هوش مصنوعی

فصل ششم : جستجوی تخاصمی

دانشگاه پیام نور شهرکرد

مدرس:ایمان مختاری

محیط های چند عاملی

- در محیطهای چند عاملی هر عامل باید فعالیت سایر عاملها و تأثیر آنها بر روند کار خود را در نظر بگیرد.

- رفتارهای غیر قابل پیش‌بینی عاملهای دیگر می‌تواند باعث بروز مقتضیات بسیاری در فرآیند حل مسأله شود.

- محیطهای چند عاملی :
  - همکار
  - رقابتی

بازیها

- محیطهای رقابتی که در آنها هدف هر یک از عاملها در تعارض با دیگر عاملها می‌باشد، باعث پیدایش مسائل جستجوی تخاصمی گردید.

- نظریه بازیهای ریاضی، هر محیط چند عاملی را به صورت یک بازی در نظر می‌گیرد، به شرط این که اثر هر عامل بر دیگر عاملها "معنی‌دار" باشد.

نظریه بازیهای ریاضی شاخه ای از علم اقتصاد

بازیها

- در هوش مصنوعی، "بازیها" نوع خاصی از مسائل به شمار می‌روند.

- این مفهوم به معنای محیطهایی معیّن و کاملاً رؤیت‌پذیر می‌باشد که در آن، دو عامل به صورت نوبتی عمل می‌کنند و مقادیر امتیاز طرفین در نهایت و پس از خاتمه بازی باید برابر و مخالف یکدیگر باشد.

بازیها

- برای مثال، اگر یک بازیکن در شطرنج یک بازی را به نفع خود خاتمه دهد دارای امتیاز (۱+) و امتیاز طرف مقابل (۱-) خواهد بود.

- وجود همین تضاد در توابع امتیاز عاملها باعث تخاصمی شدن شرایط می‌شود.

در بازی مثل شطرنج فاکتور انشعاب حدود ۳۵ می باشد و یک بازیکن حدودا ۵۰ حرکت در یک بازی می کند. لذا درخت جستجوی حدود $۳۵^{۱۰۰}$ ایجاد می شود

تصمیمهای بهینه در بازیها

- حالت اوّلیّه: شامل موقعیت اوّلیه بازی و شخصی که باید بازی را شروع نماید، می‌باشد.
- تابع پسین: که لیستی از زوجهای (*move, state*) را بر می‌گرداند.
  - حالت نتیجه
  - حرکت مجاز
- آزمون پایانی، که مشخص می‌کند چه زمانی بازی به اتمام رسیده است.
- حالات پایانی: حالتهایی که در آنها بازی به اتمام می‌رسد.

تصمیمهای بهینه در بازیها

تابع هدف یا تابع تسویه

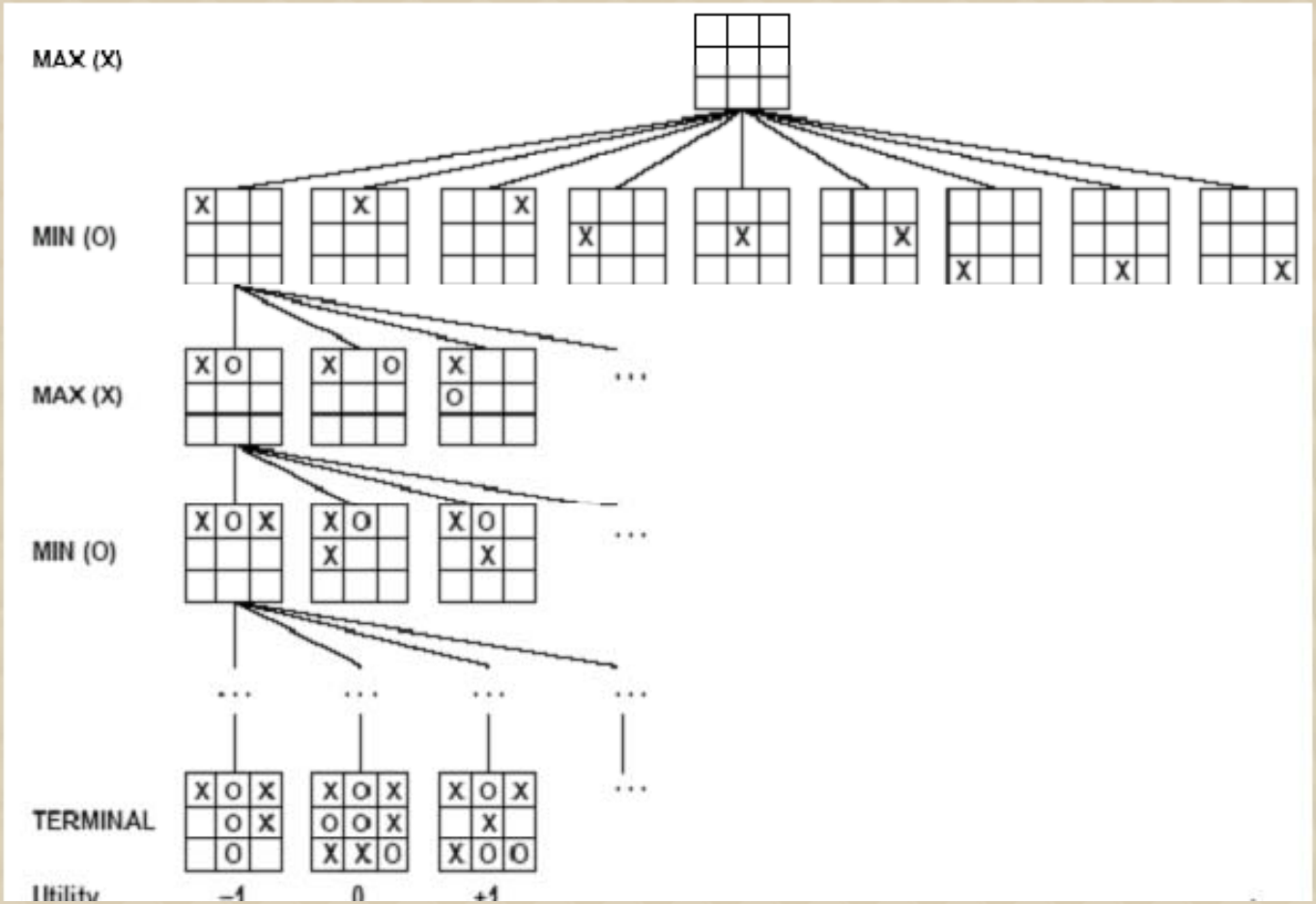
- تابع سودمندی: به حالات پایانی یک مقدار عددی اختصاص می‌دهد.

- در شطرنج نتیجه پایانی برد، باخت یا مساوی خواهد بود. که امتیاز هر یک به ترتیب ۱+، ۱- و ۰ می‌باشد.

- برخی از بازیها دارای نتایج متنوعتری می‌باشند؛ مثلاً در تخته نرد، این مقادیر از ۱۹۲- تا ۱۹۲+ تغییر می‌کند.

درختِ بازی

- حالت اوّلیه و حرکات مجاز برای هر طرف، درختِ بازی را برای آن بازی، تعریف می‌کند



درختِ بازی

- *MAX* در اوّلین حرکت، دارای ۹ انتخاب می‌باشد. با هر حرکت *MAX* یک *X* و *MIN* یک *O* به نوبت، در خانه مربوطه قرار می‌دهند تا در نهایت به یک حالت پایانی برسیم

- عدد نوشته شده روی هریک از گرههای پایانی، مبین میزان سودمندی آن حالت از دیدگاه طرف *MAX* می‌باشد. مقادیر بزرگ برای *MAX* مناسب و برای *MIN*، نامناسب می‌باشند.

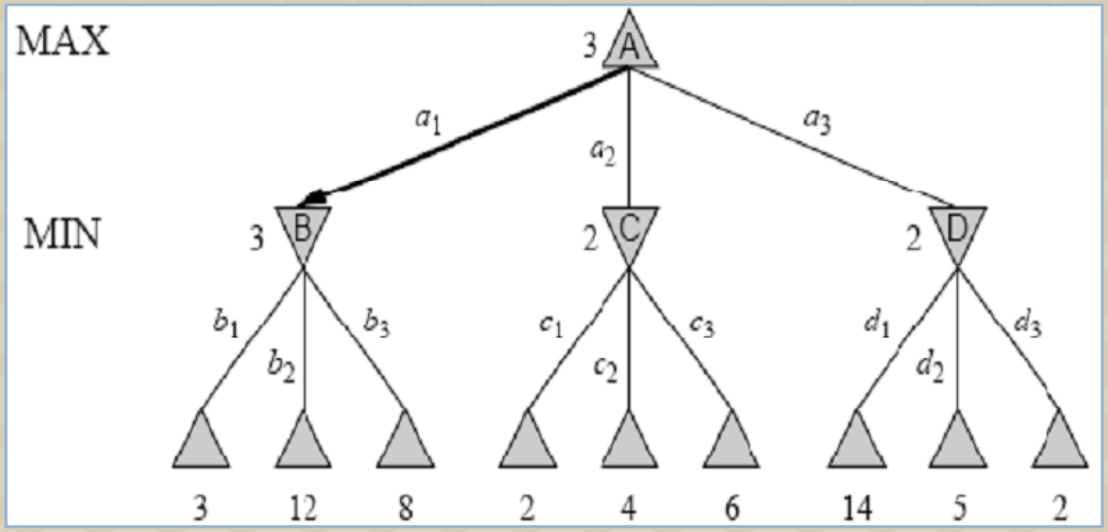
راهبردهای بهینه

- در یک مسأله جستجوی عادی، راه حل بهینه ترکیبی از حرکتهایی خواهد بود که در نهایت به یک حالت هدف (یک حالت پایانی به نام بُرد) منتهی گردد.

راهبرد اقتضایی یا استراتژی محتمل

- در یک بازی از آنجا که ***MIN*** هم، یک طرفِ بازی می‌باشد، ***MAX*** باید یک راهبرد اقتضایی تشکیل دهد که حرکت ***MAX*** را در حالت اوّلیه و سپس حرکتهای آن را در تمامی حالات ممکن بعدی که تحت تأثیر حرکتهای ***MIN*** نیز می‌باشد، مشخص کند.

- حرکتهای ممکن برای MAX در گره ریشه با a_1 و a_2 و a_3 نشان داده شده‌اند و پاسخهای ممکن برای MIN به حرکت a_1 برابر است با b_1 ، b_2 و b_3 و به همین ترتیب.



- این بازی ویژه با انجام یک حرکت از MAX و یک حرکت از MIN به اتمام می‌رسد.

- مقادیر سودمندی مربوط به حالات پایانی در این بازی از ۲ تا ۱۴ تغییر می‌کند.

راهبردهای بهینه

- با در نظر گرفتن یک درخت بازی، راهبرد بهینه می‌تواند به وسیله محاسبه مقدار بیشینه‌کمینه مربوط به هر گره، مشخص شود. MINIMAX-VALUE(n)

- مقدار بیشینه‌کمینه مربوط به هر گره (مثلاً برای MAX) برابر است با میزان سودمندیِ قرار داشتن در حالت متناظر با آن گره با فرض آن که هر دَو بازیکن بازی را از آن گره شروع و تا پایان به صورت بهینه ادامه دهند.

راهبردهای بهینه

- مقدار بیشینه‌کمینه یک حالت پایانی، برابر با مقدارسودمندی آن خواهد بود.

- اگر امکان انتخاب وجود داشته باشد:
  - MAX تمایل به حرکت به حالتی دارد که دارای بیشترین مقدار باشد
  - MIN تمایل به حرکت به حالتی را دارد که دارای کمترین مقدار باشد.

تابع چند حالته برای MINMax_Value

$$Minimax\text{-}Value(n) = \begin{cases} UTILITY(n) & \text{if } n \text{ is a terminal state} \\ \max_{s \in Successor(n)} MINIMAX\text{-}VALUE(s) & \text{if } n \text{ is a MAX node} \\ \min_{s \in Successor(n)} MINIMAX\text{-}VALUE(s) & \text{if } n \text{ is a MIN node} \end{cases}$$

الگوریتم بیشینه‌کمینه

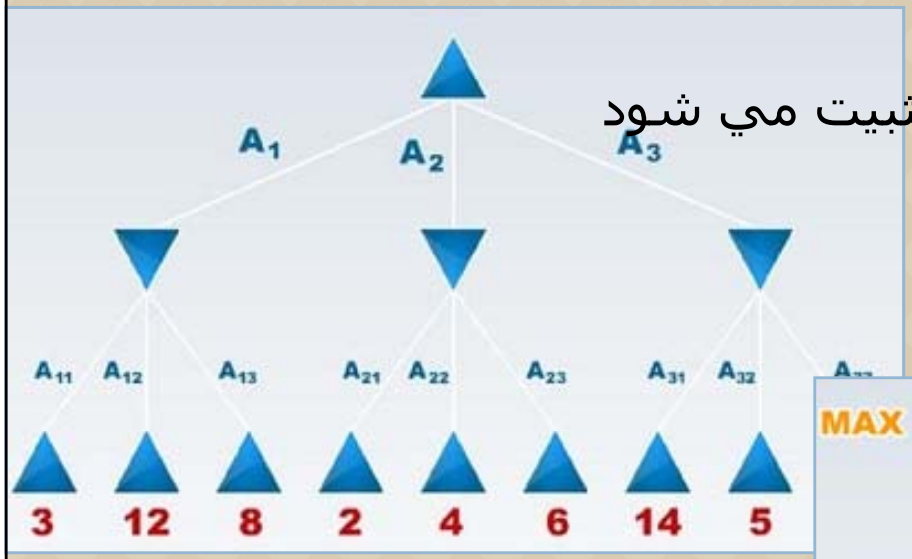
- الگوریتم بیشینه‌کمینه انتخاب بیشینه‌کمینه را بر اساس حالت فعلی، محاسبه می‌کند.

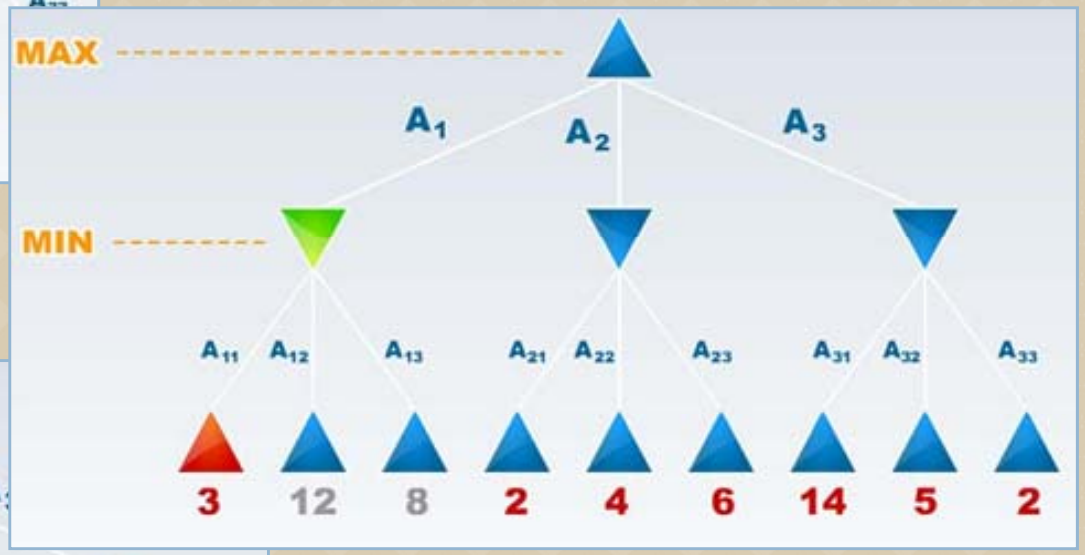
- این الگوریتم برای هر حالت پسین بیشینه‌کمینه با استفاده مستقیم از معادلات تعریف شده، از یک محاسبه بازگشتی ساده برای مقادیر استفاده می‌کند.

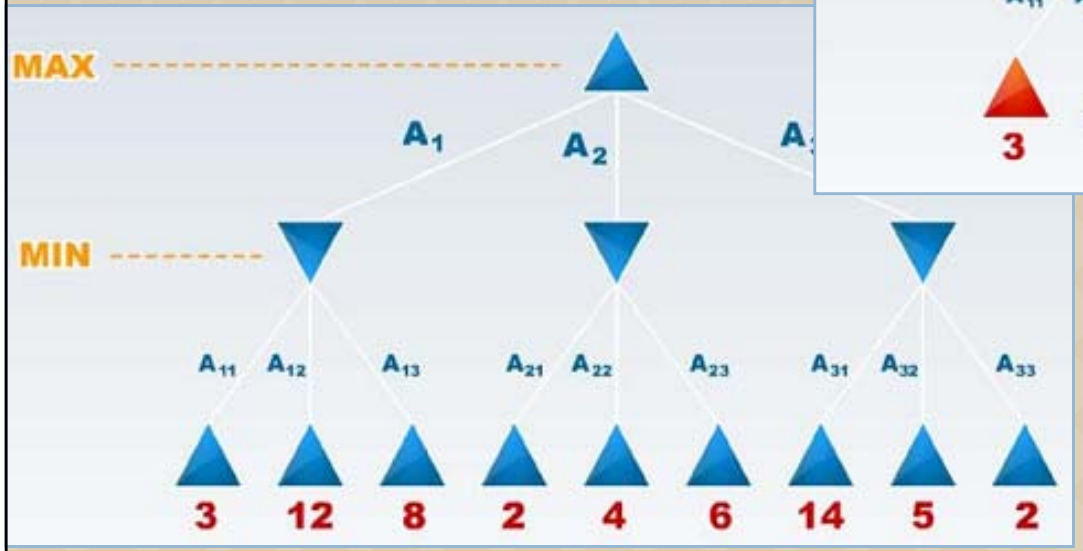
- در این روش، عملیات بازگشت، تمام مسیر را رو به پایین و تا رسیدن به برگهای درخت پیش‌روی می‌کند و سپس مقادیر بیشینه‌کمینه مربوط به هر گره، به صورت معکوس از پایین به بالا اختصاص می‌یابد.

