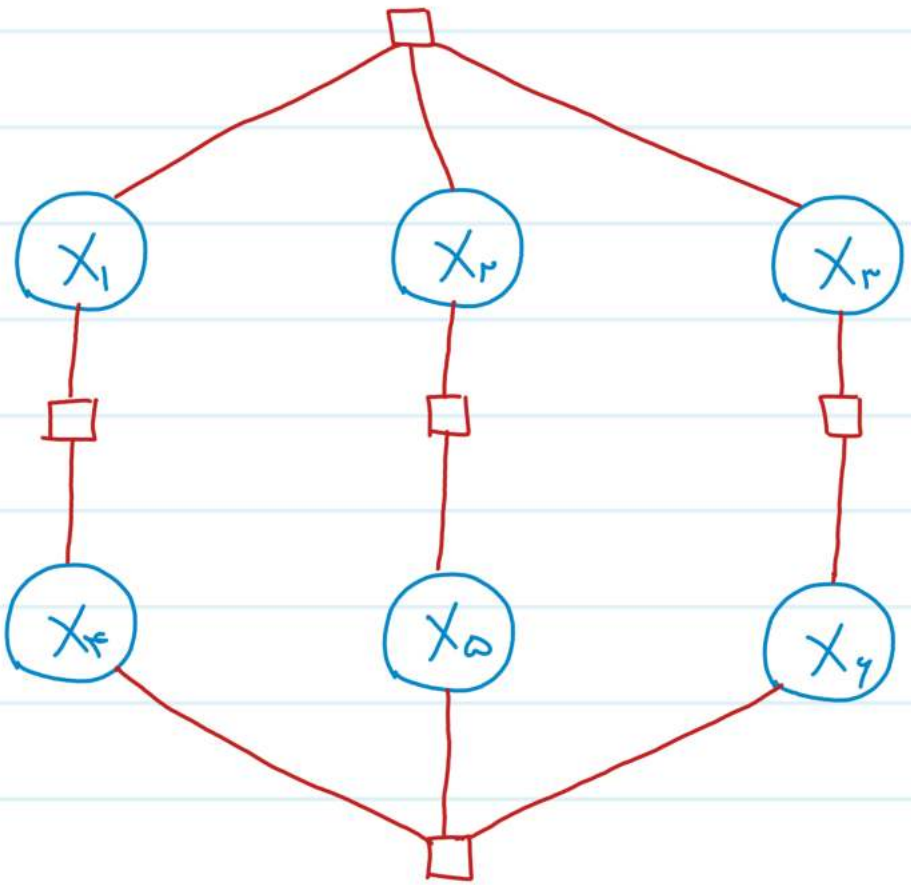
الف) مقادیر را به ترتیب $X_1, X_2, \dots, X_6$ می نامیم:

$$X = \{X_1, X_2, \dots, X_6\}$$

$$D_{X_i} = \{\lfloor X_i \rfloor, \lceil X_i \rceil\}$$

$$C = \{X_1 + X_4 = 7,\ X_2 + X_5 = 6,\ X_3 + X_6 = 8,\ X_1 + X_2 + X_3 = 14,\ X_4 + X_5 + X_6 = 7\}$$

ب)



ج) در ابتدا داریم: $\{X_1, X_2, X_3, X_4, X_5, X_6\}$ همه‌ی X_iها، سایز دامنه‌شان برابر است، و تعداد قیود روی هرکدام برابر است پس فرقی نمی‌کند، کدام اول انتخاب شود در backtrack. بدون کم شدن از کلیت مسئله، فرض کنیم X_1 اول انتخاب می‌شود: فرقی ندارد که به X_1، ۴ داده شود، در این صورت در forward checking تضمین داده می‌شود که مقداری برای X_4 نداریم پس X_1 برابر با ۵ می‌شود. سپس X_4 تنها مقدار ۲ را می‌تواند داشته باشد، پس در MRV چون کمترین تعداد value را دارد انتخاب می‌شود ← $X_1 = 5$ و $X_4 = 2$

حالا در ادامه همان X_i باقی مانده همه شرایطشان یکسان است، بدون کم شدن از کلیت مسئله فرض کنیم X_2 مقدار دهی می‌شود. X_2 برابر ۵ می‌شود، در ادامه برای X_5 فقط مقدار ۱ باقی مانده است، پس: $X_2 = 5, X_5 = 1$

