دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

هوش مصنوعی

بهار ۱۴۰۵

آرش ماری‌اوریاد

مدت زمان: ۶۰ دقیقه

پاسخ کوئیز دوم (۱۰۰ نمره) CSP, Local, and Adversarial Search تاریخ: ۱۴۰۵/۰۲/۱۵

می خور که شیخ و حافظ و مفتی و محتسب چون نیک بنگری همه تزویر می‌کنند

«حافظ»

گردشی در کلاس هوش مصنوعی

امروز سری به کلاس هوش مصنوعی می‌زنیم. در سه گوشه‌ی کلاس، سه گروه مختلف مشغول فعالیت‌های جانبی هستند و هر کدام با چالش‌هایی روبه‌رو شده‌اند. شما باید در حل این چالش‌ها به آن‌ها کمک نمایید.

۱. (۳۰ نمره) در گوشه‌ای از کلاس، سبحان، آروین، هنا و نیما غرق در مطالعه‌ی رمان «۱۹۸۴» هستند. آن‌ها در حال بررسی کاراکتر وینستون اسمیت، قهرمان داستان، می‌باشند که در «وزارت حقیقت» کار می‌کند. وینستون هر روز با بیزاری تمام پشت میز کارش می‌نشیند و به بازنویسی اسناد تاریخی و جعل حقایق در راستای اهداف حزب می‌پردازد؛ شغلی که از آن متنفر است. بچه‌ها که از تماشای رنج وینستون به ستوه آمده‌اند، تصمیم می‌گیرند با دانش هوش مصنوعی خود به او کمک کنند تا کار امروز خود را زودتر تکمیل نموده و به خانه برگردد. هدف آن‌ها این است که این فرایند خسته‌کننده را به یک مسئله‌ی Local Search تبدیل کنند.

(آ) (۱۰ نمره) فرض کنید یک سند قدیمی دارای N کلمه و زبان دنیای رمان ۱۹۸۴ دارای V واژه‌ی منحصر به فرد است. بچه‌ها می‌خواهند سند را طوری تغییر دهند که ضمن معنادار بودن، رضایت حزب نیز کسب شود. این مسئله را با مشخص کردن فضای استیت‌ها، استیت اولیه، تعریف همسایگی، و تابع Fitness به عنوان یک مسئله‌ی Local Search مدل کنید.

(ب) (۵ نمره) نیما متوجه می‌شود که اگر فقط از الگوریتم Hill Climbing ساده استفاده کنند، وینستون مدام در وضعیت‌هایی گیر می‌کند که با تغییر یک یا دو کلمه نمی‌توان وضعیت سند را بهتر کرد، در حالی که می‌دانیم یک سند کاملا متفاوت وجود دارد که بسیار بیشتر مورد پسند حزب است. نام علمی این پدیده چیست و چرا رخ می‌دهد؟

(ج) (۵ نمره) با توجه به نگرانی نیما، هنا پیشنهاد می‌دهد که وینستون باید یک دفترچه یادداشت مخفی داشته باشد و لیستی از آخرین سندهایی که با اعمال تغییرات به آن‌ها رسیده است را در آن یادداشت کند. به کمک این دفترچه، وینستون می‌تواند سندهای تکراری را مشخص کرده و به سمت طراحی مجدد آن‌ها حرکت نکند. برای برآورده کردن پیشنهاد هنا، چه روش جست‌وجوی محلی را به اون معرفی می‌کنید و چرا؟

(د) (۵ نمره) سبحان که به آرام بودن معرف است، معتقد است وینستون نباید همیشه به دنبال بهترین تغییر ممکن در هر لحظه باشد. او می‌گوید گاهی اوقات، وینستون باید اجازه داشته باشد تغییری را بپذیرد که سند را کمی بدتر می‌کند! البته این اجازه نباید همیشگی باشد؛ در ابتدا که وقت زیاد است باید ریسک‌پذیرتر عمل کند، اما هرچه به پایان ساعت کاری نزدیک می‌شویم، باید سخت‌گیرتر شود و فقط به سمت بهبود حرکت کند. برای براورده کردن پیشنهاد سبحان، چه روش جست‌وجوی محلی را به اون معرفی می‌کنید و چرا؟

(ه) (۵ نمره) اما آروین که شخصیتی ادایی دارد، پیشنهاد می‌دهد که وینستون نباید تمام تمرکزش را روی یک نسخه بگذارد. او باید k نسخه‌ی مختلف را همزمان پیش ببرد، اما این نسخه‌ها نباید از هم بی‌خبر باشند! در هر مرحله، از بین تمام تغییرات ممکنی که می‌تواند روی این k نسخه بدهد، باید مجدداً k تا از بهترین‌ها را انتخاب کرده و نسخه‌های ضعیف‌تر را دور بریزد. برای براورده کردن پیشنهاد آروین، چه روش جست‌وجوی محلی را به اون معرفی می‌کنید و چرا؟

حل.