مثال : الگوريتم MiniMax

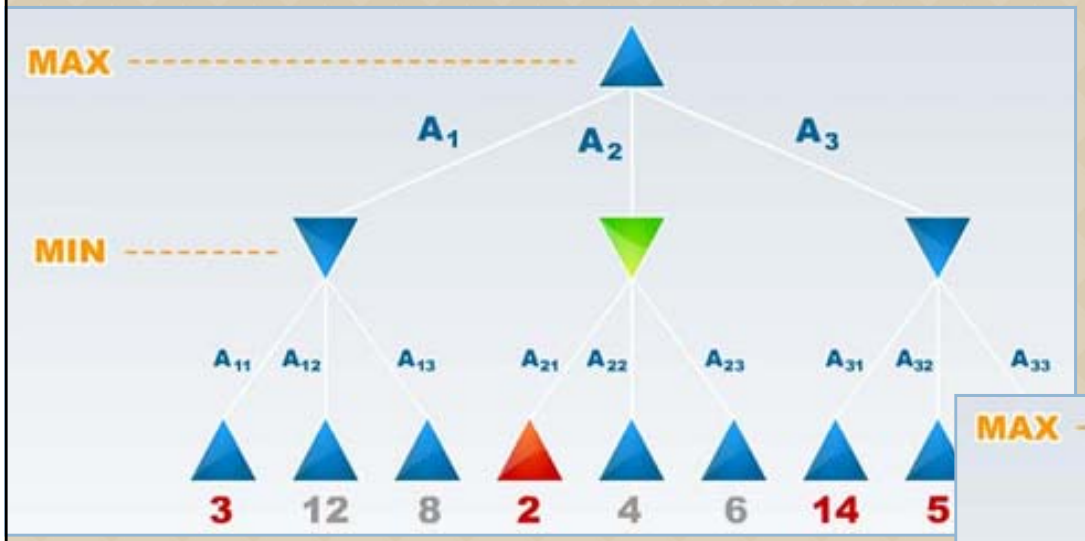
اگر همه فرزندان بررسي شدند مقدار تثبيت مي شود

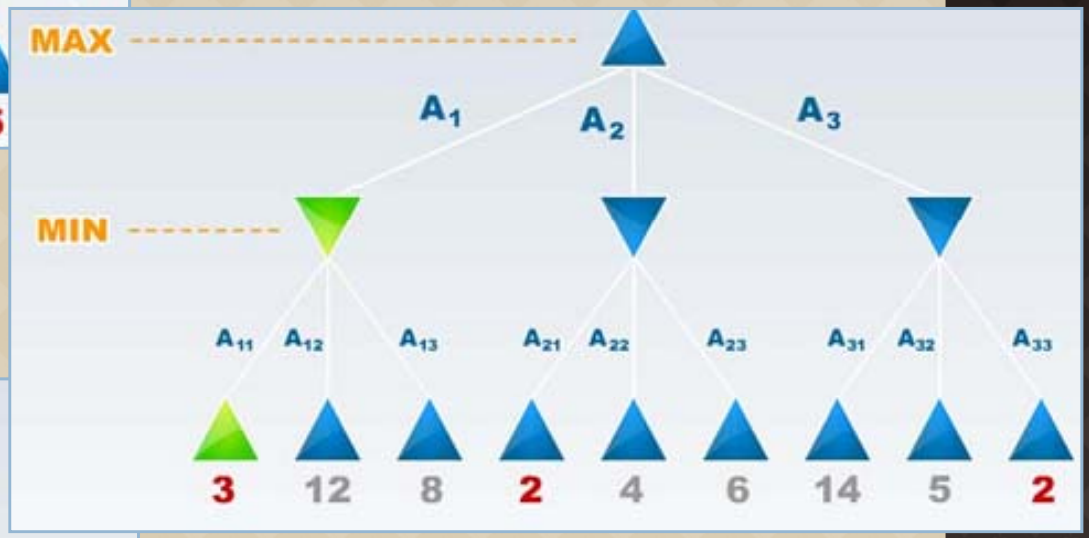


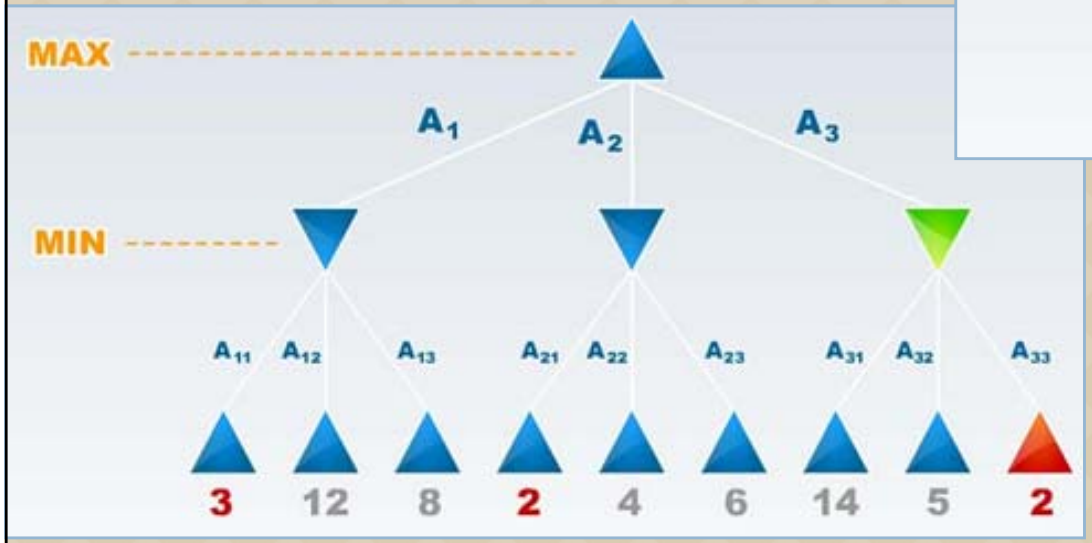




مثال : الگوریتم MiniMax







ارزیابی الگوریتم بیشینه کمینه (MiniMax)

- الگوریتم بیشینه‌کمینه، یک کاوش کامل اوّل عمق را در درخت بازی انجام می‌دهد.

الگوریتم بیشینه کمینه،کامل و بهینه است(بهترین) را انتخاب می کند.

- اگر فرض کنیم حداکثر عمق درخت برابر با m باشد و در هر نقطه بتوان b حرکت مجاز انجام داد، مرتبه پیچیدگی زمانی الگوریتم بیشینه‌کمینه برابر با $O(b^m)$ می‌باشد.

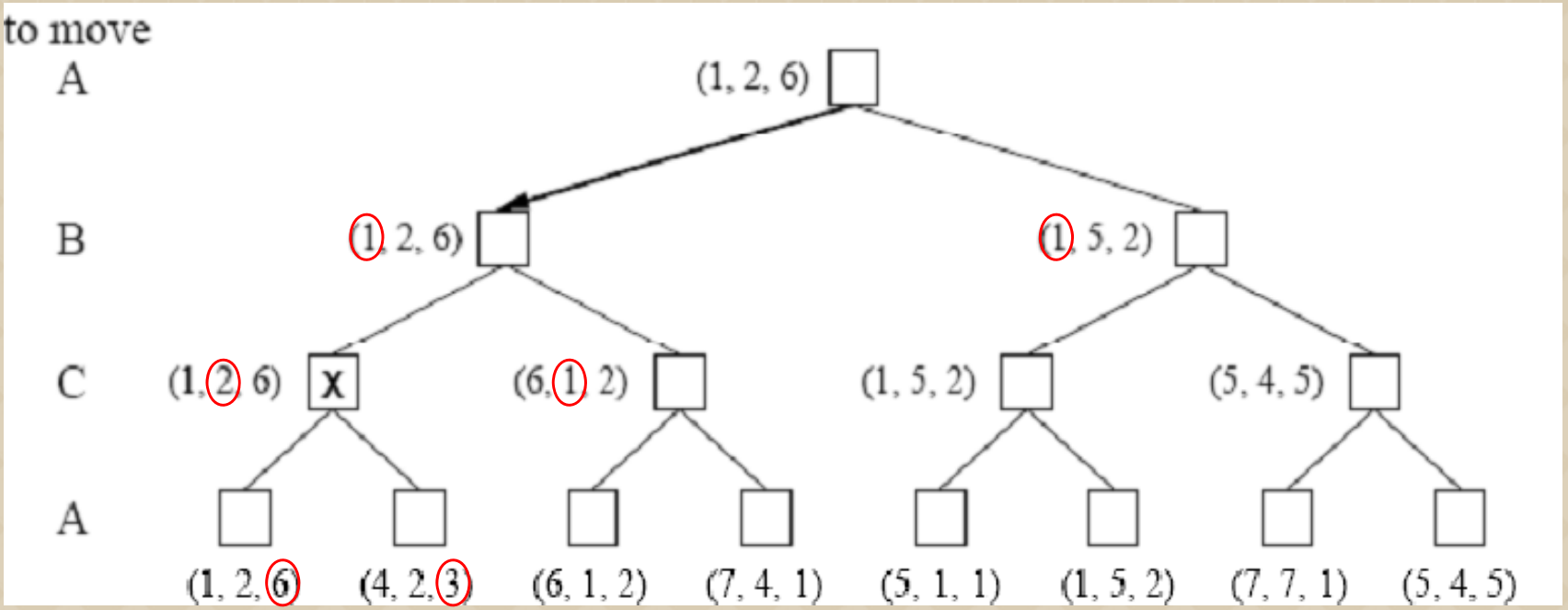
- پیچیدگی فضایی برای الگوریتمی که تمام پسینها را هم‌زمان تولید می‌کند برابر با $O(bm)$ و برای الگوریتمی که در هر لحظه تنها یک پسین را تولید می‌کند برابر با $O(m)$ می‌باشد.

تصمیمات بهینه در بازیهای چند نفره

❖در بسیاری از بازیها بیش از دو بازیکن در یک بازی شرکتد دارند. می خواهیم ببینیم که چگونه می توان ایده بیشینه کمینه را به منظور اعمال به بازیهای چند نفره، بسط داد

❖در ابتدا باید مقادیر اختصاص داده شده به هر گره را با بردارهایی از مقادیر جایگزین کنیم.مثلا در یک بازی سه نفره با بازیکنان A,B,C به هرگره یک بردار $<V_A,V_B,V_C>$ نسبت می دهیم

تصمیمات بهینه در بازیهای چند نفره



تصمیمات بهینه در بازیهای چند نفره(اتحادها)

- در بازیهای چند نفره، اتحادهایی میان بازیکنان رخ می‌دهد. همچنان که بازی ادامه می‌یابد این اتّحادها می‌توانند تشکیل یا شکسته شوند.

- فرض کنید بازیکنان A و B در موقعیت ضعیفتری نسبت به بازیکن C قرار دارند. معمولاً روش بهینه در چنین حالاتی آن است که این دو بازیکن به جای حمله به یکدیگر، به C حمله کنند.

- این اتّحاد تا زمانی پایدار خواهد بود که C موقعیت قویتر خود را از دست بدهد و از آن لحظه به بعد هر کدام از A یا B می‌توانند این اتّحاد را زیر پا بگذارند.

هرس آلفا بتا

❖مشکل عمده در الگوریتم جستجوی بیشینه کمینه آن است که تعداد حالات بازی که در این الگوریتم باید بررسی شوند، دارای یک رابطه نمایی بر حسب تعداد حرکات می باشند.

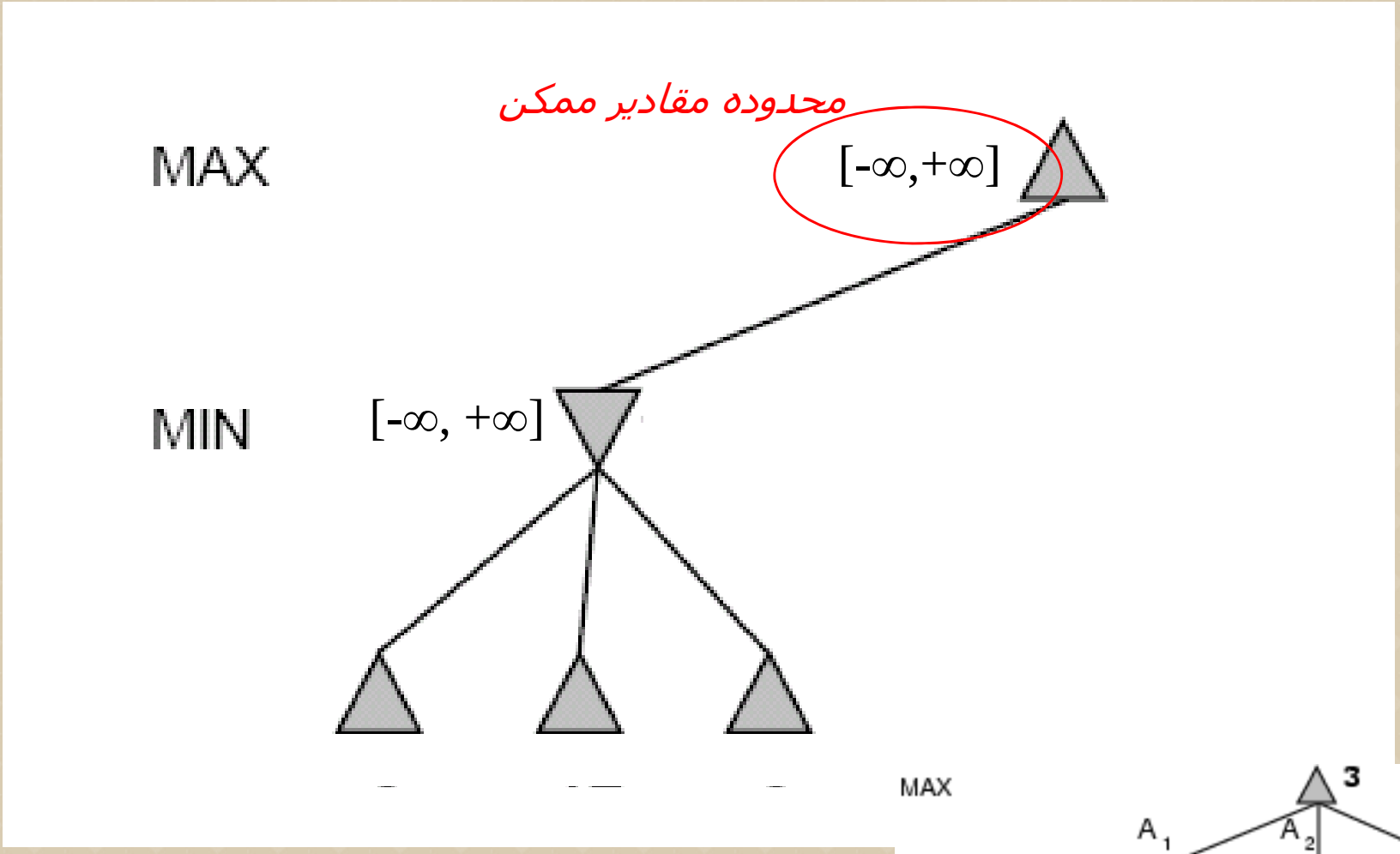
❖می توان با استفاده از یک روش، تعداد حالت بررسی را حتّی به نصف کاهش داد. می توان بدون احتیاج به دیدن تمامی گرههای درخت جستجو، بیشینه کمینه صحیح را محاسبه نمود.

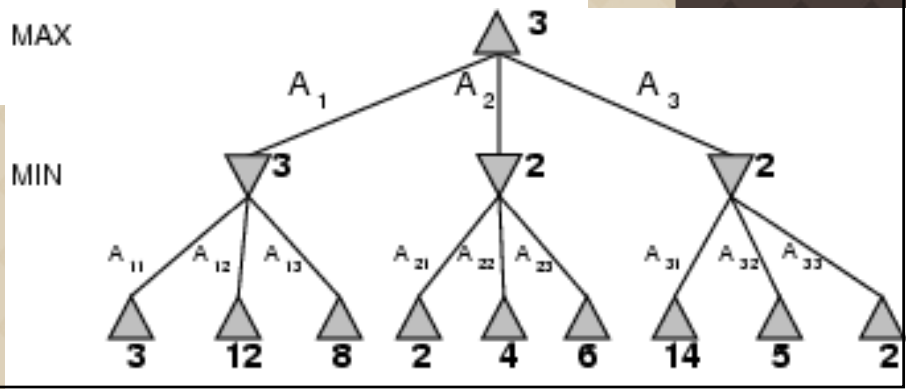
هرس آلفا بتا

❖بدین منظور با استفاده از ایده هرس کردن، بخشهای بزرگی از درخت، از نظر محو می شود.هرس کردن باعث می شود آن قسمتی از درخت که هیچ تاثیری در نتیجه نهایی ندارد بررسی نشود

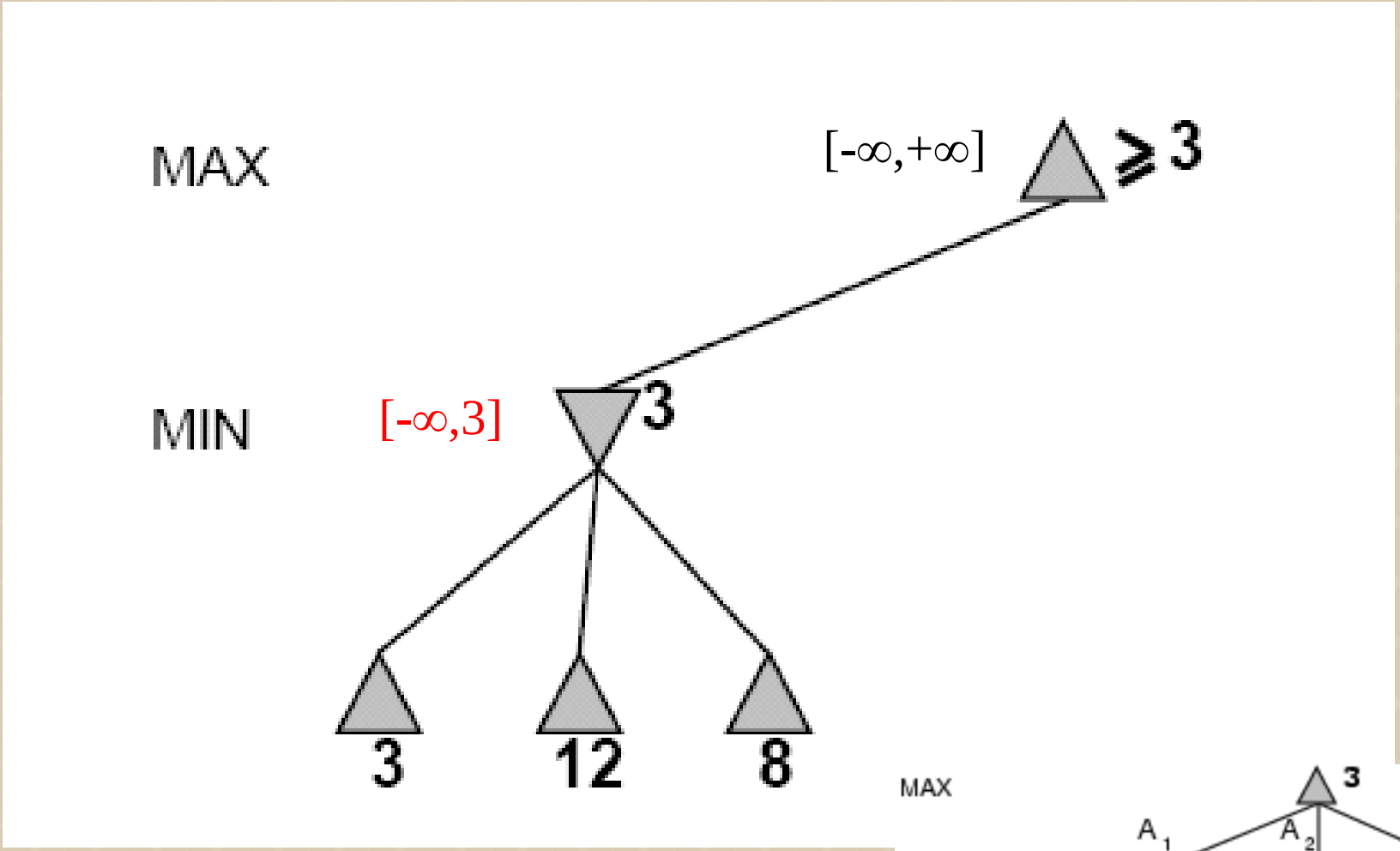
❖ هرس آلفابتا هنگامی که این الگوریتم به یک درخت بیشینه کمینه استاندارد اعمال می شود، همان حرکات قبلی را مشابه الگوریتم بیشینه کمینه، برمی گرداند با این تفاوت که در این روش، شاخه هایی که تأثیری در تصمیم گیری نهایی ندارند،هرس می شوند.