سپس همه‌ی شرایط X_3 و X_6 یکسان است، بدون کم شدن از کلیت مسئله فرض کنیم اول X_3 مقدار دهی می‌شود. X_3 بخاطر قیدی که روی هر ردیف است فقط می‌تواند مقدار ۴ را بگیرد، پس $X_3 = 4$ می‌شود و X_6 هم تنها مقدار ۴ برایش باقی خواهد ماند ← جواب پیدا می‌شود و به صورت زیر است.

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

۴. (۲۰ نمره، درجه سختی ۷)

الف) با فرض آنکه $f, g : \mathbb{R} \to \mathbb{R}$ محدب‌اند و $t \geq 0$ نشان دهید که توابع زیر محدب هستند یا خیر.(در صورت محدب بودن اثبات کنید درغیراینصورت مثال نقض ارائه دهید.)

- ✓ $h(x) = f(x) + tg(x)$
- ✓ $k(x) = max\{f(x), g(x)\}$
- ✗ $r(x) = min\{f(x), g(x)\}$
- ✗ $s(x) = f(x)g(x)$

ب) تعیین کنید که مجموعه C که برای زوج مرتب‌های شامل یک بردار x و عدد حقیقی t به شکل زیر تعریف می‌شود یک مجموعه محدب می‌باشد یا خیر.

$$C = \{(x, t) \mid \|x\| \leq t\}$$

الف) f و g محدب اند پس داریم:

- $f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y)$

$g(\theta x + (1-\theta)y) \leq \theta g(x) + (1-\theta) g(y) \rightarrow t \cdot g(\theta x + (1-\theta)y) \leq \theta t \cdot g(x) + t(1-\theta) g(y)$

$\Rightarrow f(\theta x + (1-\theta)y) + t \cdot g(\theta x + (1-\theta)y) \leq \theta(f(x) + t \cdot g(x)) + (1-\theta)(f(y) + t \cdot g(y))$

- $h(\theta x + (1-\theta)y) \leq \theta h(x) + (1-\theta) h(y)$ باید نشان دهیم:

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

$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y)$, $g(\theta x + (1-\theta)y) \leq \theta g(x) + (1-\theta) g(y) \Rightarrow$

$max(f(\theta x + (1-\theta)y), g(\theta x + (1-\theta)y)) \leq max(\theta f(x) + (1-\theta) f(y), \theta g(x) + (1-\theta) g(y)) \leq$

$max(\theta f(x), \theta g(x)) + max((1-\theta) f(x), (1-\theta) g(x)) = \theta h(x) + (1-\theta) h(y)$

$\Rightarrow h(\theta x + (1-\theta)y) \leq \theta h(x) + (1-\theta) h(y)$

- غلط است، دو تابع $y = x^2$, $y = x^4$ را در نظر بگیرید:

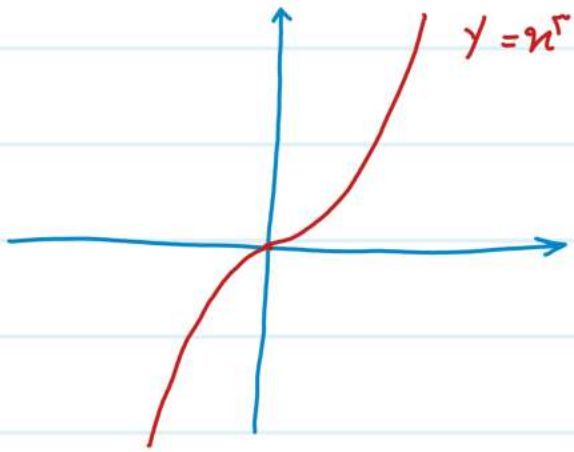
$h(0.9) = 0.6561$ $h(1.1) = 1.21$ $h(1) = 1$

خط واصل بین $h(0.9)$ و $h(1.1)$ را در نظر گیرید این خط در نقطه‌ی 1، برابر است با: $\frac{1.21 + 0.6561}{2} < 1$

که یعنی تابع حاصل محدب نیست.