(آ) برای مدل کردن این مسئله به عنوان جست‌وجوی محلی، باید این چهار مورد را مشخص کنیم:

- فضای استیت‌ها: تمام ترکیب‌های ممکن از کلمات برای یک سند با طول N. چون هر کلمه می‌تواند یکی از V واژه‌ی موجود باشد، اندازه‌ی این فضا V^N است.
- استیت اولیه: همان سند قدیمی و اصلی است که وینستون باید تغییرش دهد و شامل N کلمه می‌باشد.
- تعریف همسایگی: ساده‌ترین تعریف این است که سندهایی همسایه در نظر گرفته بشوند که دقیقاً در یک کلمه (یا تعداد بسیار کمی کلمه) با سند فعلی تفاوت دارند.
- تابع Fitness: تابعی است که به هر سند یک امتیاز می‌دهد. مثلا این امتیاز بر اساس دو معیار تعیین می‌شود: نخست اینکه جملات از نظر دستوری و معنایی درست باشند و دوم اینکه محتوای سند چقدر با دروغ‌ها و اهداف حزب هم‌خوانی دارد. هرچه بیشتر باب میل حزب باشد، این تابع خروجی بالاتری می‌دهد.

(ب) نیما دقیقاً دارد به مشکل ماکزیمم محلی یا Local Optimum اشاره می‌کند. این پدیده به این دلیل رخ می‌دهد که الگوریتم Hill Climbing فقط همسایه‌های بسیار نزدیک را می‌بیند و همیشه به سمت همسایه‌ای می‌رود که فیتنس بهتری دارد. وقتی وینستون به سندی می‌رسد که تمام همسایه‌های آن (با تغییر یک کلمه) امتیاز کمتری دارند، الگوریتم همان‌جا متوقف می‌شود. در واقع وینستون در یک قله‌ی کوچک گیر می‌کند و نمی‌تواند ببیند که اگر چند تغییر موقتاً بد روی سند اعمال کند، ممکن است به قله‌ای بسیار بلندتر (همان سند ایده‌آل حزب) برسد.

(ج) پیشنهاد هنا همان روش Tabu Search است. در این روش ما یک لیست ممنوعه یا Tabu List داریم که استیت‌های اخیراً دیده شده را در آن ذخیره می‌کنیم که همان دفترچه‌ی مخفی وینستون می‌شود. این کار باعث می‌شود الگوریتم در حلقه‌های تکراری نیفتد و وقتی در یک ماکزیمم محلی گیر می‌کند، مجبور بشود مسیرهای جدیدی را امتحان کند، به جای اینکه مدام بین چند سند تکراری جابه‌جا بشود.

(د) حرف‌های سبحان دقیقاً اساس کار الگوریتم Simulated Annealing را توضیح می‌دهد. در این روش ما یک پارامتر دما داریم که به مرور زمان کاهش پیدا می‌کند. در ابتدای کار که دما بالاست، یعنی همان ساعات اولیه‌ی کار وینستون، الگوریتم با احتمال بیشتری تغییرات بدتر را می‌پذیرد تا بتواند از ماکزیمم‌های محلی فرار کند و فضای بیشتری را بگردد. اما هرچه به پایان ساعت کاری نزدیک می‌شویم و دما کم می‌شود، احتمال پذیرش تغییرات بد پایین می‌آید و الگوریتم حریصانه عمل می‌کند تا به بهترین جواب در همان محدوده‌ای که هست برسد.

(ه) آروین دارد الگوریتم Local Beam Search را پیشنهاد می‌دهد. تفاوت این روش با این که k جست‌وجوی تصادفی را جداگانه و مستقل جلو ببریم این است که اینجا نسخه‌ها با هم در ارتباط هستند. وقتی در هر مرحله تمام همسایه‌های این k نسخه را تولید می‌کنیم و فقط k مورد برتر را از بین کل آن‌ها نگه می‌داریم، اگر یک نسخه به یک مسیر بسیار عالی رسیده باشد، ممکن است بیشتر انتخاب‌های بعدی از بین همسایه‌های همان نسخه‌ی موفق باشند و نسخه‌هایی که در مسیرهای ضعیف‌تری افتاده‌اند کلاً حذف بشوند.