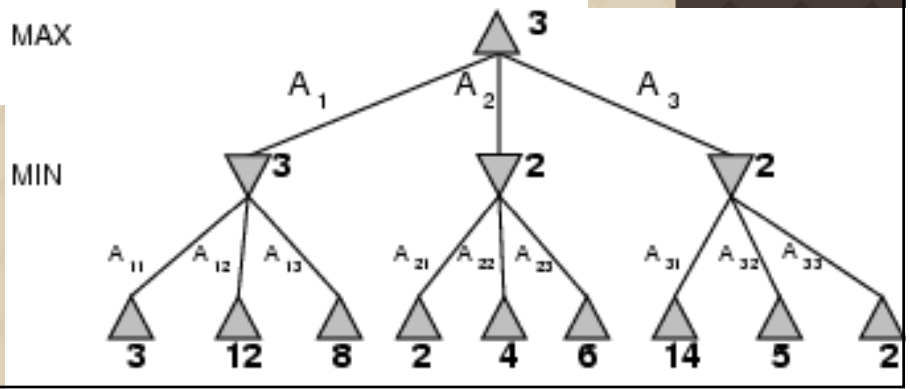
مثال هرس آلفا بتا



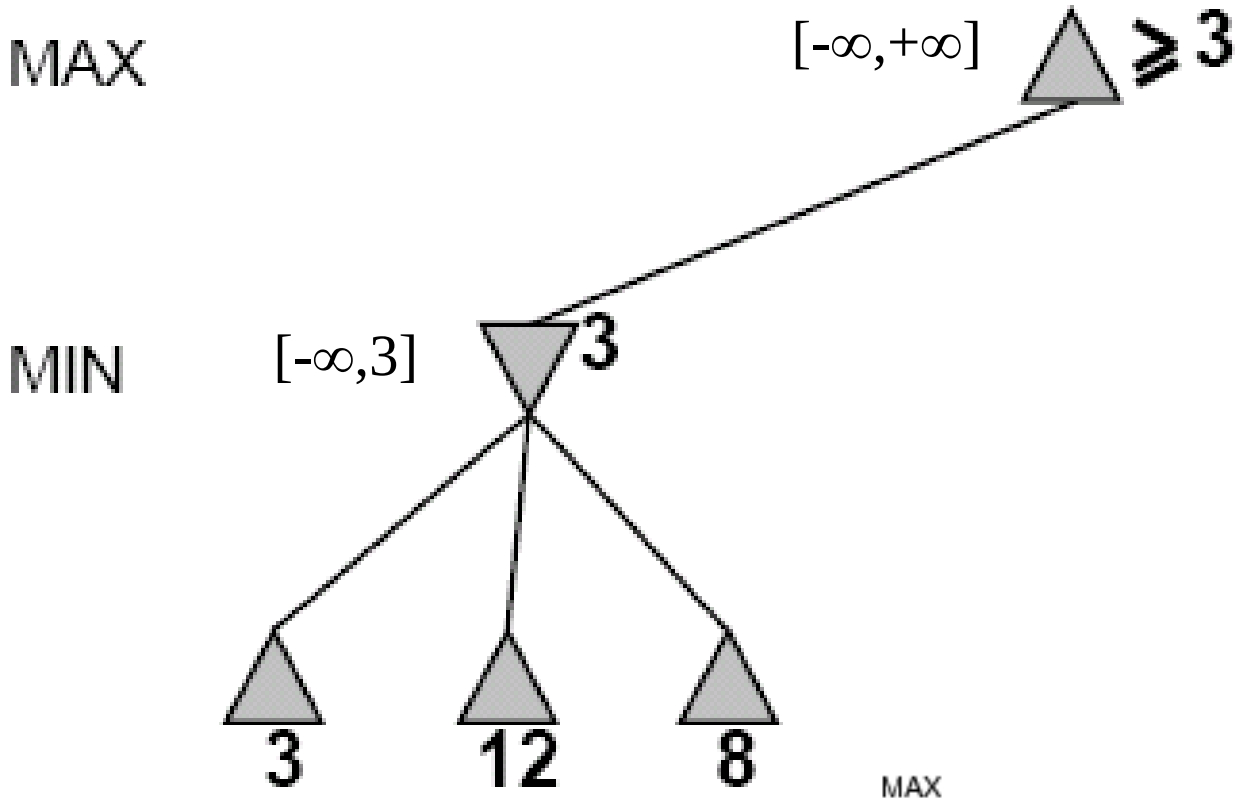


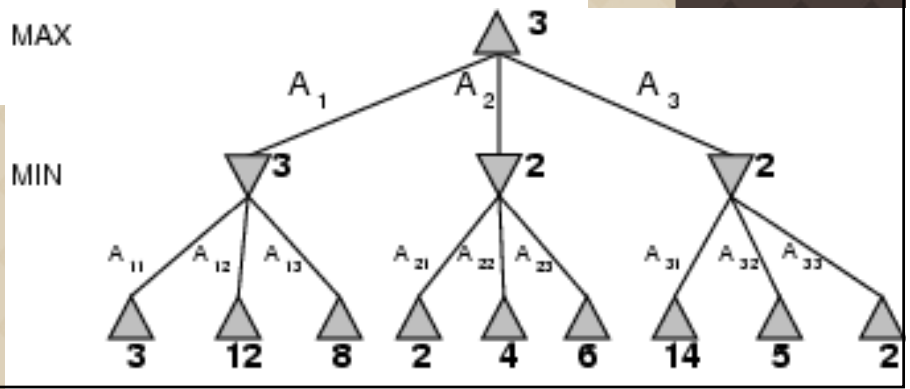
مثال هرس آلفا بتا



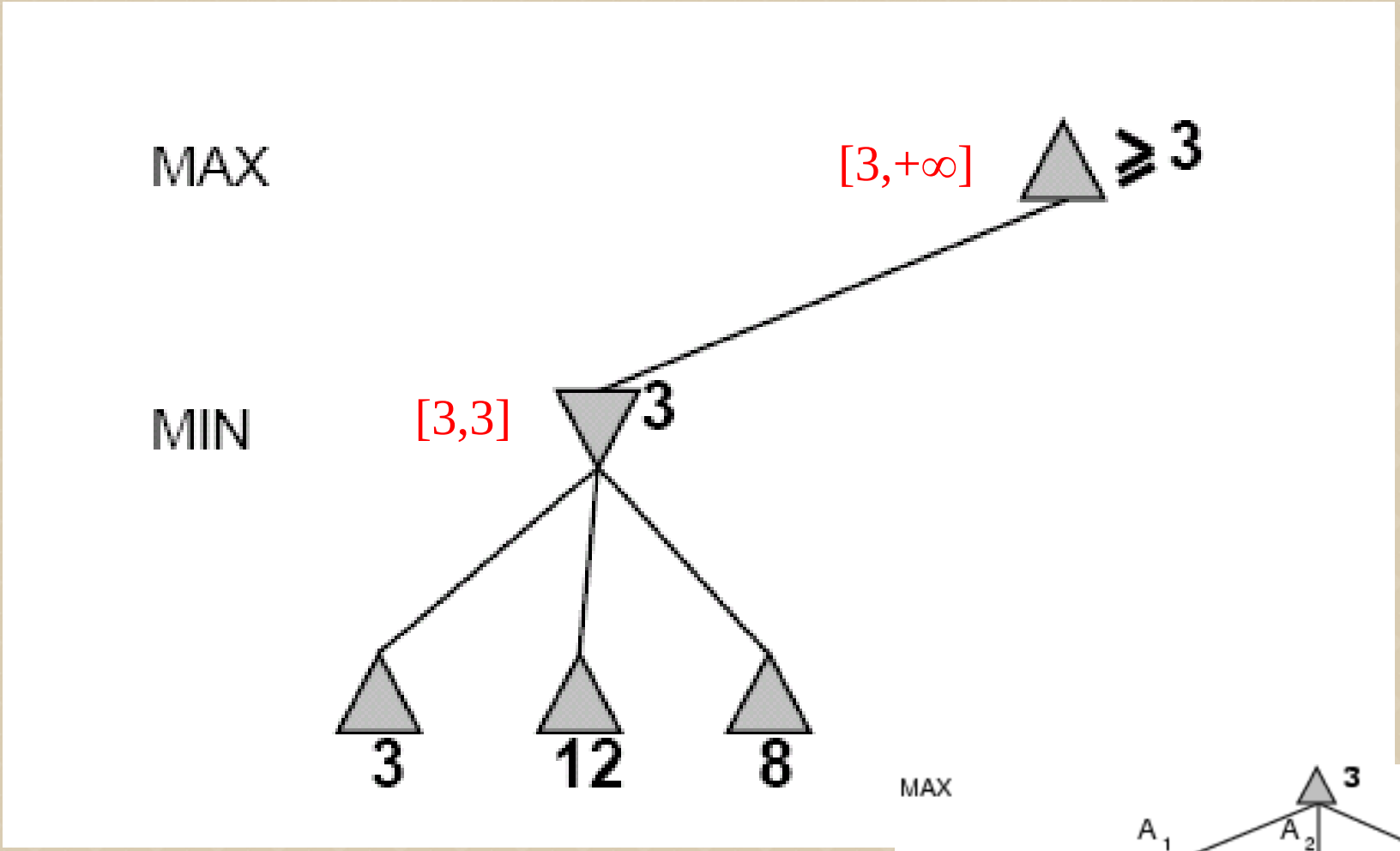


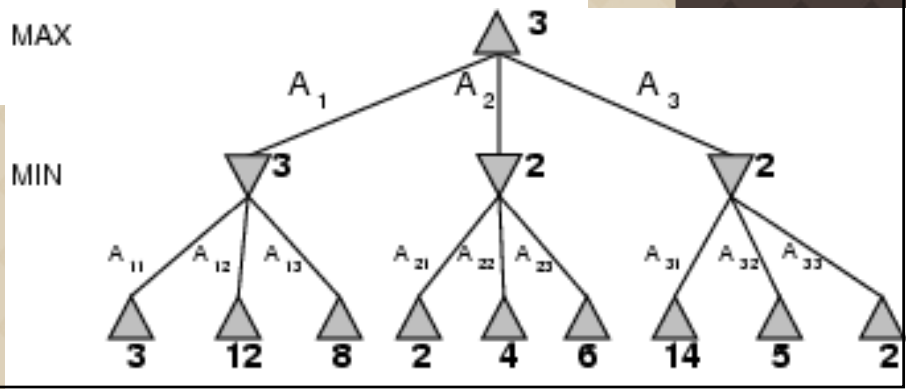
مثال هرس آلفا بتا



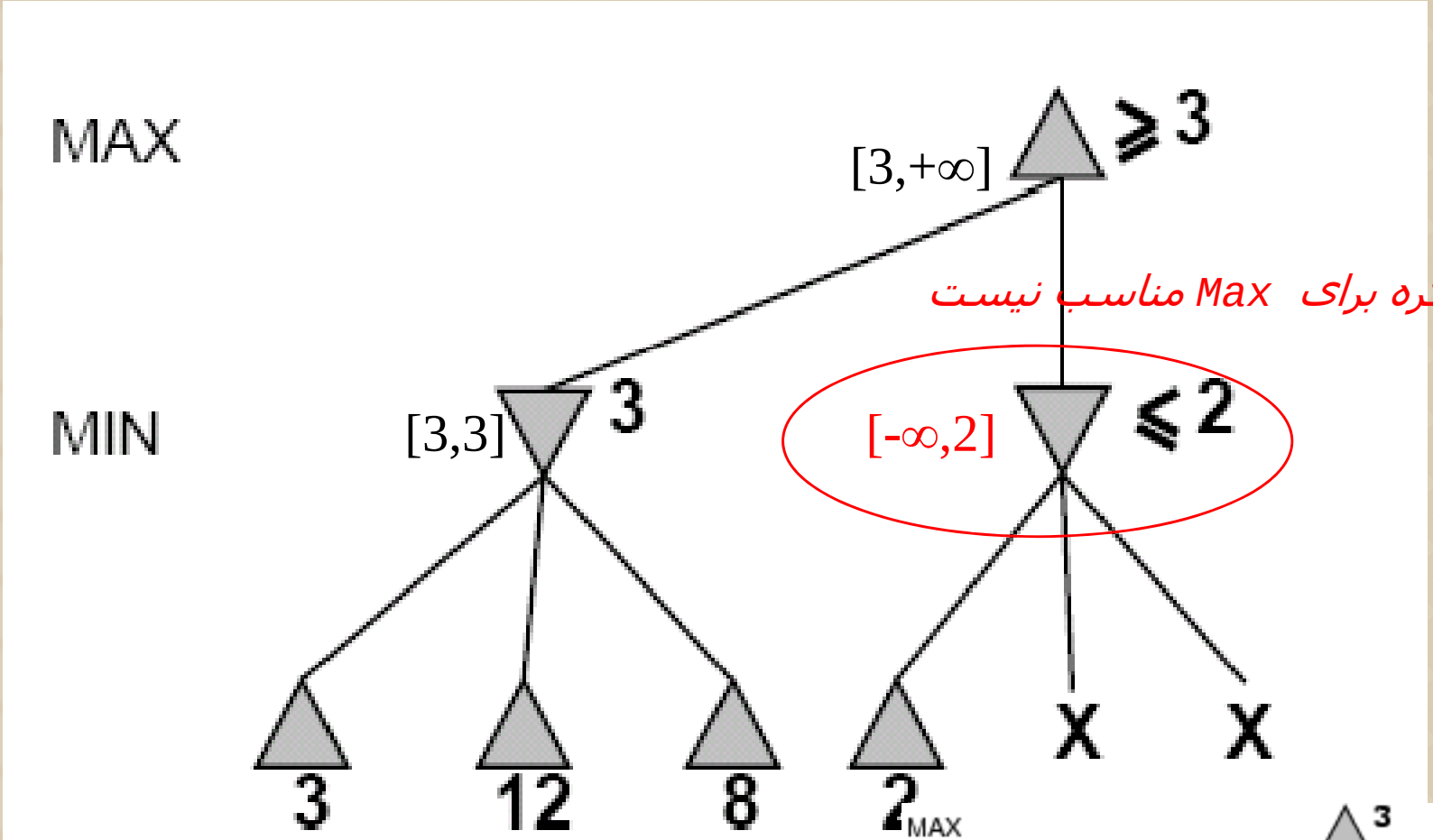


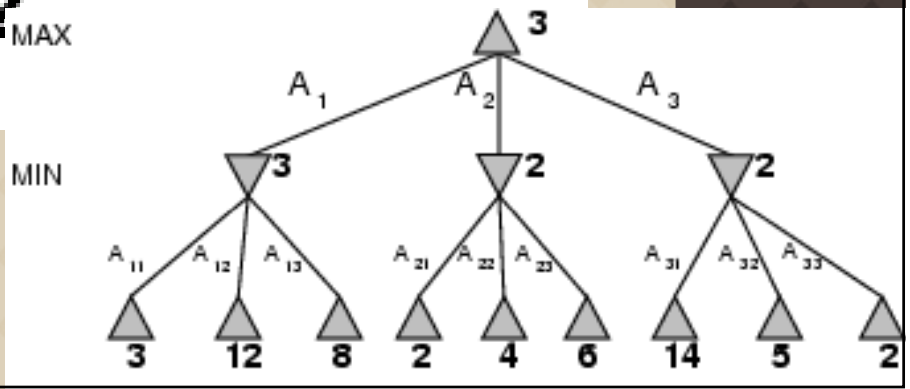
مثال هرس آلفا بتا



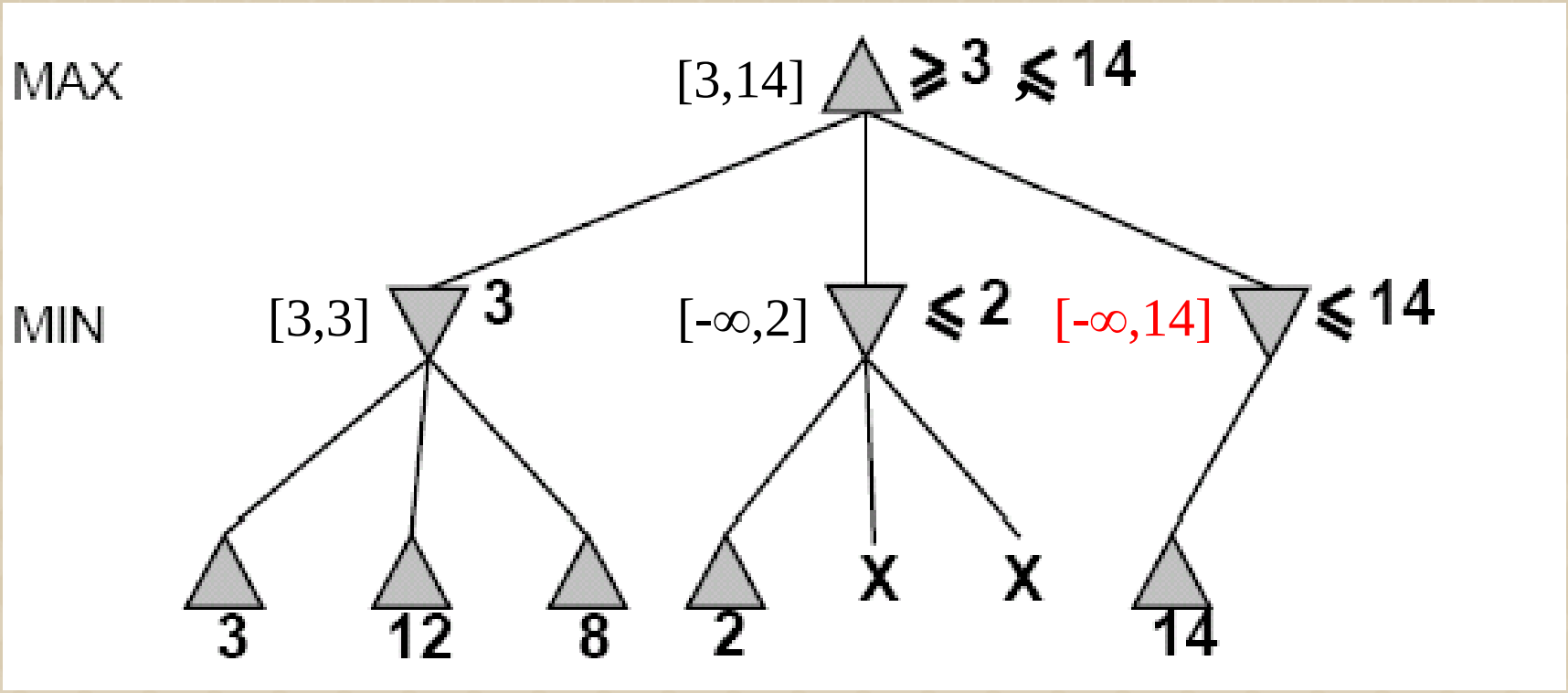


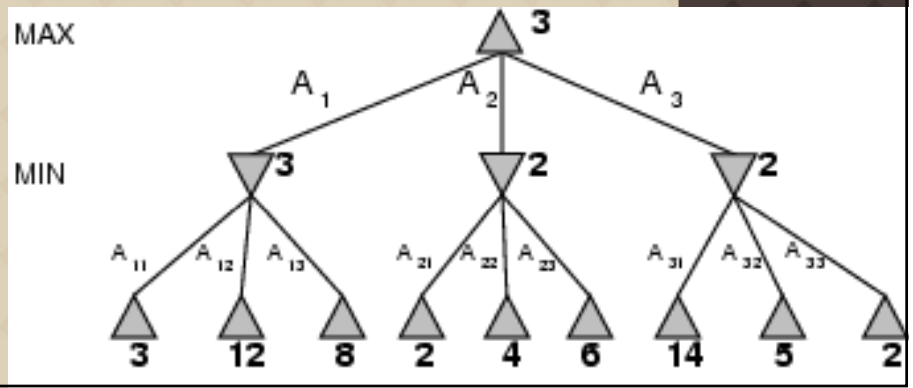
مثال هرس آلفا بتا



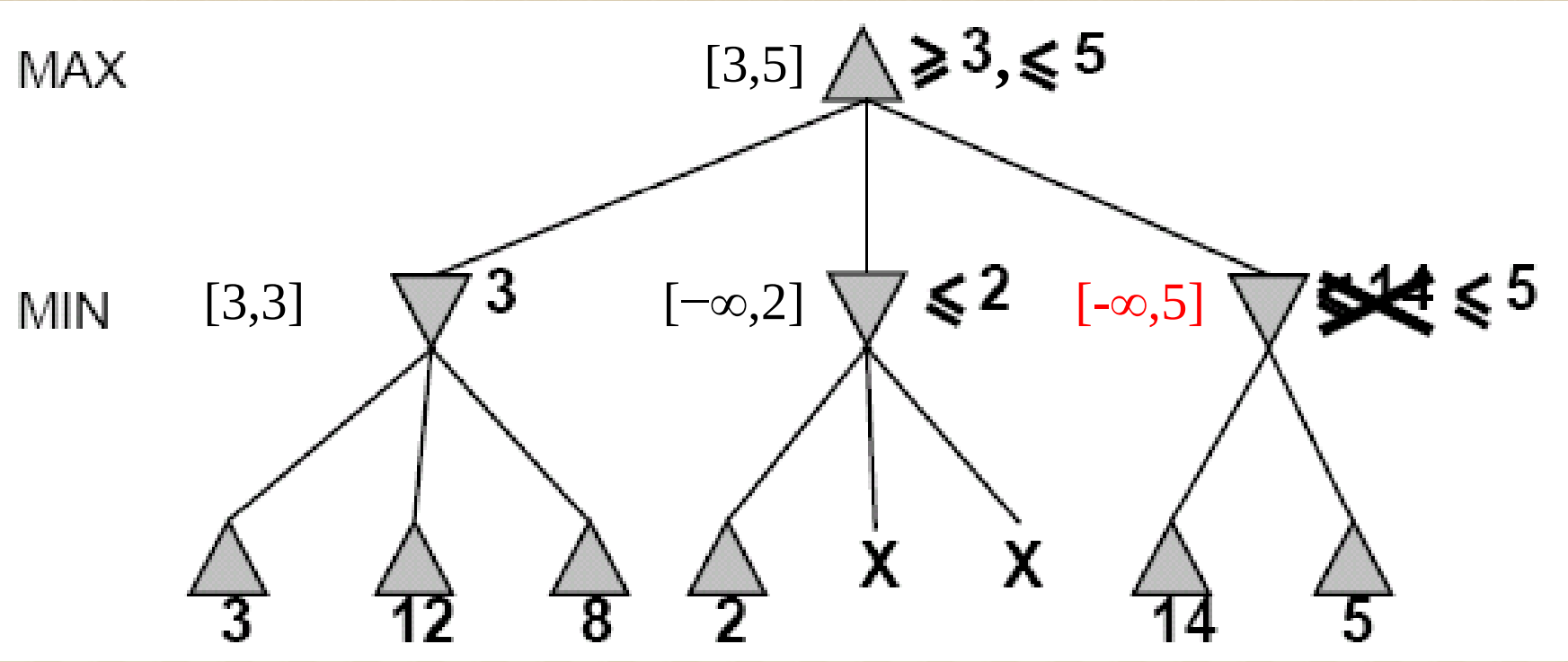


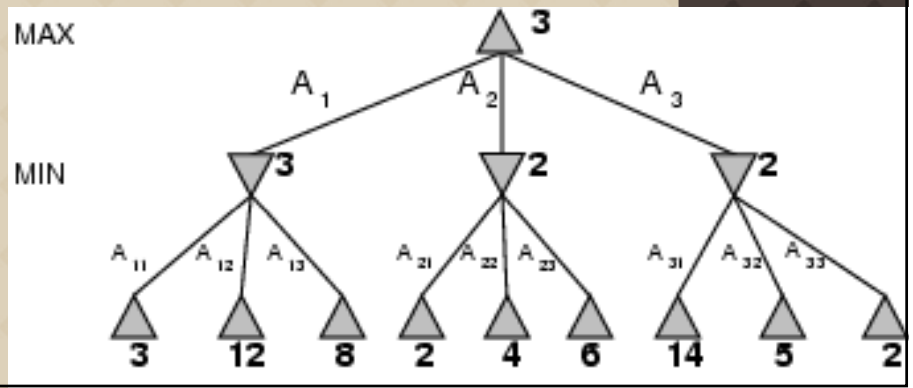
مثال هرس آلفا بتا



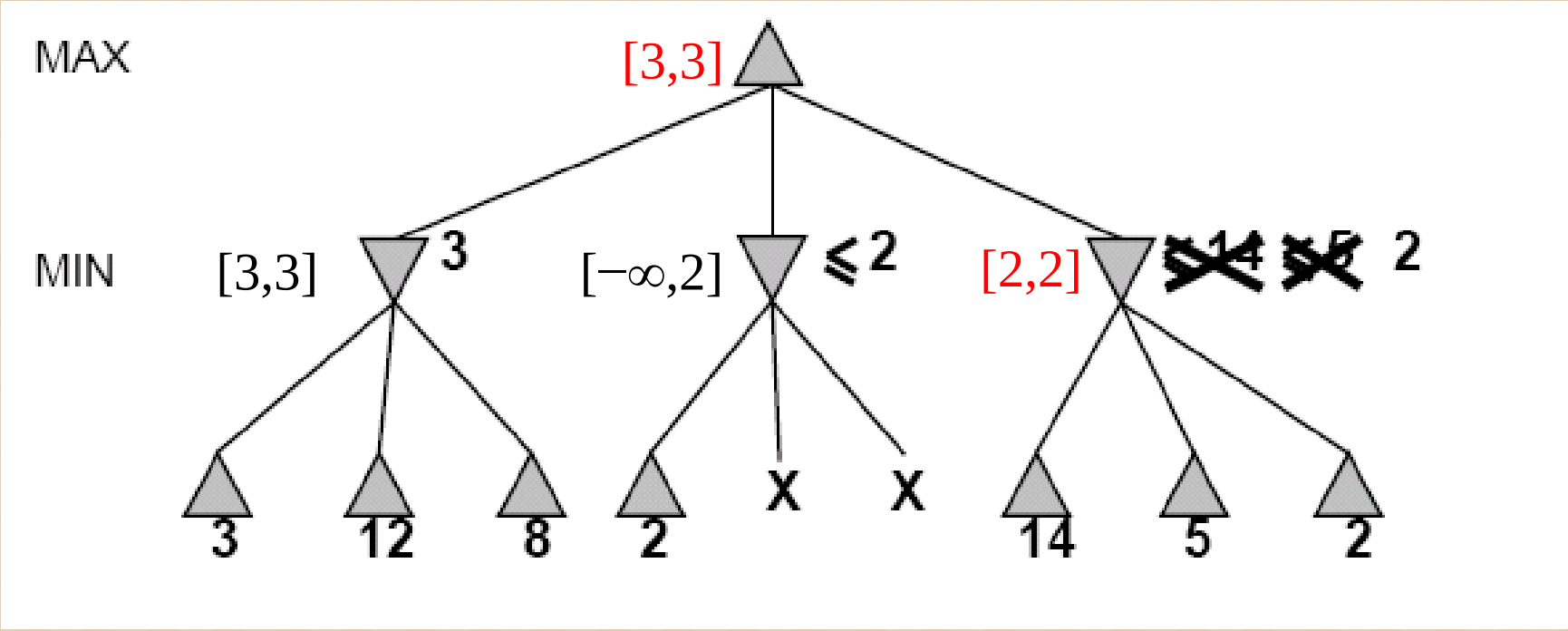


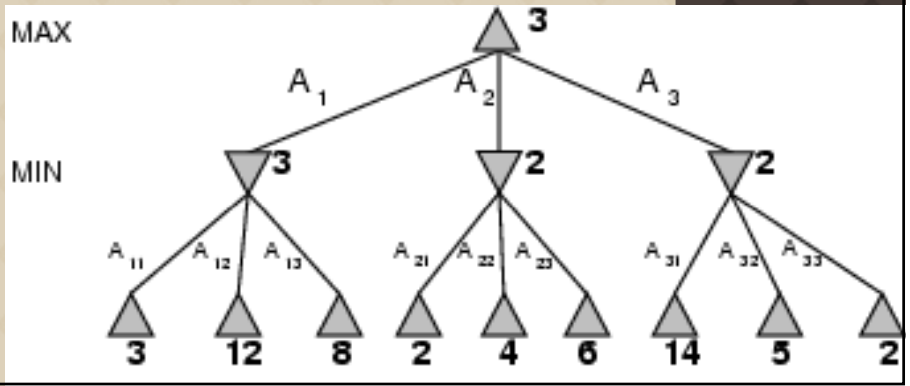
مثال هرس آلفا بتا



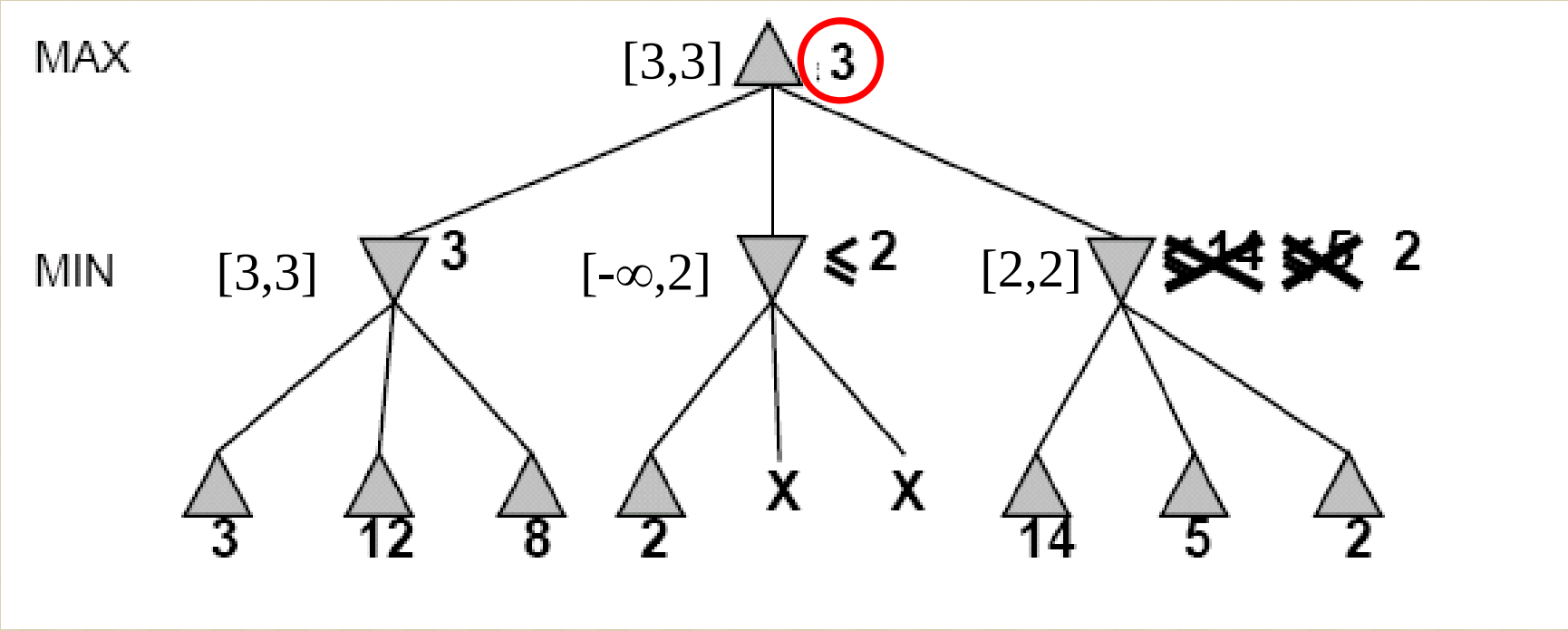


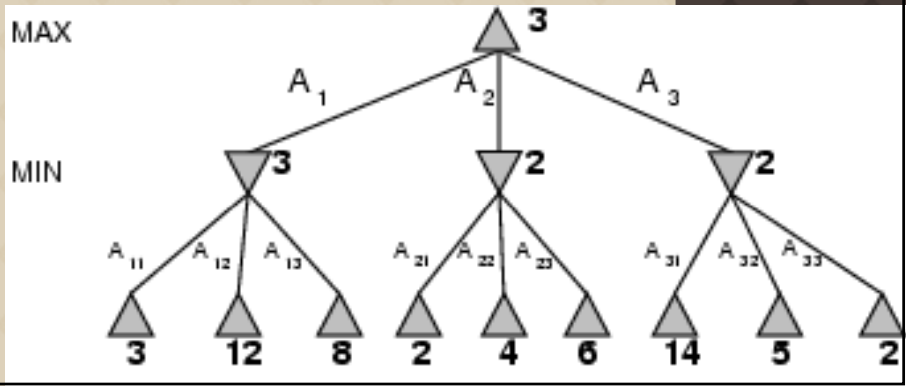
مثال هرس آلفا بتا

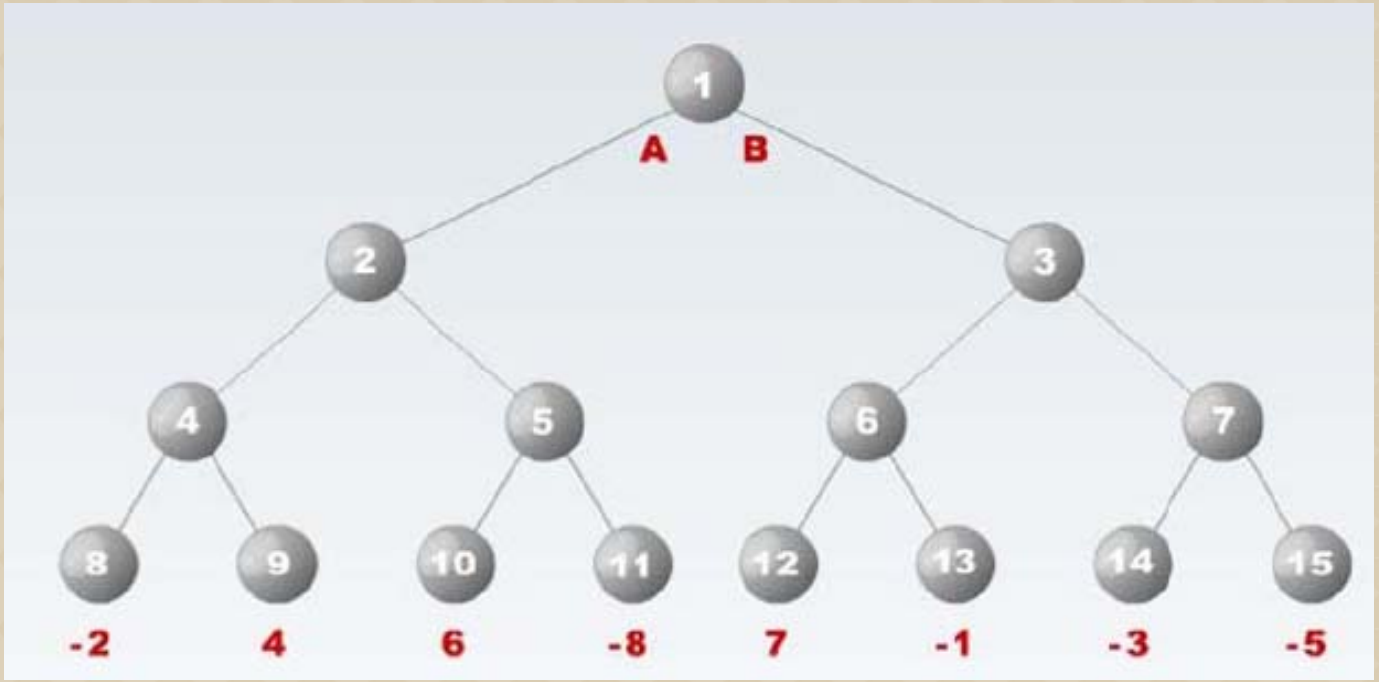


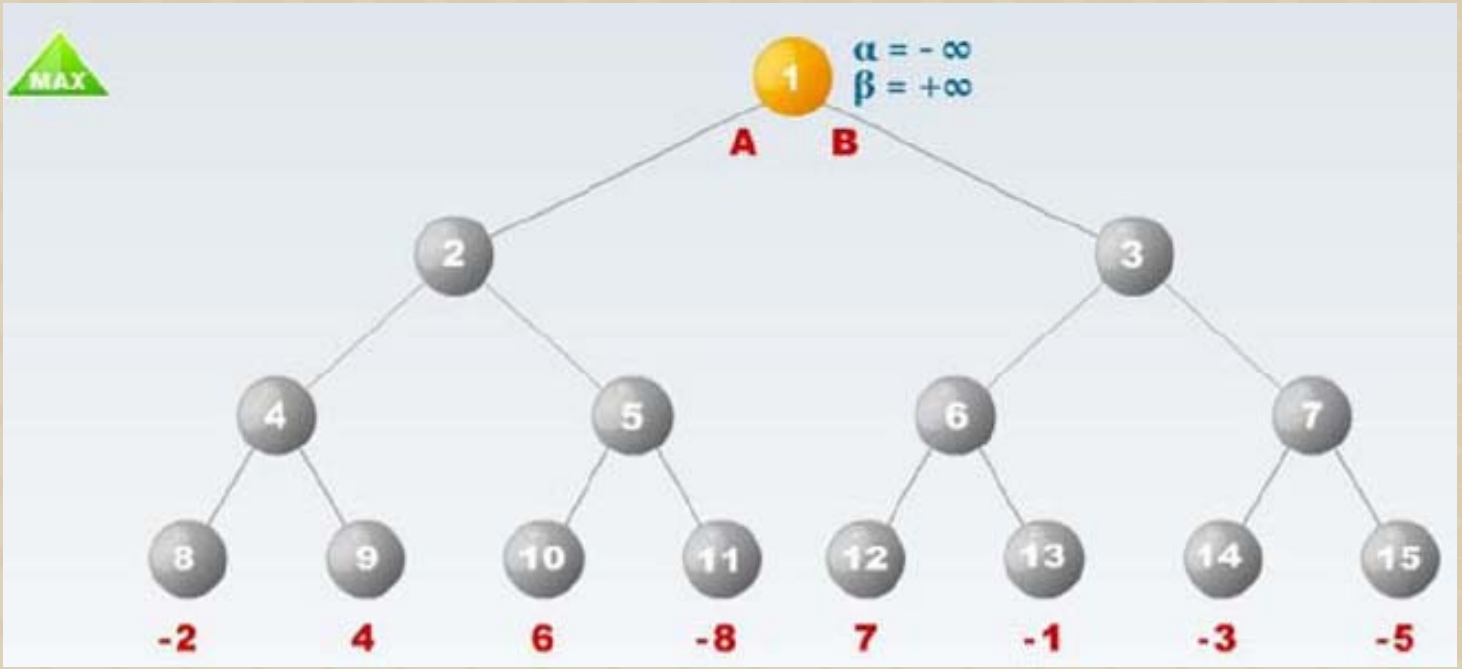


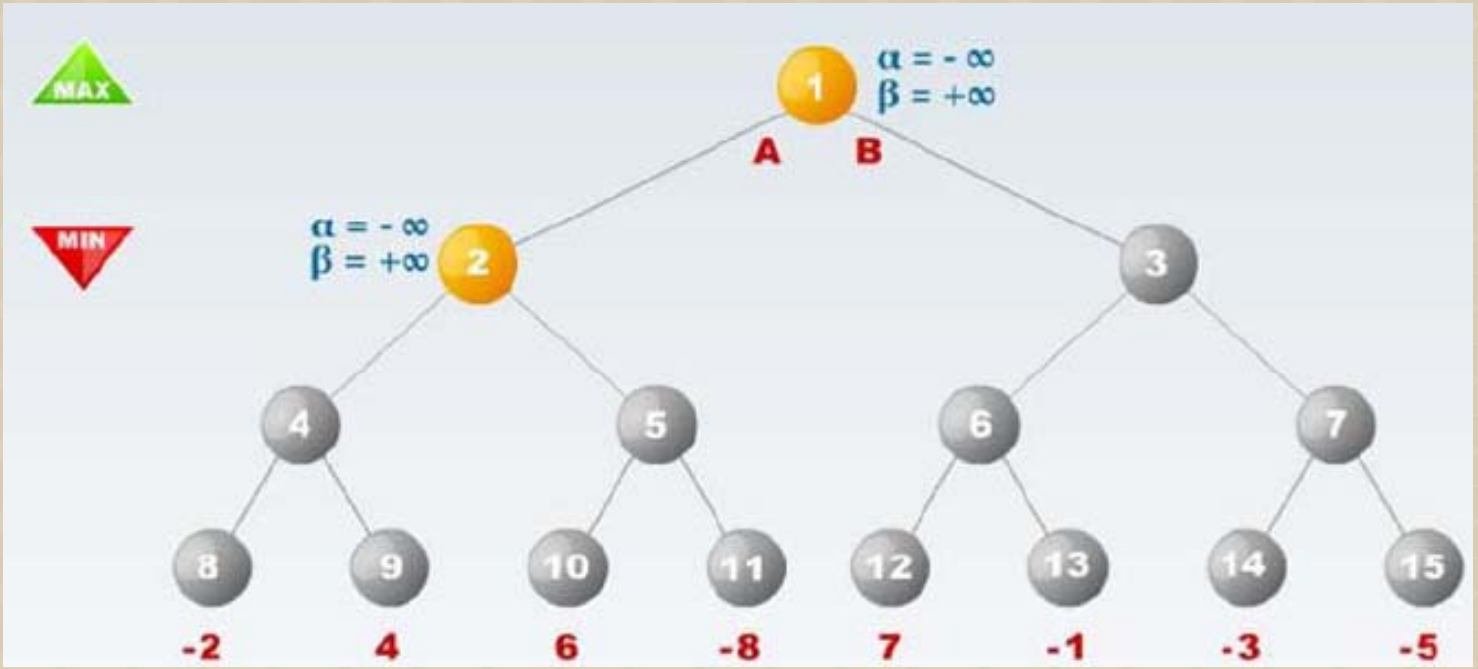
مثال هرس آلفا بتا

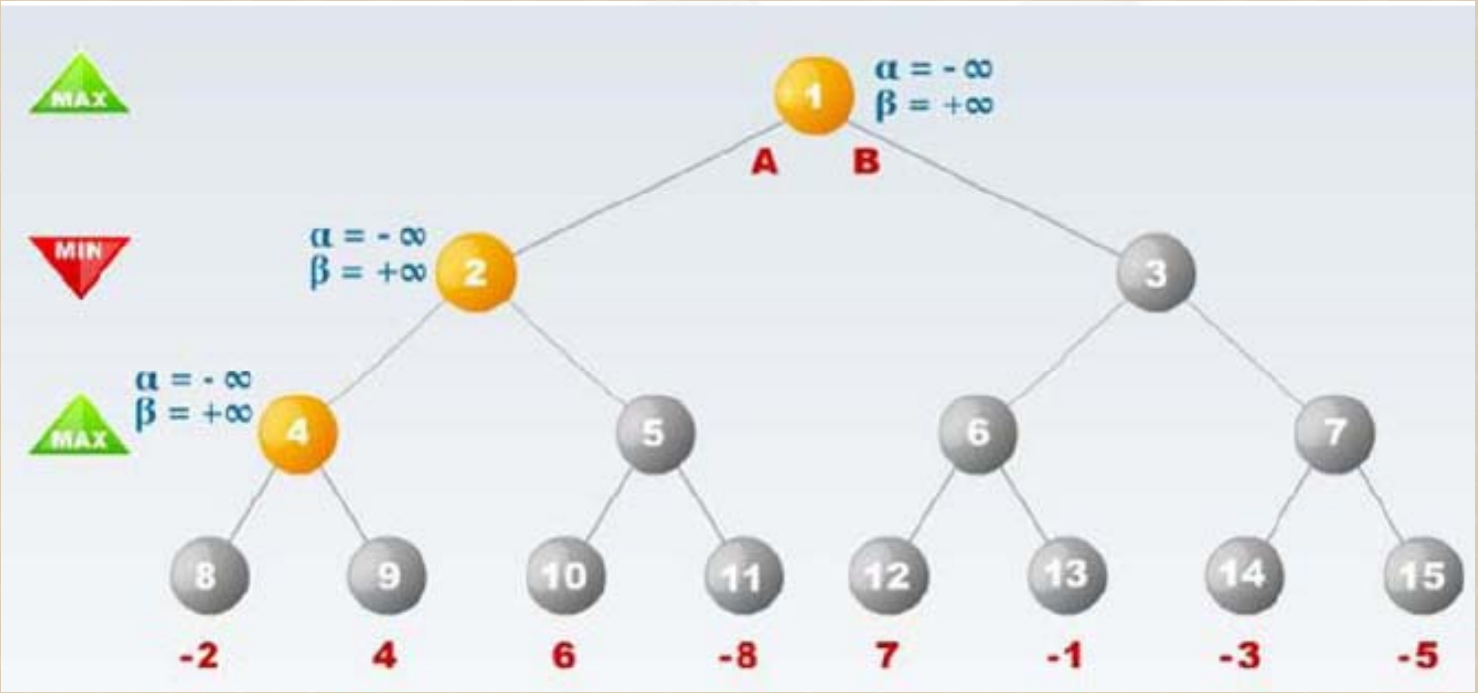


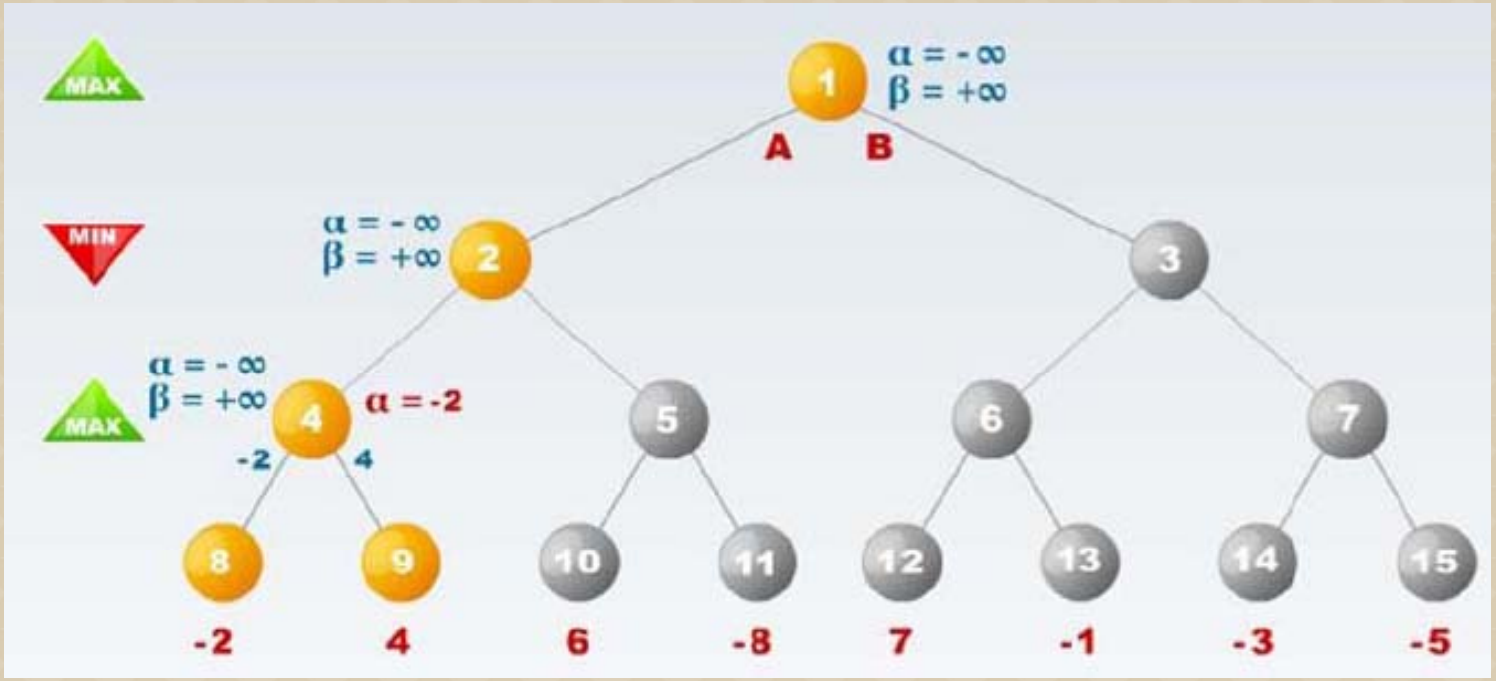


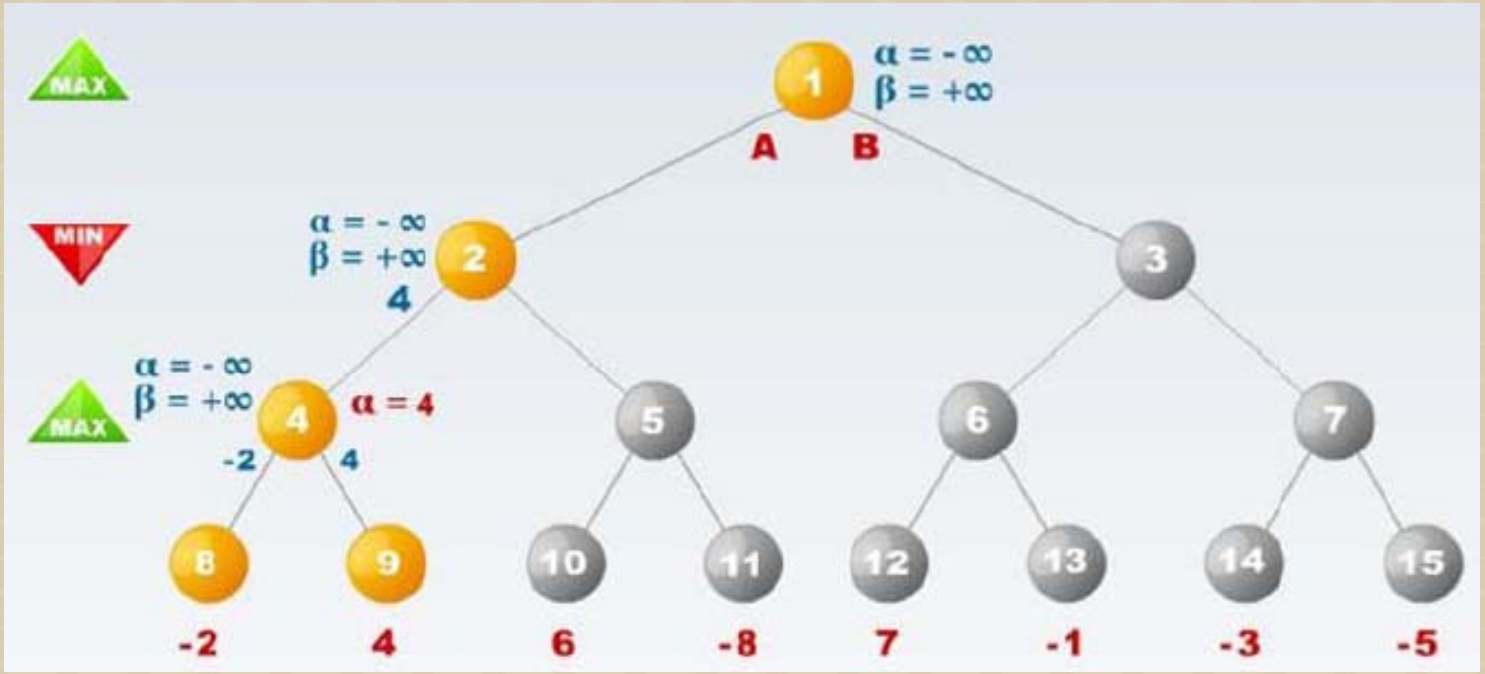

1
A
B
2
3
4
5
6
7
8
9
10
11
12
13
14
15
-2
4
6
-8
7
-1
-3
-5