● در نظر بگیرید $f(x)=x$ و $g(x)=x^3$، می دانیم که هردو توابعی محدب اند ولی حاصل ضرب آنها محدب نیست:

$s(x)=f(x)g(x)=x^3$:



واضح است که محدب نیست.

ب) می دانیم که طبق نامساوی کوشی برای دوبردار x و y داریم:

$$\|x+y\| \le \|x\| + \|y\|$$

پس اگر بردارهای θx و $(1-\theta)y$ را در نظر بگیریم، داریم:

$$\|\theta x + (1-\theta)y\| \le \|\theta x\| + \|(1-\theta)y\| = \theta\|x\| + (1-\theta)\|y\|$$

حالا اگر دوزوج (x,t) و (y,k) عضو C باشند، طبق بالا داریم که:

$$\|\theta x + (1-\theta)y\| \le \theta\|x\| + (1-\theta)\|y\| \le \theta t + (1-\theta)k$$

اگر $t=k$ باشد، سمت راست برابر مقدار آنها خواهد شد، همچنین اگر بدون کم شدن از کلیت مسئله t بزرگتر باشد، آنگاه:

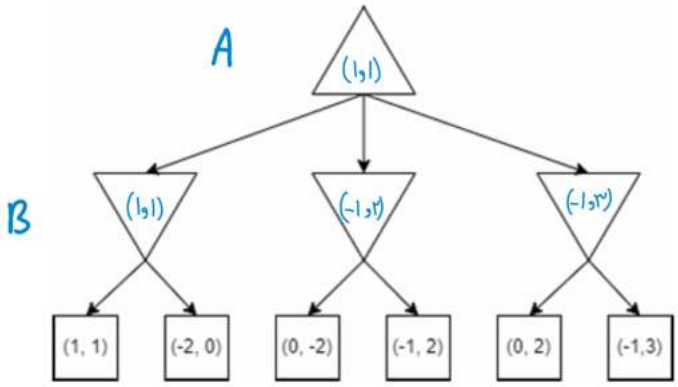
$$\theta k + (1-\theta)k \le \theta t + (1-\theta)k \le \theta t + (1-\theta)t \longrightarrow k \le \theta t + (1-\theta)k \le t$$

پس برای هردوزوج (x,t) و (y,k)، به ازای هرعبارت حاصل از آنها داریم:

$$\|\theta x + (1-\theta)y\| \le \theta\|x\| + (1-\theta)\|y\|$$

به عبارت دیگر، زوج مرتب $(\theta x + (1-\theta)y,\ \theta t + (1-\theta)k)$ بدست می آید که عضو C است چون $\|\theta x + (1-\theta)y\| \le \theta t + (1-\theta)k$

۵. (۲۰ نمره، درجه سختی ۸) فرض کنید در حال بررسی کردن یک بازی non zero-sum هستیم. درخت بازی مورد بررسی به صورت زیر می باشد: (توجه کنید که مثلث‌های رو به بالا و پایین نشان دهنده دو بازیکن متفاوت هستند وگرنه بدیهتا در چنین بازی‌هایی مشخص کردن دقیق یک بازیکن maximizer و یک بازیکن minimzier چندان معنی‌دار نیست چون شرط بازی‌های zero-sum یعنی $U_A(s)+U_B(s)=0$ دیگر برقرار نیست و بنابراین هر بازیکن به دنبال maximize کردن امتیاز خود خواهد بود.)



شکل ۲

هر جفت عدد در برگ‌ها به ترتیب امتیاز بازیکن اول و دوم را نمایش می‌دهد. بازیکن اول را A و بازیکن دوم را B می‌نامیم و بنابراین هر جفت عدد به فرمت $(U_A,\ U_B)$ می‌باشد.

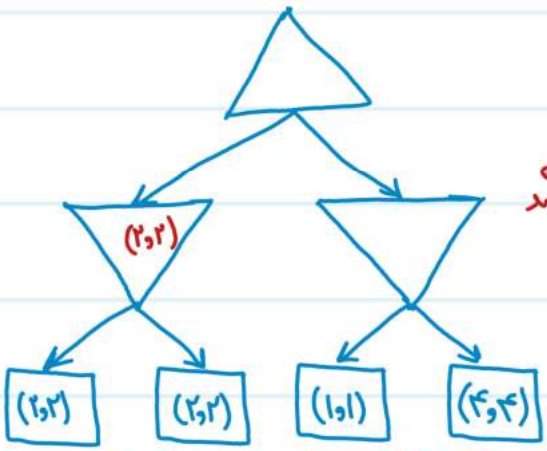
الف) مقادیر هر راس در درخت بازی مورد نظر را تکمیل نمایید.