۲. (۵۰ نمره) در گوشه‌ای دیگر از کلاس، پنج هم‌اتاقی به نام‌های حامد، آرش، مشتاق، محسن و امیرحسین دور یک میز گرد نشسته‌اند تا کلم‌پلو شیرازی بخورند. ۵ صندلی دور این میز قرار دارد که به صورت دایره‌ای از ۱ تا ۵ شماره‌گذاری شده‌اند (صندلی ۱ و ۵ کنار یکدیگر هستند). محدودیت‌های زیر برای نشستن آن‌ها وجود دارد:

- آرش و مشتاق در حال کار روی یک پروژه مشترک هستند و حتما باید کنار هم بنشینند تا بتوانند لپ‌تاپ هم را چک کنند.
- محسن و حامد در حال حاضر با یک دیگر در بیف هستند و اصلا نباید کنار هم بنشینند.
- امیرحسین به دلیل نزدیکی به پنجره، فقط صندلی‌های با شماره بزرگتر از ۳ را می‌پسندد.
- حامد فقط روی صندلی‌های با شماره فرد می‌نشیند.
- از طرف دیگر مشتاق فقط روی صندلی‌های با شماره زوج می‌نشیند.
- محسن به دلایلی نامعلوم می‌خواهد شماره‌ی صندلی بزرگ‌تری نسبت به آرش داشته باشد.
- مشخصا هیچ دو نفری نمی‌توانند روی یک صندلی بنشینند.

(آ) (۲۰ نمره) این مسئله را با مشخص کردن متغیرها، دامنه‌ی هر متغیر، و قیود به صورت یک مسئله‌ی CSP مدل‌سازی کرده و گراف محدودیت‌های مسئله را رسم نمایید. توجه نمایید که محدودیت‌های سوم، چهارم، و پنجم به عنوان دامنه‌ی اولیه متغیرها در نظر گرفته می‌شوند.

(ب) (۱۵ نمره) با اجرای الگوریتم AC3 به عنوان یک پیش‌پردازش، ویژگی Arc Consistency را میان جفت متغیرهای مسئله برقرار سازید و دامنه‌ی متغیرها را تا جای ممکن محدودتر نمایید.

(ج) (۱۵ نمره) حال با استفاده از نتایج حاصل از بخش قبل و قاعده‌ی MRV، اولین متغیری که باید مقداردهی شود را مشخص کنید. در صورت تساوی، می‌توانید اولویت را به متغیری بدهید که بیشترین تعداد قید را با سایر متغیرهای دیگر دارد. سپس با استفاده از قاعده‌ی LCV، اولین مقدار انتخاب شده برای آن متغیر را مشخص نمایید..

حل.

(آ) برای مدل‌سازی این مسئله به عنوان یک CSP، سه‌گانه‌ی متغیرها، دامنه‌ها و قیود را به این شکل تعریف می‌کنیم:

مجموعه متغیرها را با V نشان می‌دهیم که شامل حرف اول اسم هر شخص است:

$$V = \{A, M, H, Mo, Am\}$$

دامنه‌ی اولیه متغیرها را با توجه به محدودیت‌های سوم، چهارم و پنجم مسئله به صورت مجموعه‌ی D تعریف می‌کنیم: $D_{Am} = \{۴, ۵\}$ $D_H = \{۱, ۳, ۵\}$ $D_M = \{۲, ۴\}$ $D_A = \{۱, ۲, ۳, ۴, ۵\}$ $D_{Mo} = \{۱, ۲, ۳, ۴, ۵\}$

مجموعه قیود مسئله را می‌توان به صورت زیر بیان نمود:

- قید اول (همه متفاوت باشند): برای هر $x, y \in V$ داریم $x \neq y$
- قید دوم (مجاورت آرش و مشتاق): $|A - M| = ۱ \vee \{A, M\} = \{۱, ۵\}$
- قید سوم (عدم مجاورت حامد و محسن): $|H - Mo| \neq ۱ \wedge \{H, Mo\} \neq \{۱, ۵\}$
- قید چهارم (صندلی بزرگ‌تر محسن نسبت به آرش): $Mo > A$

گراف محدودیت‌ها: در این گراف، رئوس همان مجموعه V هستند. از آنجا که قید اول ایجاب می‌کند هیچ دو نفری شماره صندلی یکسان نداشته باشند، یک یال بین هر جفت متغیر وجود دارد. پس گراف پایه ما یک گراف کامل یا $K_۵$ است. قیود دیگر نیز روی همین یال‌های موجود (مثلاً یال بین A و M) اعمال می‌شوند و ساختار کلی گراف را تغییر نمی‌دهند.