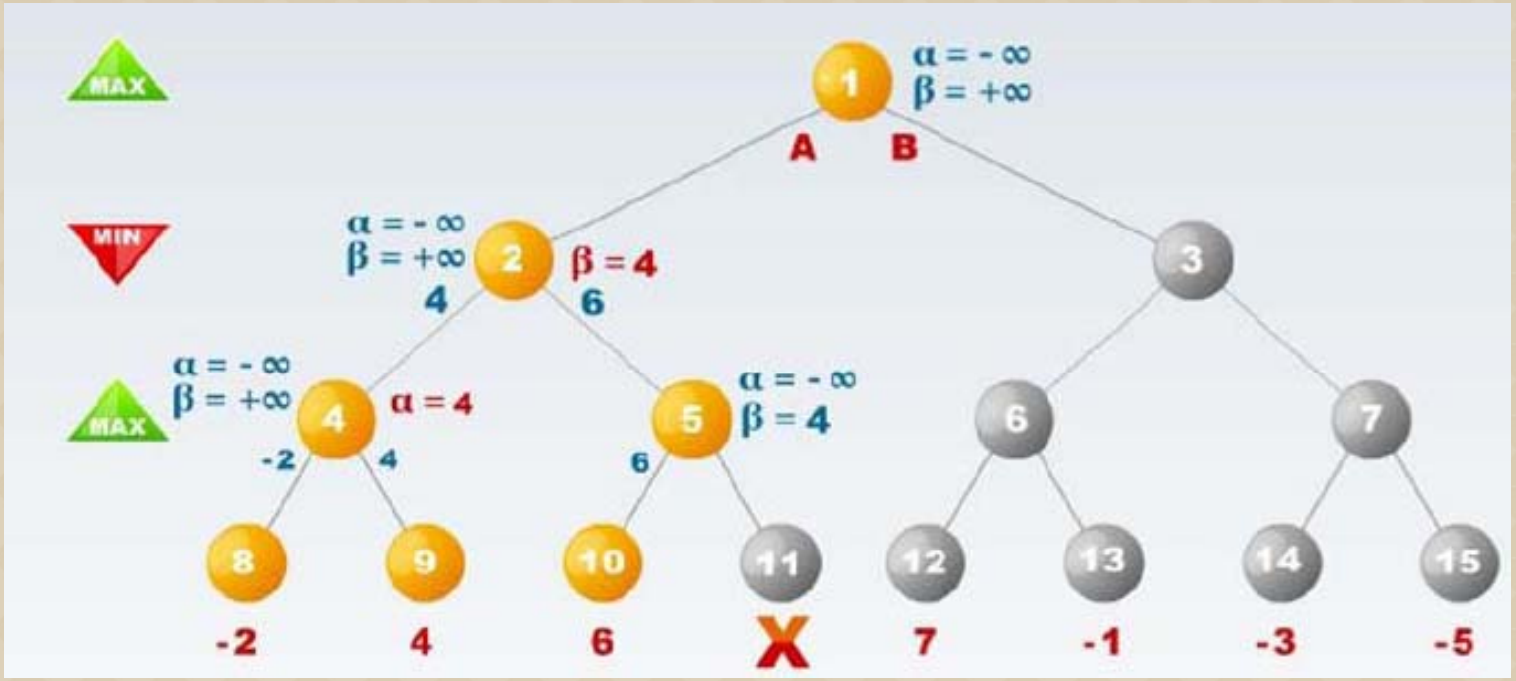

MAX
1
α = - ∞
β = +∞
A
B
2
3
4
5
6
7
8
9
10
11
12
13
14
15
-2
4
6
-8
7
-1
-3
-5


MAX
MIN
1
α = - ∞
β = +∞
A
B
2
α = - ∞
β = +∞
3
4
5
6
7
8
9
10
11
12
13
14
15
-2
4
6
-8
7
-1
-3
-5


MAX
MIN
MAX
α = - ∞
β = +∞
1
A
B
α = - ∞
β = +∞
2
3
α = - ∞
β = +∞
4
5
6
7
8
9
10
11
12
13
14
15
-2
4
6
-8
7
-1
-3
-5


MAX
MIN
MAX
1
α = - ∞
β = +∞
A
B
2
α = - ∞
β = +∞
3
4
α = - ∞
β = +∞
α = -2
-2
4
5
6
7
8
9
10
11
12
13
14
15
-2
4
6
-8
7
-1
-3
-5


MAX
MIN
MAX
1
α = - ∞
β = +∞
A
B
2
α = - ∞
β = +∞
4
3
4
α = - ∞
β = +∞
α = 4
-2
4
5
6
7
8
9
10
11
12
13
14
15
-2
4
6
-8
7
-1
-3
-5


MAX
MIN
MAX
1
α = - ∞
β = +∞
A
B
2
α = - ∞
β = +∞
4
β = 4
6
3
4
α = - ∞
β = +∞
-2
α = 4
4
5
α = - ∞
β = 4
6
6
7
8
9
10
11
12
13
14
15
-2
4
6
X
7
-1
-3
-5

هرس آلفا بتا

الگوریتم هرس آلفابتا را می توان به درختهایی با هر میـزان عمـق، اعمال نمود و در برخی موارد می توان بـه جـای فقـط برگهـا، زیـر شاخه های اصلی را نیز هرس نمود.

- راه حل کلی بدین شکل است:
  - گره n را در جایی از درخت در نظر بگیرید به ترتیبی که بازیکن بتواند به عنوان یکی از انتخابها به آن حرکت کند.
  - اگر بازیکن دارای انتخاب بهتری مانند m در شاخه والد یا هر شاخه دیگر قبل از n باشد، آنگاه *در یک بازی حقیقی هرگز به گره n نخواهیم رسید.*