ب) به طور خلاصه توضیح دهید چرا روش alpha-beta pruning در تعریف عام از بازی های non zero-sum قابل استفاده نمی‌باشد.

راهنمایی: برای مثال خود می‌توانید حالتی که شرط $U_A(s)=U_B(s)$ برای همه برگ‌ها برقرار باشد را مورد بررسی قرار دهید.

در minimax می‌دانیم که مقدار محاسبه شده برای ریشه (که فرض می‌کنیم بازیکن maxmizer باشد.) اصطلاحا یک مقدار worst-case می‌باشد؛ به این معنا که اگر بازیکن minimizer بهینه ترین عمل ممکن را انتخاب نکند نتیجه امتیاز maximizer هرگز بدتر نخواهد شد.

ب) مثال زیر را در نظر بگیرید:



نود سمت چپ برابر (۲و۲) خواهد شد،
در این هنگام در سمت راست (۱و۱) مشاهده خواهد شد
و pruning اتفاق می افتد چون ما فکر می‌کرده ایم که
رقیب انتخابی می‌کند که امتیاز ما مینیمم شود، درصورتی که
رقیب می‌خواهد امتیاز خودش را ماکزیمم کند و (۴و۴) را انتخاب می‌کند که برای ما هم (۴و۴) بهتر از (۲و۲) است
ولی چون pruning اتفاق افتاده است، حرکتی را خواهیم کرد که (۲و۲) منجر می‌شود.

یعنی pruning باعث شد حالت بهتر را از دست بدیم.

ه) یک شرط عمومی بیان کنید که تحت آن فرزند یک راس S می‌تواند هرس شود. شرط شما باید با درنظر گرفتن متغیرهایی چون $U_A(S)$ یا $U_B(S)$ برای راس S مربوطه، مقدار ϵ، α و ... بیان شود.

ج) آیا می‌توان گفت که برای یک بازی non zero-sum نیز مقدار محاسبه شده برای ریشه مشابه توضیحات داده شده worst-case می‌باشد؟ به طور خلاصه توضیح دهید.

خیر چون هر بازیکن در حال ماکزیمم کردن امتیاز خودش است و در این حین ممکن است حالتی به عنوان ریشه انتخاب شود که worst-case امتیاز ما نیست. مثلاً در مثال بخش الف، worst-case امتیاز ما برابر است با ۱- در صورتی که در ریشه برای ما امتیاز ۱ قرار دارد.

اکنون فرض کنید که بازی تقریبا zero-sum باشد به این معنی که شرط $| U_A(s) + U_B(s) | \leq \epsilon$ برای تمامی برگ های آن به ازای یک مقدار ϵ مشخص برابر برقرار باشد. مثلا درخت بازی ای که در ابتدای سوال رسم شده است برای مقدار $\epsilon = ۲$ یک بازی nearly zero-sum می‌باشد.

د) در یک بازی nearly zero-sum امکان هرس کردن وجود دارد. با در نظر گرفتن مقدار $\epsilon = ۲$ و تعمیم دادن alpha-beta pruning به بازی کنونی، راس هایی که در طی فرایند هرس کردن خط میخورند را مشخص کنید و توضیحی مختصر درباره الگوریتم در این حالت خاص بدهید. (فرض کنید فرایند هرس کردن به صورت استاندارد آن یعنی از چپ به راست و depth-first انجام می‌شود.)