(ب) با اجرای پیش‌پردازش AC3، سازگاری کمان‌ها را بررسی کرده و دامنه‌ها را به‌روز می‌کنیم:

ابتدا کمان مربوط به قید $Mo > A$ را بررسی می‌کنیم. برای اینکه به ازای هر مقدار در D_A یک پشتیبان در D_{Mo} وجود داشته باشد، مقدار ۵ باید از D_A حذف شود (زیرا اگر $A = ۵$ باشد، هیچ مقدار بزرگ‌تری برای محسن وجود ندارد). همچنین مقدار ۱ نیز از D_{Mo} حذف می‌شود. پس تا اینجا:
$D_{Mo} = \{۲, ۳, ۴, ۵\}$ $D_A = \{۱, ۲, ۳, ۴\}$

حالا به سراغ کمان قید مجاورت A و M می‌رویم. دامنه‌ی مشتاق $D_M = \{۲, ۴\}$ است. اگر A بخواهد مقادیر ۲ یا ۴ را به خود اختصاص دهد، آنگاه مقدار مجازی در دامنه‌ی M باقی نمی‌ماند. لذا دامنه‌ی جدید آرش برابر می‌شود با: $D_A = \{۱, ۳\}$

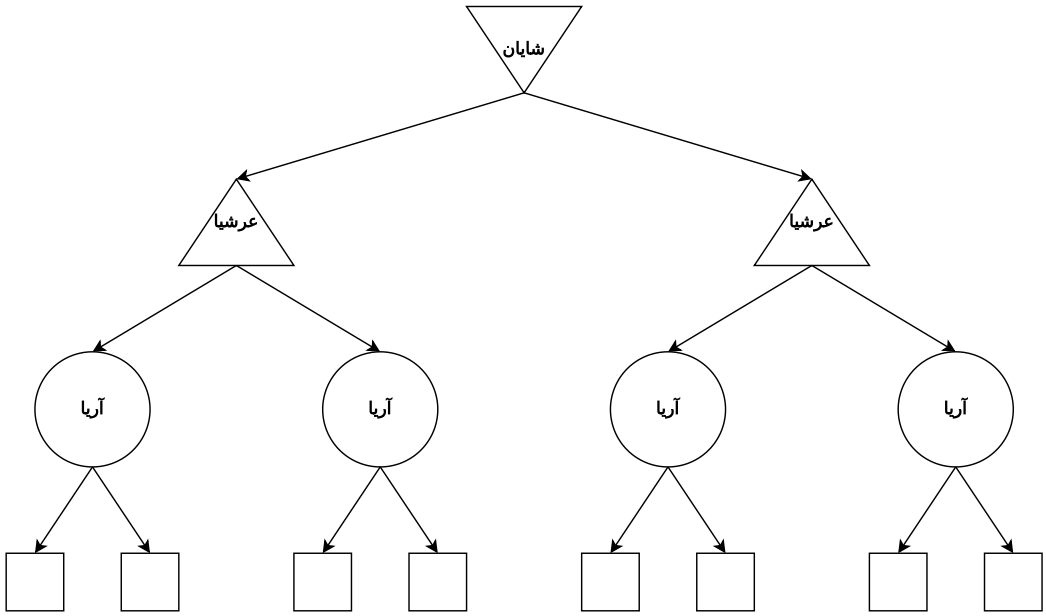
بررسی سایر کمان‌ها (مانند عدم مجاورت H و Mo) با دامنه‌های فعلی، نشان می‌دهد که برای تمام اعضای موجود در دامنه‌ها، حداقل یک مقدار سازگار در متغیر مقابل وجود دارد و عضو جدیدی حذف نمی‌شود.
پس دامنه‌های نهایی به این شکل خواهند بود: $D_H = \{۱, ۳, ۵\}$ $D_M = \{۲, ۴\}$ $D_A = \{۱, ۳\}$
$D_{Am} = \{۴, ۵\}$ $D_{Mo} = \{۲, ۳, ۴, ۵\}$