  - بنابراین زمانی که اطلاعات کافی در مورد گره n (با بررسی حالات پس از آن) برای رسیدن به این نتیجه کسب شد، می‌توان آن را هرس کرد.

هرس آلفا بتا

جستجوی بیشینه کمینه یک جستجوی اول عمق می باشد، لذا در هر زمان باید تنها گرههایی که در راستای یک مسیر از درخت باشند، مورد توجه قرار گیرند.

- هرس آلفابتا، نام خود را از دو پارامتر آلفا (α) و بتا (β) گرفته است که مشخص کننده کران مقادیر نگهداری شده در تمام طول مسیر هستند:
  - α : مقدار بهترین انتخاب ممکن (یعنی بالاترین مقدار) در هر نقطه انتخاب در طول مسیر برای MAX که تا کنون به دست آمده است.
  - β: مقدار بهترین انتخاب ممکن (یعنی کمترین مقدار) در هر نقطه انتخاب در طول مسیر برای MIN که تا کنون به دست آمده است.

هرس آلفا بتا

- در جستجوی آلفابتا، هم‌زمان با پیش‌روی در درخت بازی مقادیر α و β روزآوری می‌شوند.

- به محض این که مقدار گره جاری، به ترتیب بدتر از مقادیر فعلی α و β برای MIN و MAX تشخیص داده شود، شاخه‌های باقی‌مانده در آن گره هرس می‌شوند.

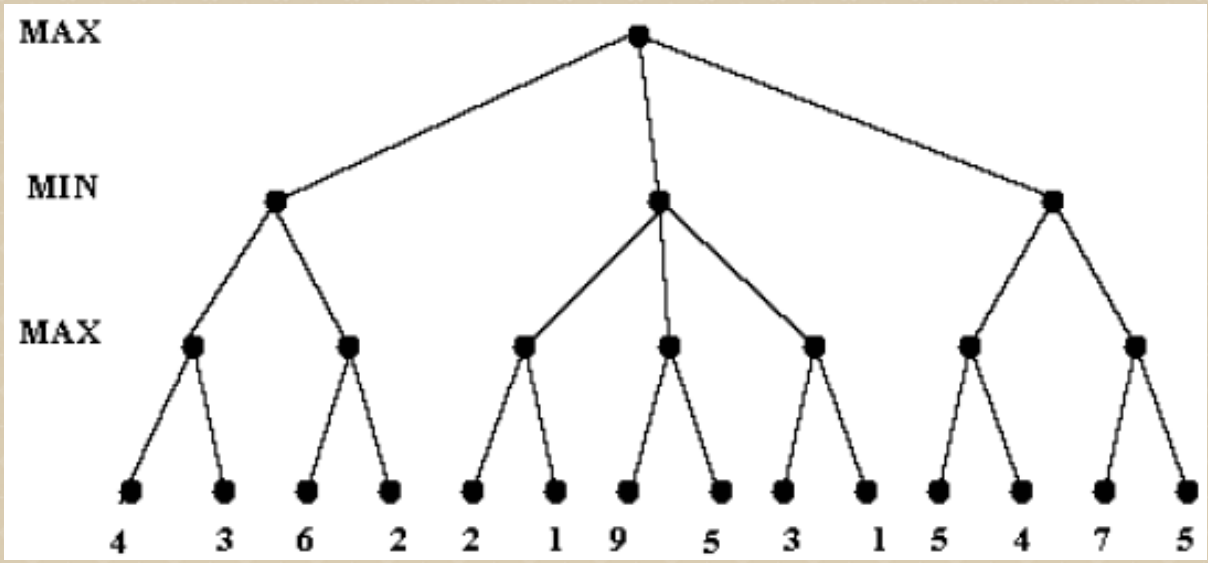
کارآیی هرس آلفابتا تا حد زیادی به ترتیب بررسی پسینها وابسته است.
بهتر است در ابتدا پسینهایی که احتمال دارد بهترین باشند،امتحان شوند.