هر شخصی دارد امتیاز خودش را ماکزیمم می‌کند، بنابراین در pruning اگر نوبت ما بود به این صورت عمل می‌کنیم: برای مثال سمت چپ‌ترین maximizer حریف a_0 باشد و maximizer بعدی a_1 باشد، همچنین فرض کنیم امتیاز ما در چپ‌ترین maximizer حریف b_0 باشد، حالا امتیاز حریف در maximizer بعدی حداقل a_1 است و امتیاز ما در آن حداکثر مقداری در بازه‌ی $[\epsilon - a_1, \epsilon - a_1]$ است. حالا اگر داشته باشیم $\epsilon - a_1 < b_0$، آنگاه کل نود های این maximizer، prune می‌شوند دیگر بررسی نمی‌شوند. به طور مشابه برای رقیب هم همین گونه داریم.

ه) یک شرط عمومی بیان کنید که تحت آن فرزند یک راس S می‌تواند هرس شود. شرط شما باید با درنظر گرفتن متغیرهایی چون $U_A(S)$ یا $U_B(S)$ برای راس S مربوطه، مقدار ϵ، α و ... بیان شود.

اگر در maximizer مربوط به A باشیم و به نود S ای در maximizer مربوط به B برسیم که داشته باشیم: $\epsilon - U_B(S) < \alpha$ چون مقداری که برای B در maximizer آن قرار می‌گیرد بزرگتر مساوی $U_B(S)$ است در نتیجه مقداری که برای A در این maximizer قرار می‌گیرد حداکثر در بازه‌ی $(U_B(S) - \epsilon \text{ و } \epsilon - U_B(S))$ است

وچون $\alpha > U_{B(s)} - \varepsilon$ است پس اینجا Pruning اتفاق می‌افتد.

همچنین اگر در maximizer مربوط به B باشیم و به نود s ای در minimizer مربوط به A برسیم که داشته باشیم: $\alpha > U_{A(s)} - \varepsilon$ ، مطابق استدلال قبلی، اینجا Pruning اتفاق می‌افتد.

$U_{B(s)} - \varepsilon < \alpha$ ۲-$U_{B(s)}$ ⇐ اگر در maximizer A باشیم و دریک راس B مربوط به maximizer B زیری داشته باشیم:

یا اگر در maximizer B باشیم و دریک راس A مربوط به maximizer A زیری داشته باشیم: $U_{A(s)} - \varepsilon < \alpha$ ۲-$U_{A(s)}$

آنگاه Pruning اتفاق می‌افتد. (شروط Pruning)

و) در یک بازی nearly zero-sum چه شرطی روی حداقل مقدار امتیازی که ممکن است توسط بازیکن اول کسب شود (برحسب U_A ریشه و ϵ) وجود دارد؟

می‌دانیم که اگر بازیکن دوم بهینه بازی کند، همواره امتیاز ما U_A ، امتیاز او U_B می‌شود ولی اگر بهینه بازی نکند، امتیاز او به جای U_B برابر با $U'_B = U_B - \delta$ ، $\delta \geq 0$ می‌شود، حالا داریم:

$$|U'_A + U'_B| \leq \varepsilon \rightarrow -\varepsilon \leq U'_A + U'_B \leq \varepsilon \rightarrow -\varepsilon \leq U'_A + U_B - \delta \rightarrow -\varepsilon + \delta - U_B \leq U'_A$$

$$\varepsilon - U_A \leq U_B \leq \varepsilon - U_A \rightarrow -\varepsilon + \delta - \varepsilon + U_A \leq U'_A \rightarrow U_A - 2\varepsilon + \delta \leq U'_A \ , \ \delta \geq 0 \rightarrow U'_A \geq U_A - 2\varepsilon$$

حالا نشان می‌دهیم که U'_A می‌تواند برابر با $U_A - 2\varepsilon$ شود:

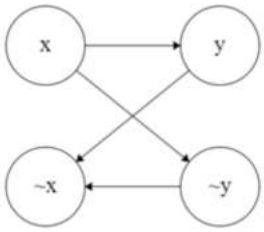
حالتی را در نظر بگیرید که $U_A + U_B = \varepsilon$ ، $U'_B = U_B$ ، $U'_A + U'_B = -\varepsilon$ ، آنگاه داریم که:

$$U'_A = -U'_B - \varepsilon = -U_B - \varepsilon = -(\varepsilon - U_A) - \varepsilon = U_A - 2\varepsilon$$ $U_A - 4$

پس کران پایین امتیاز نفر اول برابر است با $U_A - 2\varepsilon$.