(ج) برای انتخاب اولین متغیر، از قاعده‌ی MRV استفاده می‌کنیم. باید متغیری را انتخاب کنیم که اندازه مجموعه‌ی دامنه‌اش (یعنی $|D|$) کمترین مقدار را داشته باشد: $|D_A| = ۲$ $|D_M| = ۲$ $|D_{Am}| = ۲$
همان‌طور که قابل مشهده است، بین سه متغیر حالت تساوی ایجاد شده است. برای رفع این بن‌بست از قاعده‌ی مطرح شده در صورت سوال استفاده می‌کنیم. در گراف مسئله، آرش علاوه بر قید تفاوت همگانی، در دو قید دیگر (مجاورت با مشتاق و کوچک‌تر بودن از محسن) نیز دخالت مستقیم دارد که او را به محدودترین متغیر تبدیل می‌کند. پس متغیر A را برای مقداردهی انتخاب می‌کنیم.

حالا باید برای متغیر A از بین مقادیر مجموعه‌ی $\{۱, ۳\}$ یکی را انتخاب کنیم. طبق قاعده‌ی LCV، باید مقداری را برگزینیم که گزینه‌های کمتری را از دامنه‌ی سایر متغیرها حذف کند: اگر $A = ۱$ را انتخاب کنیم: شرط $Mo > A$ به راحتی برقرار است و دامنه‌ی محسن دست‌نخورده می‌ماند ($Mo \in \{۲, ۳, ۴, ۵\}$). اگر $A = ۳$ را انتخاب کنیم: شرط $Mo > A$ باعث می‌شود مقادیر ۲ و ۳ از دامنه‌ی محسن حذف شوند و D_{Mo} به مجموعه‌ی $\{۴, ۵\}$ محدود شود.

چون انتخاب $A = ۱$ دامنه‌ی متغیر محسن را بسیار کمتر محدود می‌کند، طبق قاعده‌ی LCV مقدار ۱ را به عنوان اولین مقدار به متغیر A اختصاص می‌دهیم.

۳. (۲۰ نمره) در آخرین بخش کلاس، تعدادی از بچه‌ها تصمیم گرفته‌اند یک دست «سکرت هیتلر» (Secret Hitler) بازی کنند. بازی به لحظات حساس و پایانی خود رسیده است و درخت تصمیم‌گیری بازی (مطابق تصویر) برای سه حرکت آینده به شکل زیر مدل شده است:



- بازی از سه نقش لیبرال (نقش مثبت)، فاشیست (نقش منفی) و هیتلر (نقش منفی) تشکیل شده است و هیتلر ناآگاه هست و یار خود (فاشیست) را نمی‌شناسد.
- شایان در نقش فاشیست قرار دارد. هدف او کمینه‌کردن شانس پیروزی لیبرال‌هاست. اون به عنوان ریشه‌ی درخت (بازیکن اول) می‌تواند دو کنش مختلف انجام دهد و سپس نوبت به عرشیا می‌رسد.
- عرشیا در نقش لیبرال است و می‌خواهد شانس پیروزی تیمش (لیبرال‌ها) را بیشینه کند. عرشیا نیز دو کنش متفاوت می‌تواند انجام دهد و سپس نوبت به آریا می‌رسد.
- آریا در نقش هیتلر است. از آنجایی که هیتلر در این بازی یاران فاشیست خود را نمی‌شناسد، کاملا سردرگم شده و به صورت تصادفی (با احتمال ۵۰/۵۰ برای هر تصمیم) بازی می‌کند. آریا نیز دو کنش متفاوت می‌تواند انجام دهد و پس از آن بازی خاتمه می‌یابد.

مربع‌های پایین درخت (برگ‌ها) نمایانگر امتیاز نهایی لیبرال‌ها در پایان بازی هستند. با فرض اینکه از الگوریتم Expectiminimax مجهز به هرس آلفا-بتا (Alpha-Beta Pruning) استفاده می‌کنیم و گره‌ها از چپ به راست پیمایش می‌شوند، ارزش ۸ برگ پایین درخت (از چپ به راست) را با اعدادی صحیح با ذکر استدلال به گونه‌ای مقداردهی کنید که در حین اجرای الگوریتم، دقیقاً دو برگ آخر سمت راست توسط الگوریتم هرس شوند و اصلاً دیده نشوند.