ارزیابی هرس آلفا بتا

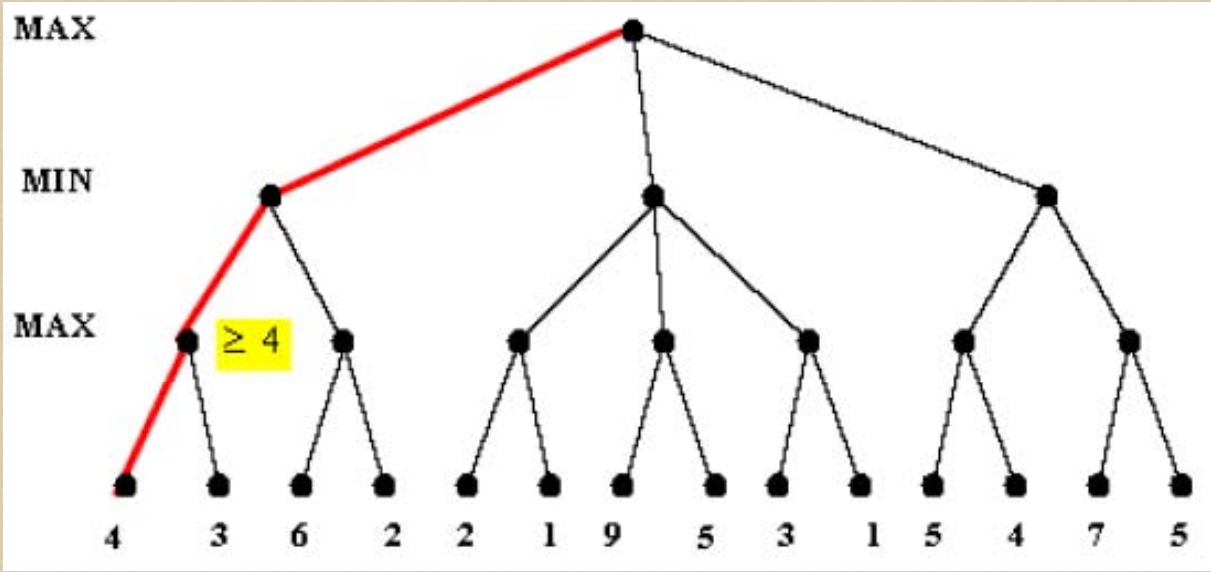
- روش آلفابتا به جای بررسی $O(b^d)$ گره، مانند روش بیشینه‌کمینه، تنها می‌تواند با $O(b^{d/2})$ بررسی، بهترین حرکت را انتخاب نماید.

- روش آلفابتا در زمانی برابر، می‌تواند تا دو برابر روش بیشینه‌کمینه، پیش‌بینی کند. اگر پسینها به جای انتخاب بهترین به صورت تصادفی انتخاب شوند، تعداد گرههایی که برای رسیدن به یک b مناسب بررسی خواهد شد، در حدود $O(b^{3d/4})$ خواهد بود.
- در شطرنج، یک عملکرد نسبتاً ساده ترتیبی (مانند: اوّل تلاش برای زدن مهره حریف، دوّم تهدید مهرهٔ حریف، سوّم حرکت پیش‌رو و چهارم حرکت رو به عقب) تقریباً دو برابر حالات مورد بررسی در بهترین حالت یعنی $O(b^{d/2})$ را نتیجه می‌دهد.