$U_A - 4$

۶. (- نمره، درجه سختی ؟) (**سوال امتیازی**) می‌دانیم که در حالت کلی پیچیدگی زمانی حل مسئله CSP از اردر نمایی‌است. در این سوال میخواهیم که حالت خاصی از این مسئله به اسم 2-SAT را در اردر خطی حل کنیم. در این حالت خاص تمام متغیرها باینری (با دامنه ۰ یا ۱) هستند. همچنین تمامی قیود مسئله دوتایی و به شکل $a \vee b$ هستند. یعنی مثلا هیچ قیدی به شکل $a \vee b \vee c$ وجود ندارد.
برای حل ابتدا گرافی جهت دار می‌سازیم که به ازای هر متغیر مثل a دو راس متناظر a و $\neg a$ را قرار می‌دهیم. برای نمایش قید $p \vee q$ دو یال $\neg p \to q$ و $\neg q \to p$ را به گراف اضافه می‌کنیم. درواقع این دو یال به ترتیب معادل این هستند که اگر که گزاره p را False در نظر بگیریم، آنگاه حتما q باید True باشد. و اگر q را False در نظر بگیریم، آنگاه حتما p باید True باشد. به عبارتی از هم‌ارزی $p \vee q \equiv (\neg p \to q) \vee (\neg q \to p)$ استفاده شده است.
به عنوان مثال اگر قیدهای مسئله به شکل $(\neg x \vee y) \wedge (\neg y \vee \neg x)$ باشد، آنگاه گراف متناظر آن به شکل زیر خواهد بود:



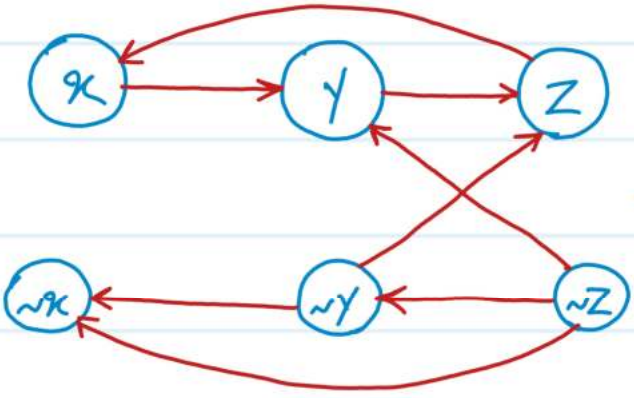
شکل ۳

الف) ابتدا مسئله را به صورت یک مسئله CSP بیان کرده و سپس گراف مدنظر را برای مسئله نمونه با قیدهای $(\neg x \vee y) \wedge (\neg y \vee z) \wedge (x \vee \neg z) \wedge (y \vee z)$ رسم کنید و یک جواب برای آن بنویسید.

$X = \{X_1, X_2, \dots, X_n\}$

$D_{X_i} = \{0, 1\}$

$C = \{X_i \vee X_j \equiv T \mid 1 \le i, j \le n,\ i \neq j,$ {در عبارتمان $X_i \vee X_j$ وجود داشته باشد} $\}$



→ جواب : $x = y = z = T$

ب) ادعا میکنیم یک مسئله 2-SAT جواب خواهد داشت اگر و تنها اگر هیچ یک از مؤلفه های قویا همبند این گراف به طور همزمان شامل یک متغیر و نقیض آن نباشد. این ادعا را اثبات کنید. (مؤلفه قویا همبند: زیرمجموعه‌ای از رئوس گراف که برای هر جفت راس آن مثل x و y مسیری جهتدار از x به y و برعکس وجود دارد.)

اگر یک متغیر و نقیضش در یک مولفه‌ی همبند باشند، خواهیم داشت:

اگر $X_i \equiv T$ ← همگی گزاره‌های $X_n \dots X_{i+1}$ باید هم‌ارز با T باشند چون $P \Rightarrow Q$ به شرط $P \equiv T$ زمانی درست است که Q هم هم‌ارز با T باشد، پس به صورت استقرایی $X_{i+1} \equiv T$ و $X_{i+2} \equiv T$ و... در نهایت $X_n \equiv T$ و درنتیجه $X_i' \equiv T$ ⁂