حل.

برای مدل‌سازی دقیق‌تر، گره‌ها را نام‌گذاری می‌کنیم: گره‌ی ریشه که شایان (کمینه‌ساز) است را با Min نشان می‌دهیم. دو گره‌ی لایه‌ی بعد که عرشیا (بیشینه‌ساز) است را به ترتیب از چپ به راست Max_1 و Max_2 می‌نامیم. چهار گره‌ی دایره‌ای که آریا (تصادفی با احتمال ۰/۵) است را به ترتیب C_1, C_2, C_3, C_4 نام‌گذاری می‌کنیم. ارزش ۸ برگ پایین درخت را نیز از چپ به راست با v_1 تا v_8 مشخص می‌کنیم.

روند پیمایش درخت و استدلال برای مقداردهی برگ‌ها به این شکل است:

الگوریتم از ریشه (Min) با مقادیر اولیه $\alpha = -\infty$ و $\beta = +\infty$ شروع می‌شود و به سمت چپ یعنی گره Max_1 می‌رود. گره Max_1 باید ارزش دو فرزند خود یعنی C_1 و C_2 را محاسبه کند.

برای گره C_1، الگوریتم ارزش برگ‌های اول و دوم را می‌خواند. فرض کنیم مقادیر را این‌گونه تنظیم کنیم:

$$v_1 = 2$$

$$v_۲ = ۴$$

از آنجا که آریا کاملاً تصادفی بازی می‌کند، ارزش این گره برابر با امید ریاضی (میانگین) برگ‌هاست:

$$C_۱ = (۰/۵ \times ۲) + (۰/۵ \times ۴) = ۳$$

سپس الگوریتم سراغ گره $C_۲$ می‌رود و برگ‌های سوم و چهارم را می‌خواند:

$$v_۳ = ۴$$

$$v_۴ = ۶$$

$$C_۲ = (۰/۵ \times ۴) + (۰/۵ \times ۶) = ۵$$

حالا گره $Max_۱$ که هدفش بیشینه‌سازی است، بین ۳ و ۵، مقدار بزرگ‌تر یعنی ۵ را انتخاب می‌کند. پس ارزش زیردرخت سمت چپ مشخص می‌شود. گره ریشه (Min) که به دنبال کمینه‌سازی است، متوجه می‌شود که تا اینجا حداقل می‌تواند یک مسیر با ارزش ۵ را به حریف تحمیل کند.

حالا الگوریتم به سمت راست درخت یعنی گره $Max_۲$ می‌رود تا ببیند آیا مسیر بهتری (با ارزش کمتر از ۵) برای شایان وجود دارد یا خیر. گره $Max_۲$ ابتدا ارزش فرزند اولش یعنی $C_۳$ را با دیدن برگ‌های پنجم و ششم ارزیابی می‌کند:

$$v_۵ = ۸$$

$$v_۶ = ۱۰$$

$$C_۳ = (۰/۵ \times ۸) + (۰/۵ \times ۱۰) = ۹$$

در این لحظه، گره $Max_۲$ می‌داند که اگر این مسیر را ادامه دهد، ارزشی حداقل برابر با ۹ به دست می‌آورد (زیرا او بیشینه‌ساز است و اگر فرزند بعدی‌اش از ۹ بزرگ‌تر باشد آن را انتخاب می‌کند). اما گره ریشه (Min) به دنبال کمترین عدد ممکن است و از قبل تضمین شده که از سمت چپ می‌تواند مقدار ۵ را به دست بیاورد.

چون گره ریشه هرگز مقدار ۹ (یا بیشتر از آن) را به جای ۵ انتخاب نخواهد کرد، پیمایش بقیه‌ی زیردرخت سمت راست کاملاً بی‌فایده است. در زبان ریاضی، چون مقدار محاسبه شده در این مرحله (که معادل ۹ است) بزرگ‌تر یا مساوی مقدار β (که ۵ بود) شده است، شرط هرس آلفا-بتا برقرار می‌شود.

بنابراین، الگوریتم بلافاصله ارتباط با گره $C_۴$ را هرس می‌کند و برگ‌های هفتم و هشتم اصلاً دیده نمی‌شوند. مقادیر این دو برگ می‌تواند هر عدد دلخواهی باشد.