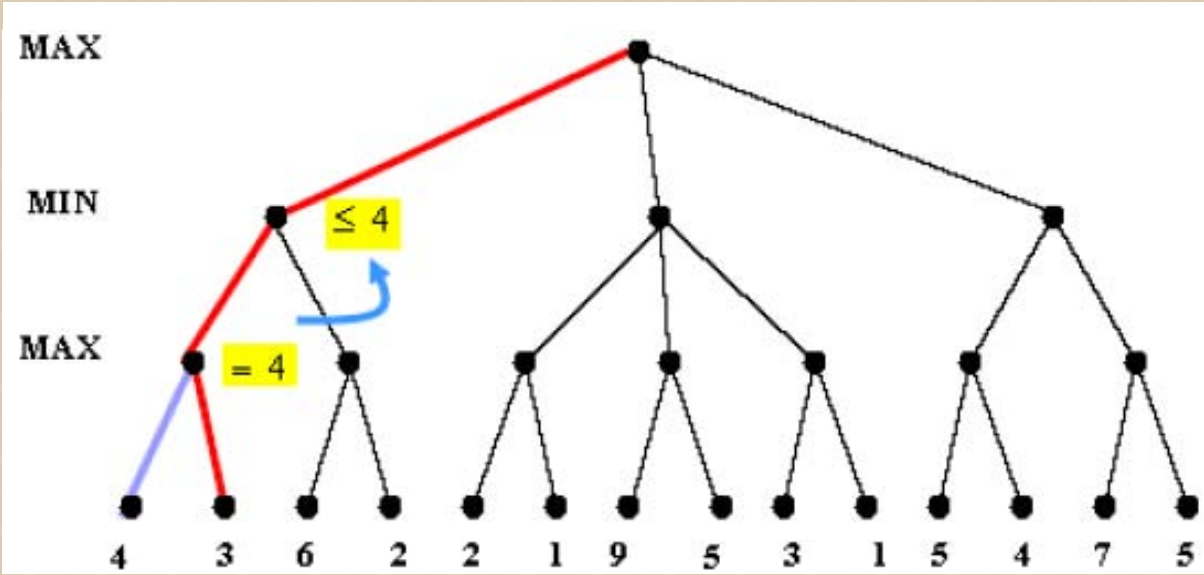
مثال : هرس آلفا بتا



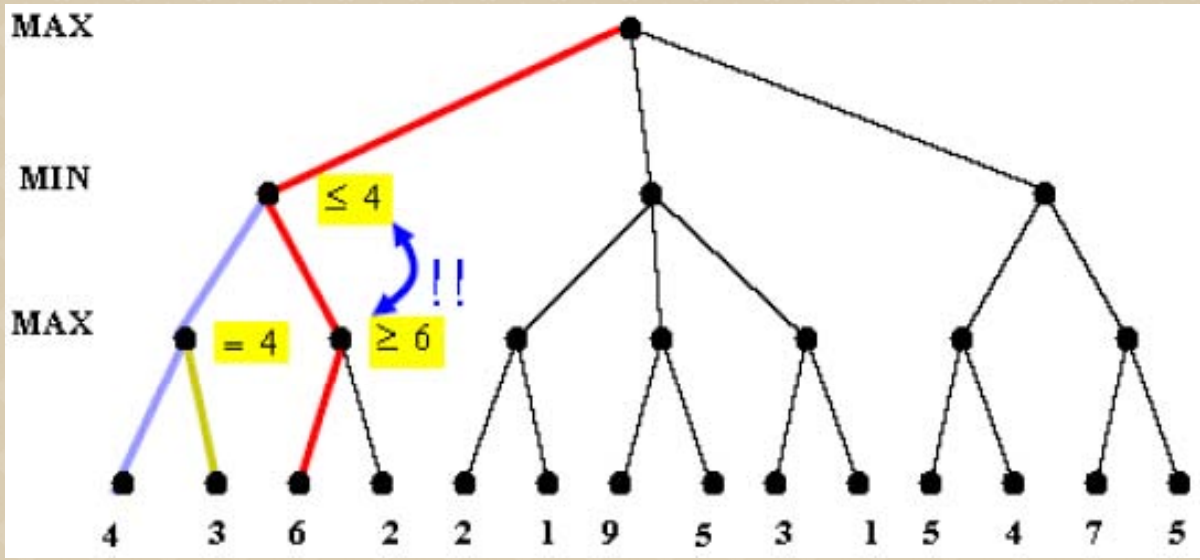
مثال : هرس آلفا بتا



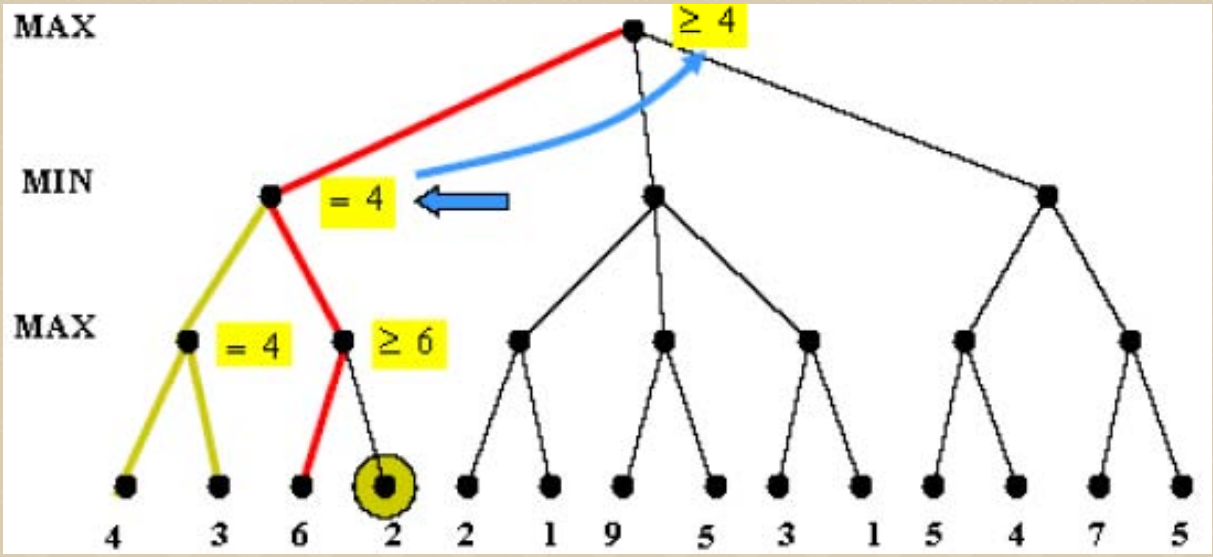
مثال : هرس آلفا بتا



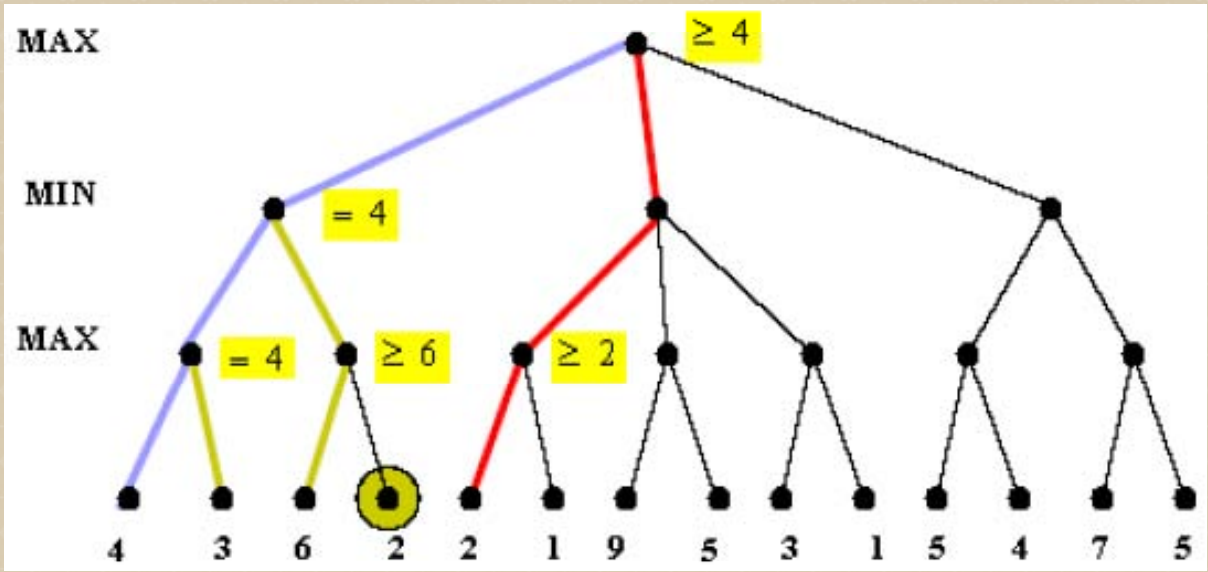
مثال : هرس آلفا بتا



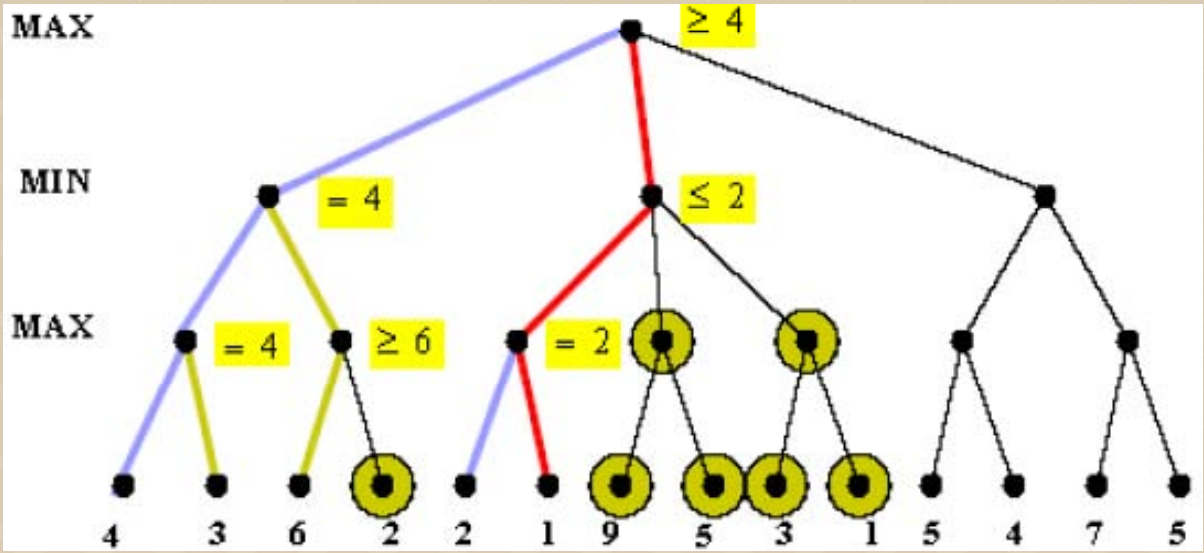
مثال : هرس آلفا بتا



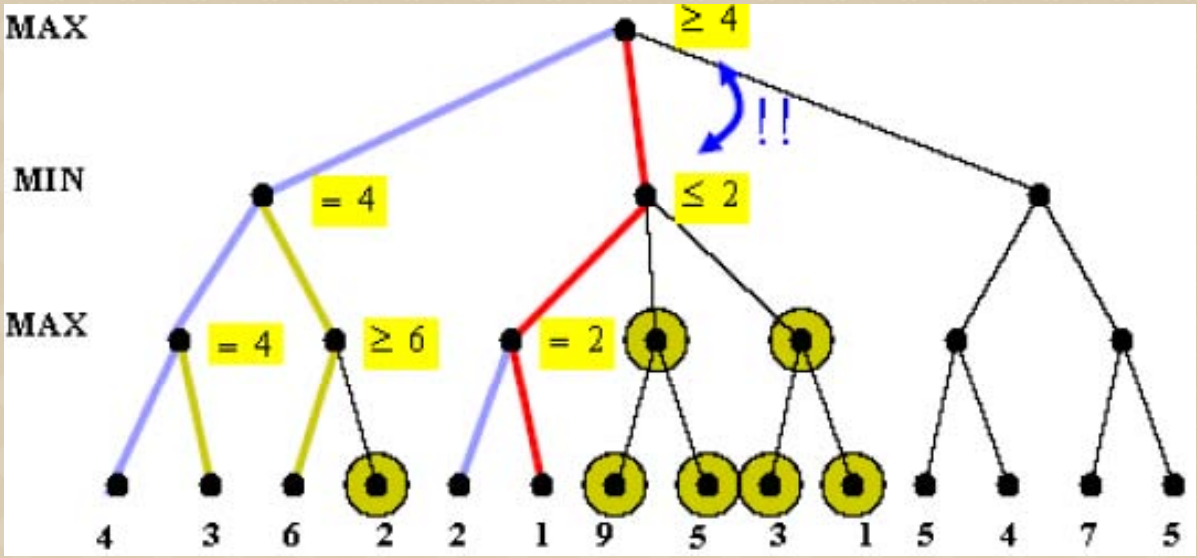
مثال : هرس آلفا بتا



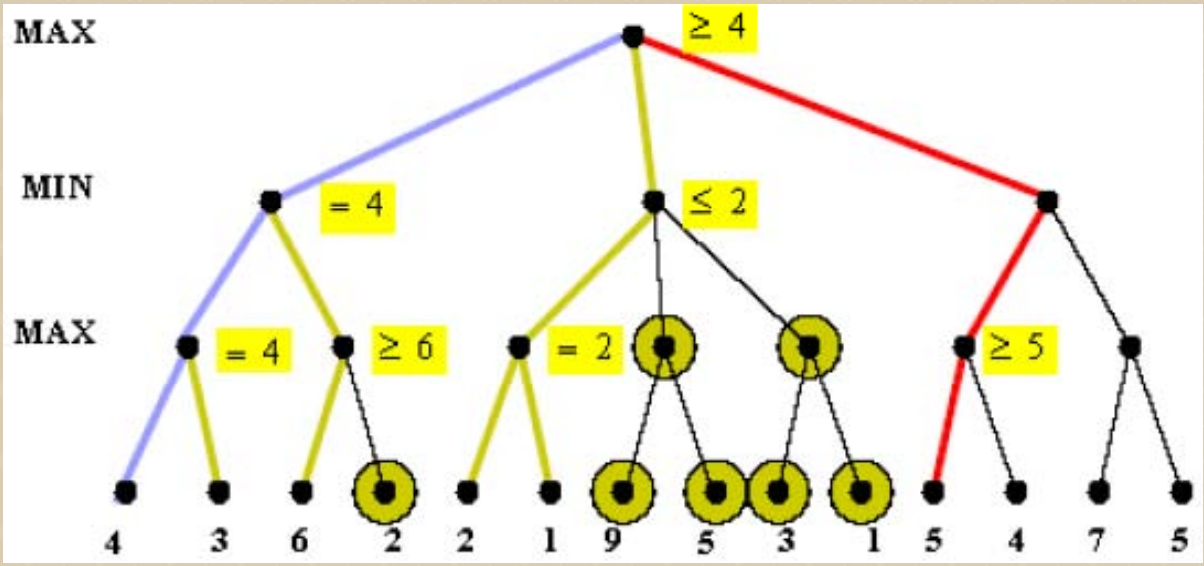
مثال : هرس آلفا بتا



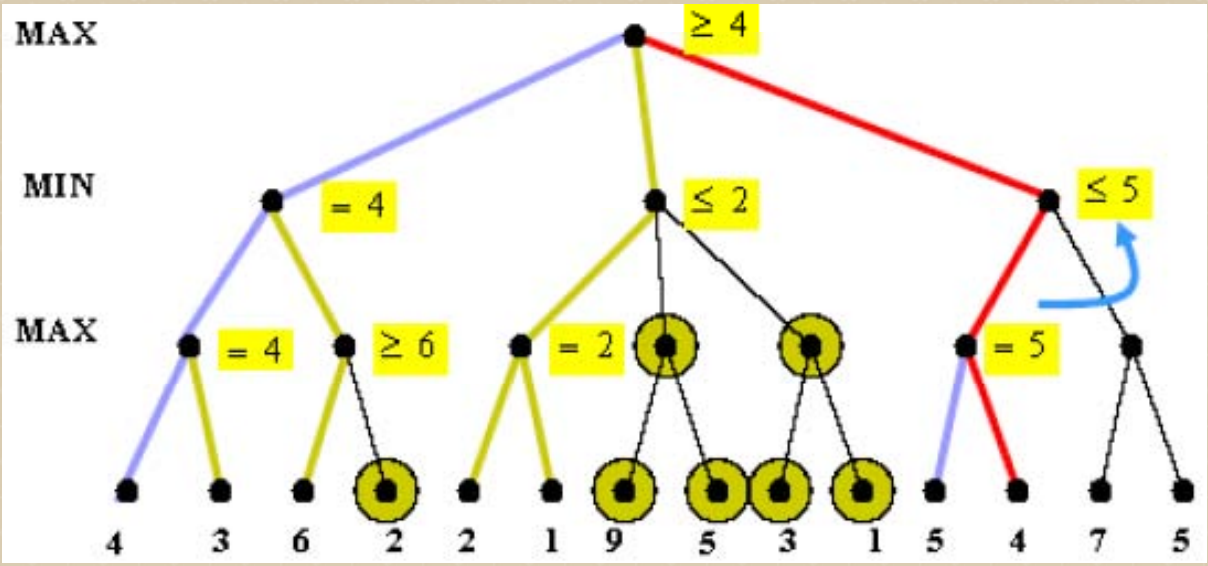
مثال : هرس آلفا بتا



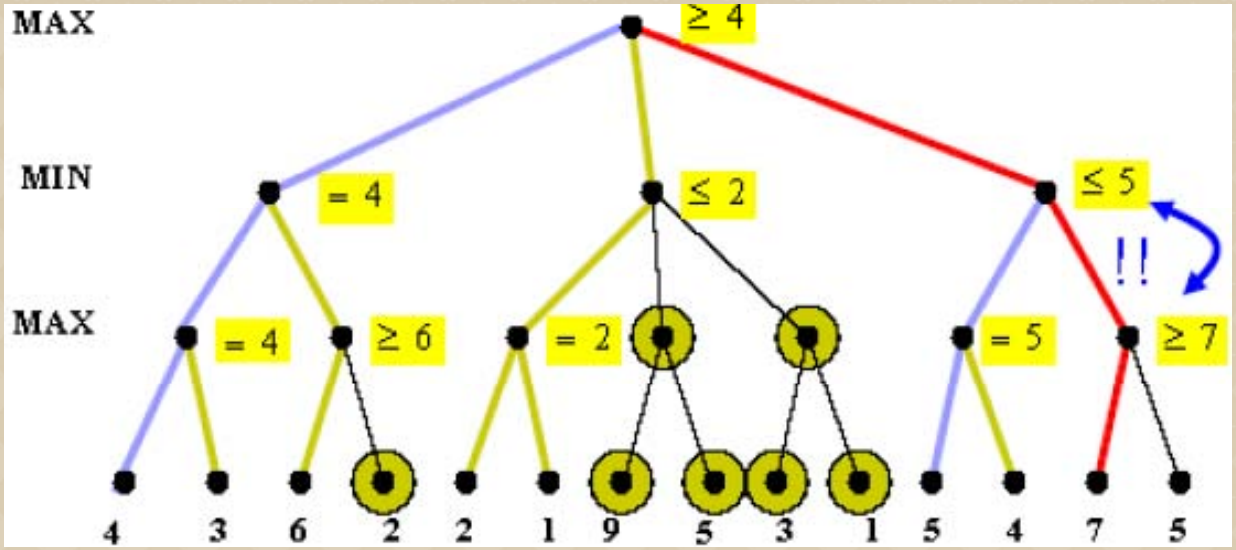
مثال : هرس آلفا بتا



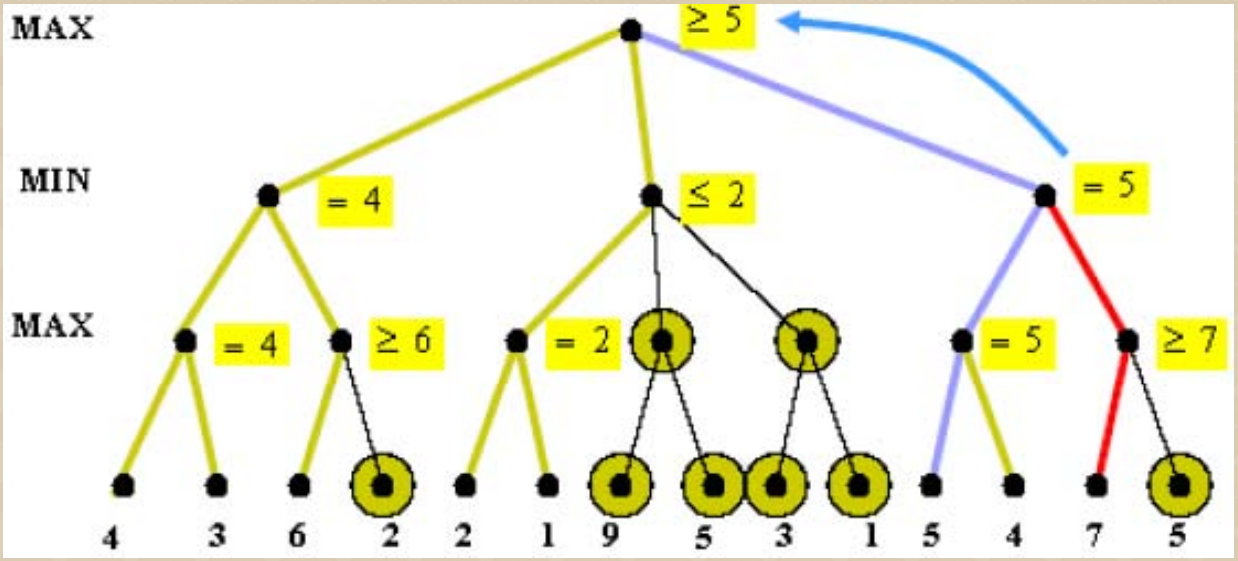
مثال : هرس آلفا بتا



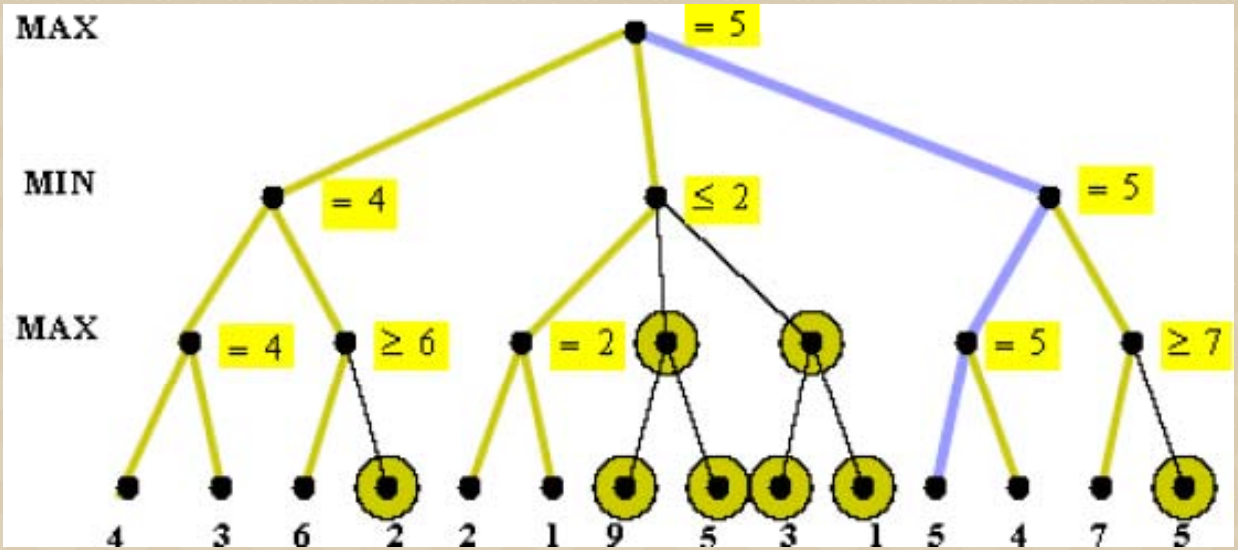
مثال : هرس آلفا بتا



مثال : هرس آلفا بتا



مثال : هرس آلفا بتا



حالات تکراری

❖حالات تکراری در درخت جستجو می توانند هزینه جستجو را به طور نمایی افزایش دهند.

❖در بازیها، به دلیل انجام جابه جاییهای متعدد، بروز حالات تکراری متداول است.

❖منظور از جابه جایی جای گشتهای مختلف توالی حرکات می باشد که به یک حالت انتهایی منتهی می شود.

- مثال: اگر سفید حرکت a_1 انجام دهد و سیاه آن را با b_1 پاسخ دهد و در ادامه حرکت مستقل دیگر a_2 با حرکت b_2 پاسخ داده شود، توالی حرکات $[a_1, b_1, a_2, b_2]$ و $[a_1, b_2, a_2, b_1]$ هر دو به یک موقعیت منتهی می‌شوند.
- همچنین می‌توان جای‌گشتها را با a_2 آغاز نمود.

اجتناب از حالات تکراری

❖شایسته است که ارزیابی این موقعیت را در اولین مرتبه وقوع، در یک جدول هش، ذخیره نماییم تا در حالات مشابه بعدی دیگر نیازی به محاسبه مجدد نباشد.

❖جدول هش(Hash) مربوط به موقعیتهای محاسبه شده قبلی که بدین ترتیب به دست می آید، به جدول جابه جایی معروف است.

- این جدول اساساً مشابه لیست *closed* در GRAPH-SEARCH می‌باشد.

معایب جدول جابجایی

❖استفاده از یک جدول جابه جایی *گاهی* اوقات میتواند تأثیرچشم گیری به اندازه دو برابر کردن عمق مجاز جستجو در شطرنج، داشته باشد.

❖از سوی دیگر، اگر فرض کنیم در جریان کار در هر ثانیه یک میلیون گره ارزیابی شوند، نگهداری و ذخیره تمام آنها در یک جدول جابه جایی ممکن نخواهد بود.

❖روشهای مختلفی به منظور انتخاب ارزشمندترین این گرهها به کار گرفته شده اند.

تصمیمهای بلادرنگ ناقص برای افزایش سرعت

❖الگوریتم بیشینه کمینه تمام فضای جستجوی بازی را تولید می کند، در حالی که الگوریتم آلفابتا به ما امکان حذف قسمت بزرگی از آن را می دهد.

❖الگوریتم آلفابتا باید تمام مسیر را تا رسیدن به حالات پایانی، حداقل در بخشی از فضا جستجو نماید.

❖این عمق جستجو معمولاً عملی نمی باشد، چون تمامی حرکات باید حداکثر در ظرف چند ثانیه انجام پذیرند.

❑روشهای تصمیم بلادرنگ(ناقص)

۱-تابع ارزیاب ← خطی / غیر خطی

۲-قطع جستجو

روش تصمیمهای بلادرنگ ناقص:قطع جستجو

- تابع سودمندی با یک تابع ارزیاب هیوریستیک (*EVAL*) که تخمینی از میزان سودمندی موقعیت ایجاد می‌کند، جایگزین شود.
- آزمون پایانی نیز با یک آزمون قطع جایگزین گردد که تشخیص می‌دهد در چه زمانی باید *EVAL* اعمال شود.

توابع ارزیاب(Eval)

❖یک تابع ارزیاب، تخمینی از مقدار سودمندی مورد انتظار یک موقعیت مفروض در بازی را در اختیار قرار می دهد.(مثل هیورستیکها)

❖نکته اول: تابع ارزیاب باید حالات انتهایی را همانند یک تابع سودمندی، مرتب نماید؛ در غیراین صورت، عاملی که از آن استفاده می کند، هر چند قادر باشد که انتهای تمامی مسیرها را بررسی کند، ممکن است حرکاتی غیر بهینه را انتخاب نماید.

❖نکته دوّم: محاسبات انجام شده نباید زیاد طولانی باشند.

❖نکته سوّم: این که در حالتهای غیر پایانی، تابع ارزیاب باید وابستگی زیادی به شانسهای حقیقی برد داشته باشد

روش کار توابع ارزیاب(Eval)

بیشتر توابع ارزیاب به وسیله محاسبه خصوصیات گوناگون حالت کار می کنند.

برای مثال، تعداد سربازهایی که هر کدام از بازیکنان در بازی شطرنج در اختیار دارند.

این خصوصیات در مجموع، طبقات یا دسته های مشابهی از حالات را تعریف می کنند: حالاتی که در یک دسته قرار می گیرند، دارای مقادیر یکسانی برای تمامی خصوصیات می باشند.

برای مثال فرض کنید از میان حالتهای این دسته، ۷۲ ٪ به یک برد (با مقدار سودمندی ۱) ۲۰ ٪ به یک باخت (با مقدار سودمندی ۱-) و ۸٪ به یک تساوی (با مقدار سودمندی ۰)منجر می شوند.

یک ارزیابی معقول برای حالات موجود در یک دسته، میانگین وزنی یا مقدار مورد انتظار می باشد:

(.72 × 1) + (.20 × -1) + (.08 × 0) = .52

روش کار توابع ارزیاب(Eval)

❖به طور کلی، هر دسته مفروض شامل حالاتی خواهد بود که برخی از آنها به برد، برخی به تساوی و برخی نیز به شکست منجر خواهند شد.
❖تابع ارزیاب قابلیت تشخیص این حالات از هم را ندارد ولی می تواند مقداری را که نشانگر نسبت حالتها با هر خروجی است، تولید نماید.
❖اغلب توابع ارزیاب برای سهم هر خصوصیت، مقادیر عددی تفاوتی را محاسبه و در نهایت در راستای پیدا کردن مقدارنهایی، آنها را با هم ترکیب می کنند.

❖برای مثال، در شطرنج، برای هر مهره یک ارزش ذاتی در نظر گرفته شده است: ارزش هر سرباز ۱ امتیاز، هر اسب یا فیل ۳ امتیاز، هر رخ ۵ امتیاز و وزیر ۹ امتیاز می باشد.
❖خصوصیات دیگر نظیر "مهره چینی مناسب سربازها" و "امنیت شاه" می تواند امتیازی معادل با نیم سرباز داشته باشد.
❖سپس مقادیر این خصوصیات برای ارزیابی موقعیت جاری با هم جمع می شوند.
❖یک برتری امن معادل با یک سرباز، احتمال برد را تا حد زیادی افزایش می دهد و یک برتری امن معادل با سه سرباز، به طور حتم به پیروزی خواهد انجامید.

انواع توابع ارزیاب

- توابع خطی وزن دار
- توابع غیر خطی

این نوع تابع ارزیاب، تابع خطی وزن دار نامیده می شود و می تواند به صورت ریاضی زیر بیان شود:

$$Eval(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s) =$$

- در این رابطه، w_i یک وزن و fi یک خصوصیت موقعیت را مشخص می‌کنند.

- در شطرنج fi می‌تواند تعداد هر نوع از مهره‌ها بر روی صفحه و w_iامتیاز مربوط به هر کدام از آنها باشد (۱ برای سرباز، ۳ برای اسب و فیل و ....)

جمع مقادیر خصوصیات، مستلزم یک فرض اساسی است و آن این است که سهم هر ویژگی، مستقل از مقادیر سایر ویژگیها می باشد.
برای مثال، در نظَر گرفتن مقدار ۳ برای فیل این حقیقت را که فیل در حرکتهای انتهایی بازی به دلیل داشتن فضای مانور بالا، بسیار مفید خواهد بود، در نظر نمی گیرد(مهمتر شده ولی...)

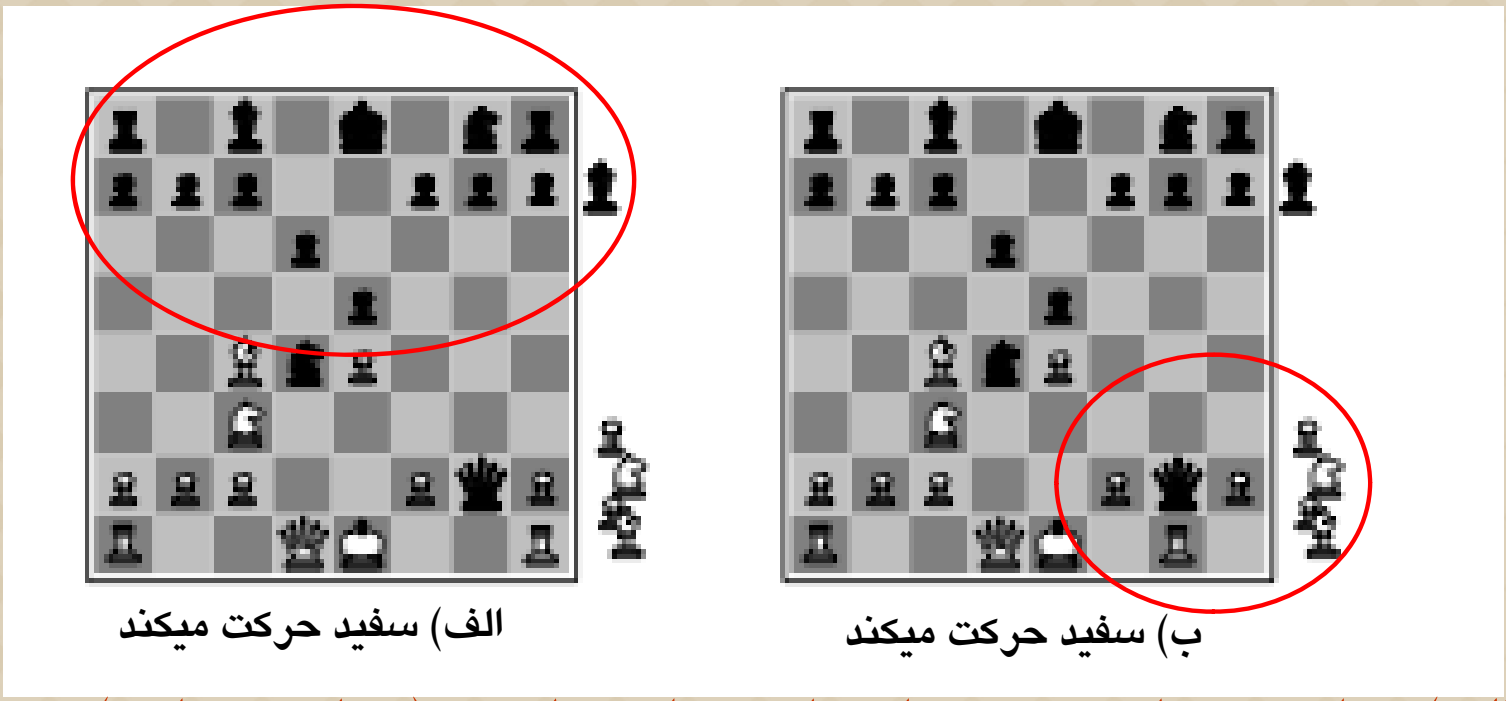
انواع توابع ارزیاب

طبق نظریه جمع مقادیر خصوصیات، مستلزم یک فرض اساسی است و آن این است که سهم هر ویژگی، مستقل از مقادیرِ سایر ویژگیها می باشد.