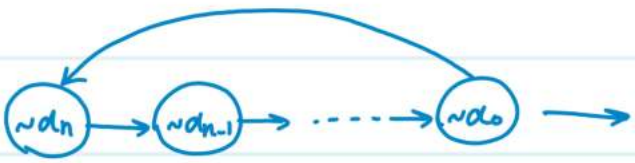
همچنین اگر $X_i \equiv F$ ← از X_i' شروع کنیم و با پیمایش نودها به X_i برسیم، چون $X_i' \equiv T$ پس تمام جملات بعدی هم باید هم‌ارز با T باشند، چون مؤلفه قویاً همبند است، پس حتماً از X_i' به X_i مسیر هست ← مانند استدلال بالا، $X_i \equiv T$ ⁂

اگر 2-SAT بخواهد جواب داشته باشد، یک متغیر و نقیض آن نباید در هیچ مولفه‌ی قویاً همبند باشند.

حالا باید نشان دهیم که اگر در هیچ مولفه همبندی یک متغیر و نقیضش نباشد، مسئله 2-SAT جواب دارد.

مولفه همبندی A را در نظر بگیرید:

می‌دانیم که هر عبارت $P \vee Q$ معادل $\neg P \to Q \wedge \neg Q \to P$ است. پس اگر داشته باشیم $a_i \to a_{i+1}$ باید $\neg a_{i+1} \to \neg a_i$ ، به ازای هر i ای این یک عبارت دوطرفه است، پس مولفه‌ای قویاً همبند به صورت زیر داریم:



A' می‌نامیم

هرجایی در A داشتیم $a_i \to a_{i+1}$، در اینجا عکس نقیضش را داریم و برعکس.

حالا یا این دو مجموعه هیچ یالی بینشان نیست یا فقط از یکی به دیگری یال هست چون اگر هردو به همدیگر یال داشته باشیم یعنی a_i ها و $\neg a_i$ ها در مولفه قویاً همبندمان هستند.

پس فقط یکی می‌تواند به دیگری یال داشته باشد، بعد کم شدن از کلیت مسئله فرض کنیم A به A' یال دارد.

حالا اگر راس P در A به راس $\neg q$ در A' یال داشته باشد، چون قیدها به شکل $2VS$ بوده اند، پس راس q در A هم به $\neg p$ در A' یال دارد.

همه‌ی متغیرهای دسته‌ی A را هم‌ارز با F قرار می‌دهیم ← تمام متغیرهای دسته‌ی A' هم‌ارز با T می‌شوند، همچنین داریم:

$$F \to F \to \dots \to F \equiv T \quad , \quad T \to T \to \dots \to T \equiv T \quad , \quad F \to T \equiv T$$

پس این مقدار دهی مشکلی ندارد فقط درصورتی مشکل خواهیم داشت که به نمونه‌ی زیر برخورد کنیم: (یعنی از یک A به یک F برسیم)

$$\neg h \in A' \to h \in A \quad , \quad K \in B \to \neg K \in B'$$

B' به A یال دارد پس کل A و A' و B و B' یک مولفه‌ی قویاً همبند است که یعنی یک متغیر و نائض در یک مولفه قویاً همبند قرار گرفته اند، که این در تناقض با فرض اصلی است ← چنین حالتی رخ نخواهد داد. پس از هیچ T ای به F نمی‌رسیم، پس این نوع مقداردهی قابل قبول است و مسئله‌ی 2-SAT جواب دارد اگر و تنها اگر در هر مولفه‌ی قویاً همبند آن هیچ متغیری همزمان با نقیضش حضور نداشته باشد.

داریم: $\neg h \to k \implies$ داریم: $\neg K \to h$ ←

ج) با فرض اینکه در مسئله شرط بخش ب برقرار است (یعنی حتما مقداردهی صحیحی دارد)، یک روش از $O(n+m)$ برای یافتن یک مقداردهی صحیح ارائه کنید. که در آن n تعداد متغیرهاست و m تعداد قیود. (راهنمایی: به مرتب‌سازی توپولوژیک مولفه‌ها فکر کنید.)

↳ $O(n+m)$

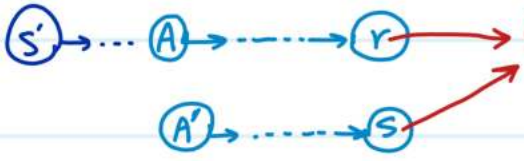
با توجه به بخش ب، گرافمان از تعدادی مولفه‌ی قویاً همبند درست شده است. حالا با استفاده از الگوریتم کوسارا جو مولفه‌های همبندی را بدست می‌آوریم. طبق بالا هم دیدیم برای هر مولفه‌ی قویاً همبند، یک مولفه‌ی قویاً همبند وجود دارد که شامل نقیض تمام متغیرهای اولی است و جهت یال‌هایش برعکس است.

حالا اگر مولفه‌ی A ای به مولفه با نقیض متغیرهایش یعنی A' مسیر داشت، تمام متغیرهای A را برابر F قرار می‌دهیم و تمام متغیرهای A' هم ارز با T خواهند شد و طبق بخش ب مشکلی نخواهیم داشت.

نحوه انجام، ابتدا از یک مولفه A شروع می‌کنیم و به ترتیب مولفه‌هایی که می‌توانیم برویم را یکی یکی می‌رویم، یک آرایه هم متناظر دیدن مولفه‌ها و یک آرایه متناظر دیدن معکوس یک مولفه تعریف می‌کنیم، حالا اگر در این مسیرمان مولفه‌ای همزمان با نقیضش دیده شده، اونی که اول دیده شده را تمام متغیرهایش را هم ارز F قرار می‌دهیم و دومی تمام متغیرهایش هم ارز T می‌شود،

در ضمن تمام مولفه‌های قبل از اولی هم ← این کار را در آخر انجام می‌دهیم *

تمامی مقادیر متغیرهایشان هم ارز با F می‌شود چون $T \to F \equiv F$، به این صورت بیمایش میزنیم، حالا سراغ مولفه‌ی قویاً همبند بعدی می‌رویم اگر دیده شده بود که هیچ دگرنه همین عملیات را انجام می‌دهیم، همچین موقع این کار اگر معکوس یک مولفه (مولفه‌ای که شامل نقیض تمام متغیرهای مولفه‌ی فعلی است) مقدار دهی شده بود، بر آن اساس تمام متغیرهای این مولفه را مقدار دهی می‌کنیم، * در نهایت یک بار از آخر به اول بیمایش می‌کنیم و از اولین جایی که یک مولفه با مقادیر همه‌ی متغیرها F دیدیم به عقب همه را F می‌گذاریم، طبق ب هم میدانیم که مبل از یک مولفه تماماً F، دو مولفه‌ی A و A' همزمان حضور ندارند پس مشکلی نداریم که همه‌ی این مولفه‌ها را تمامی اعضایشان را F قرار دهیم. به این مشکل که



(تعداد مولفه‌ها) تعداد

پس از دیدن تمام مولفه‌ها

نخواهیم خورد، زیرا می دانیم که در اولی تمام متغیرهای مولفه‌های قبل از x هم ارز با F اند و همچنین چون از $\acute{A}$ به s مسیر هست، از $\acute{s}$ به A هم باید مسیری داشته باشیم پس: $\acute{s}$ قبل از A قرار دارد از طرفی همه‌ی متغیرهای $\acute{s}$ هم ارز با T هستند که این خلاف فرض اولیه است که تمام متغیرهای مولفه‌های قبل از x هم ارز با F هستند.

در این الگوریتم اگر به مولفه ای رسیدیم که مولفه‌های بعدی مسیرش قبلاً ویزیت شده بودند، براساس آنها رو به عقب مقداردهی می کنیم. ← (تعداد متغیرها)O > (تعداد مولفه‌ها)O

در نهایت هر مولفه‌هایی که مقداردهی نشده اند، یعنی معکوسشان هم مقداردهی نشده چون اگر شده بود، خودشان هم مقداردهی می شدند، به دلخواه هر مولفه و معکوسش را می گیریم و برعکس هم به دلخواه یکی را کامل F و یکی را کامل T قرار می دهیم. ← $O(n)$

⇐ درکل اردر الگوریتممان برابر با $O(n+m)$ می شود و یک مقداردهی valid برای تمام متغیرهای مسئله 2-SAT پیدا خواهیم کرد.