بدین منظور، برنامه های فعلی شطرنج و نیز سایر بازیها، از ترکیبات غیر خطی خصوصیات نیز بهره می گیرند.

مثلاً ارزش یک جفت فیل ممکن است کمی بیش از دو برابر ارزش یک فیل باشد و یک مهره فیل در انتهای بازی ارزشمندتر از یک فیل در ابتدای بازی می باشد.

اهمیت توابع ارزیابی غیرخطی



الف) سفید حرکت میکند

ب) سفید حرکت میکند

الف) سیاه، مزیت اسب و دو پیاده دارد و بازی را میبرد(تعداد مهم باشد)

ب) پس از اینکه سفید، وزیر را در اختیار میگیرد، سیاه میبازد(تعداد مهم نیست)

قطع جستجو(تا هرجا رفتیم بسه!!!)

❖صریح‌ترین رهیافت برای کنترل میزان جستجو قراردادن محدودیتی برای داشتن یک عمق ثابت است، بنابراین تست قطع برای تمام گره‌ها در زیر عمق dموفق می‌شود. عمق طوری انتخاب می‌شود که میزان زمان استفاده شده از آنچه که قوانین بازی اجازه می‌دهد تجاوز نکند.

❖با این حال، چنین روشهایی به دلیل ماهیت تخمینی تابع ارزیاب، ممکن است دچار خطا شوند.

❖بنابراین به یک روش قطع جستجوی مناسبتر احتیاج می باشد.

❖تابع ارزیاب تنها در موقعیتهای ساکن (که در آینده نزدیک در آنها احتمال تغییرات شدید در مقادیر بسیار کم می باشد) باید اعمال شود.مثلا در شطرنج گرفتن موقعیتهای مناسب هدف است، تابعی که تعداد مهره را بررسی می کند ساکن نیست ،با بسط موقعیتهای غیر ساکن می توان موقعیتهای ساکن ایجاد کرد.

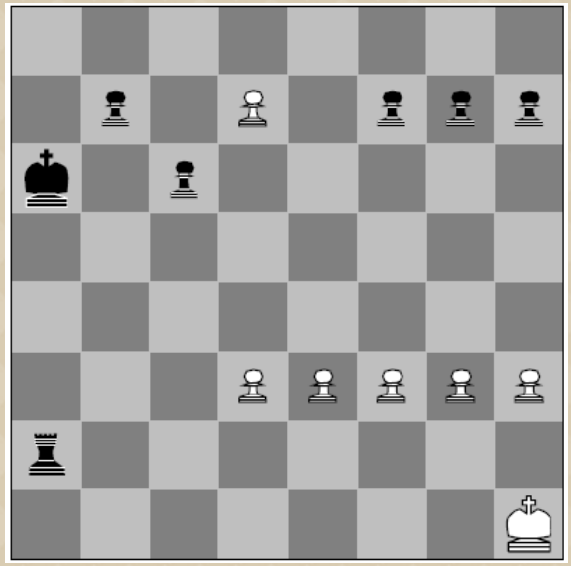
❖ایم تابع ارزیابی فقط باید برای مواقعی به کاربرده شود که خاموش هستند، یعنی اینکه تفاوتهای چشمگیر در مقدار، در آینده نزدیک بعید به نظر می‌رسد.

❖این جستجوی اضافی به جستجوی ساکن(خاموش) معروف می باشد.در بعضی از شرایط، تنها محدود به انجام یک سری از حرکتهای خاص می باشیم تا به واسطهٔ آنها عدم قطعیتهای موجود در یک موقعیت خاص از میان برود.

افق دید

- اثر افق دید زمانی رخ خواهد داد که برنامه به واسطه یک حرکت حریف با شرایطی روبرو شود که بروز یک لطمه شدید، اجتناب ناپذیر باشد.

- انجام یک سری کیش به وسیله رخ سیاه، تبدیل اجتناب ناپذیر سرباز سفید به وزیر را ، "از افق دید خارج می سازد" و چنین می نماید که این موقعیت به برد مهره سیاه خواهد انجامید، در حالی که در حقیقت یک برد برای سفید رقم خواهد خورد.



مقابله با افق دید

- از آنجا که پیشرفتهای سخت افزاری موجب انجام جستجوهای عمیقتری شده است، انتظار می رود که اثر افق به ندرت رخ دهد یعنی رویه های تأخیری بسیار به ندرت ایجاد شوند.

- به منظور جلوگیری از اثر افق، استفاده از تعمیمهای فردی (تکین)نیز بدون افزودن هزینه جستجوی بالا، می تواند بسیار مؤثر باشد.

- یک تعمیم تکین، حرکتی است که در یک موقعیت مفروض، از تمام حرکتهای دیگر ممکن در آن موقعیت "به وضوح، بهتر" باشد.

- به دلیل دارا بودن فاکتور انشعاب ۱، یک جستجوی تعمیم تکین می تواند بدون صرف هزینه بالا از حد نرمال عمق جستجو فراتر رود.

هرس پیش رو(بدون علت منطقی)

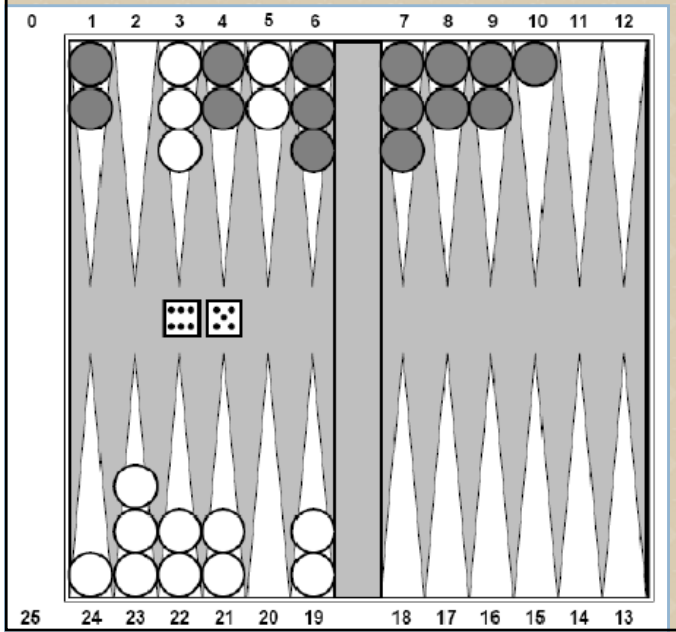
- **تاکنون در مورد قطع جستجو در یک مرحله خاص و هرس آلفابتای بدون تأثیر بر نتیجه نهایی صحبت شد.نوع دیگر هرس که گاها روی نتیجه تاثیر گذار است هرس پیش رو است**
- **هرس پیش رو به معنای آن است که یک سری از حرکتها در یک گره مفروض بدون در نظر گرفتن حذف شوند.**
- **بسیاری از بازیکنان شطرنج در هر موقعیت یا گره، تنها یک سری حرکات محدود (حداقل آگاهانه) را در نظر می گیرند، نه تمام آنها را.**
- هیچ تضمینی در این روش مبنی بر عدم حذف حرکت بهینه به واسطه هرس وجود ندارد.همچنین انجام این روش در مجاورت ریشه به دلیل امکان حذف یک سری از حرکتهای مناسب فاجعه است.
- هرس پیش رو در شرایط خاص می تواند مؤثر واقع شود. مثلاً هنگامی که دو حرکت قرینه یا معادل هم هستند، تنها یکی از آنها در نظر گرفته می شود یا در مورد گرههایی که در عمق پایین در درخت جستجو واقع شده اند می توان از این روش استفاده کرد.

بازیهای دارای شانس(غیر قطعی)

- در بسیاری از بازیها شرایطی پیش بینی نشده با افزودن یک عنصر احتمالی، نظیر پرتاب تاس، در نظر گرفته شده است.
- تخته نرد یکی از معروفترین بازیهایی است که در آن هم مهارت و هم شانس در رسیدن به موفقیت نقش دارند.
- تاسها در نوبت بازی هر بازیکن به منظور تعیین حرکات مجاز ریخته می شوند.

بازی تخته نرد

- هدف این بازی، انتقال تمامی مهره ها به خارج از بازی می باشد.
- سفید در جهت حرکت عقربه های ساعت به سمت ۲۵ و سیاه در جهت عکس آن یعنی به سمت صفر حرکت می کند.
- یک مهره می تواند به هر خانه انتقال یابد مگر آن که دو یا تعداد بیشتری از مهره های حریف در آن خانه باشند و در صورت بودن یک مهره حریف، آن مهره به اسارت گرفته می شود و باید از ابتدا شروع کند.



بازی تخته نرد

- اگر چه سفید حرکات مجاز خود را می‌داند ولی هیچ آگاهی از حرکت بعدی سیاه ندارد، زیرا پرتاب تاسها آن را معیّن می‌کند.
- این بدان معنی است که سفید نمی‌تواند یک درخت بازی استاندارد بسازد.
- درخت بازی در تخته نرد علاوه بر گرههای MAX و MIN باید دارای گرههای شانس نیز باشد.

بازی های دارای عامل شانس

- قدم بعد، اتخاذ تصمیم صحیح می باشد. هدف ما انتخاب حرکتی خواهد بود تا بهترین موقعیت را نتیجه شود.
- موقعیتهای حاصله دارای مقادیر بیشینه کمینه مشخصی نیستند.
- می توان مقادیر بیشینه کمینه مورد انتظار را برای احتمالهای مختلف پرتاب تاسها در یک گره محاسبه کرد.
- این امر به تعمیم مقدار بیشینه کمینه در بازیهای معین، به مقدار بیشینه کمینه مورد انتظار در بازیهای دارای گره شانس۷۸ می باشد، می انجامد.

- گره های برگ و نیز گره های **Max** و **Min** (که در آنها مقادیر تاسها مشخص است) دقیقاً همانند قبل عمل می کنند،ولی گرههای شانس با محاسبه میانگین وزنی مقادیر حاصل از تمامی حالات ممکن، به ترتیب ذیل ارزیابی می شوند:

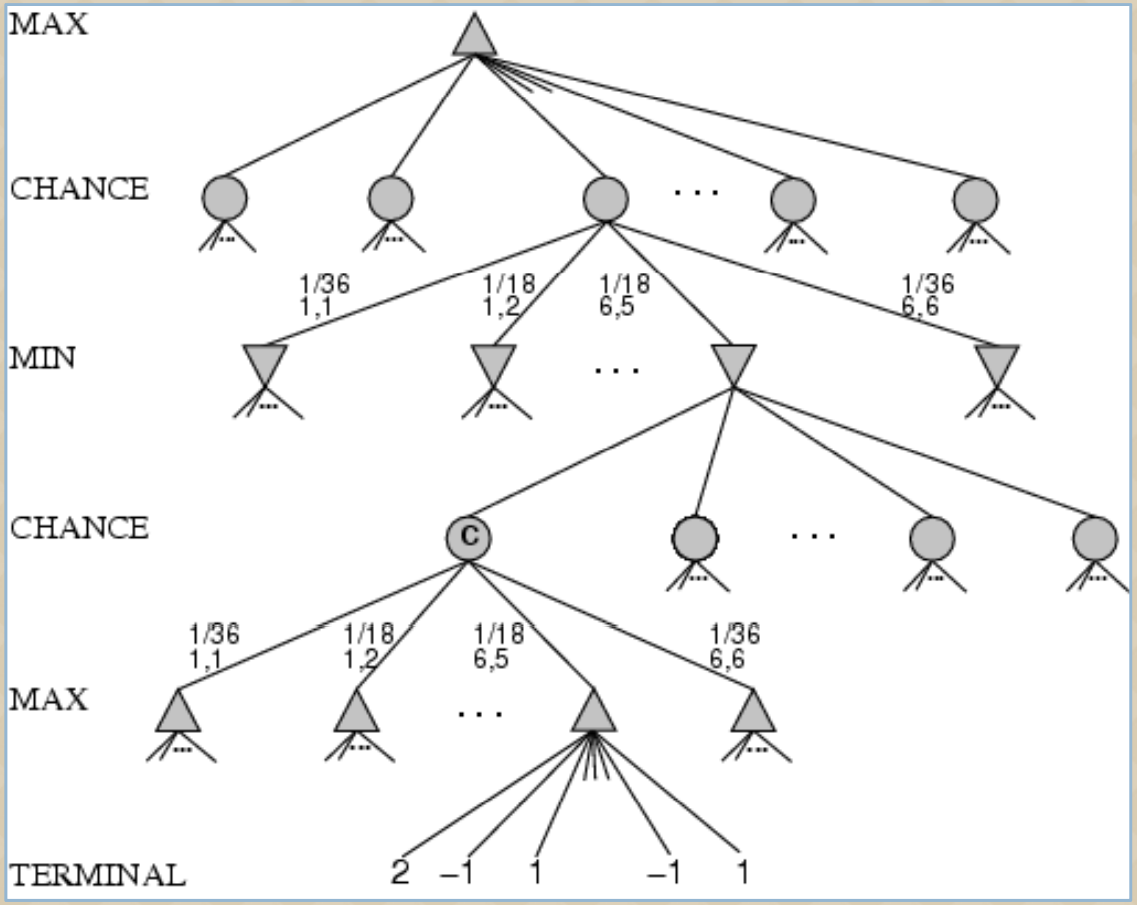
$$
\begin{aligned}
&Expectiminimax\ (n) = \\
&\begin{cases}
UTILITY\ (n) & \text{if } n \text{ is a terminal state} \\
\max_{s \in Successors(n)} EXPECTIMINIMAX\ (s) & \text{if } n \text{ is a MAX node} \\
\min_{s \in Successors(n)} EXPECTIMINIMAX\ (s) & \text{if } n \text{ is a MIN node} \\
\sum_{s \in Successors(n)} P(s).\ EXPECTIMINIMAX\ (s) & \text{if } n \text{ is a chance node}
\end{cases}
\end{aligned}
$$

بازی های دارای عامل شانس

- تابع جایگزینِ گره شانس n، حالت n را با تمامی پرتابهای ممکن تاس، گسترش می‌دهد تا تمامی جایگزینها مانند s را تولید کند و $P(s)$ احتمال وقوع آن پرتاب تاس است.

- این تساویها مانند روش بیشینه‌کمینه، می‌توانند به صورت بازگشتی، تمام مسیر را از پایین به سمت ریشه درخت، برگرداند.

درخت بازی برای تخته نرد در یک موقعیت خاص

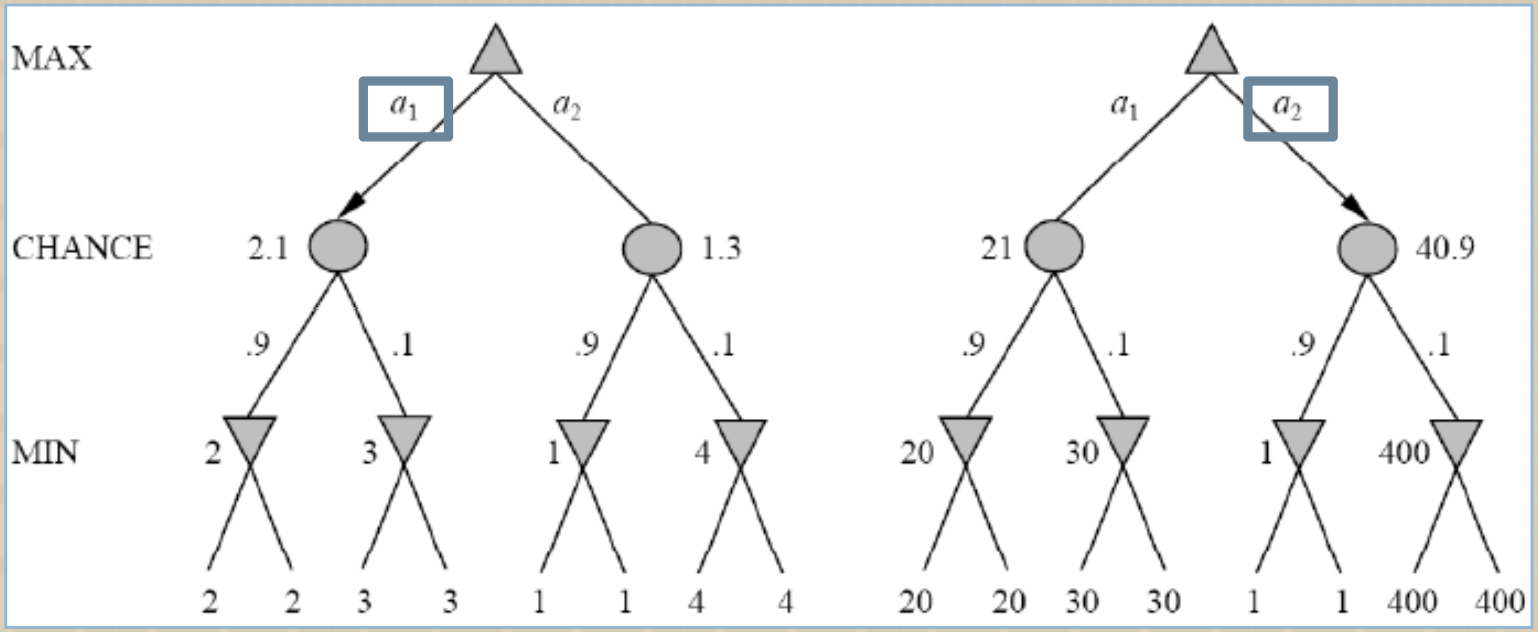


ارزیابی موقعیت در بازی های شانسی

- در بیشینه کمینه مورد انتظار هم، تخمین به صورت قطع جستجو در بعضی نقاط و اعمال یک تابع ارزیاب به هر یک از آن نقاط می باشد.
- ممکن است در ابتدا تصور شود که تابع ارزیابِ بازیهایی مانند تخته نرد، همانند تابع ارزیابِ شطرنج می باشد که بَه موقعیتهای بهتر، امتیاز بالاتر می دهد.
- ولی در حقیقت حضور گرههای شانس در درخت بازی، به معنی دقت بیشتر در تفسیر معنای مقادیر ارزیابی می باشد.

یک تبدیل برای مقادیر برگی(راهبرد بهینه)

- با تابع ارزیابی که مقادیر [۱و۲و۳و۴] را به گرههای برگی نسبت می‌دهد، حرکت A_1بهترین حرکت می‌باشد، ولی با مقادیر [۱و ۲۰و ۳۰و ۴۰۰] حرکت A_2 بهترین حرکت تشخیص داده می‌شود.



2.1=2*.9+3*.1

پیچیدگی روش بیشینه کمینه مورد انتظار(شانسی)

- با فرض دانستن مقادیر تاسها تا انتهای بازی، بررسی راه حل یک بازی دارای تاس همانند بازیهای مدرن با تاس خواهد بود که به کمک روش بیشینه‌کمینه در $O(b^m)$ تکرار، به دست خواهد آمد.

- روش بیشینه‌کمینه مورد انتظار، تمامی توالیهای ممکن پرتاب تاسها را نیز در نظر می‌گیرد، لذا راه حل در $O(b^m n^m)$ بار تکرار حاصل خواهد شد، که در آن n، تعداد حالات مختلف پرتاب تاسها می‌باشد.

b = فاکتور انشعاب(تعداد حالات مجاز در هر گره)
m = حداکثر عمق درخت
n =تعداد حالات شانش

پیچیدگی روش بیشینه کمینه مورد انتظار(شانسی)

- اگر عمق جستجو، محدود به مقدار کوچک d باشد، هزینه اضافی جستجو نسبت به بیشینه‌کمینه، مانع از در نظر گرفتن آینده بسیار دور، در اغلب بازیهای دارای شانس می‌شود.

- در تخته نرد، n برابر ۲۱ است و b معمولاً در حدود ۲۰ می‌باشد که در برخی حالات، برای تاسهای جفت، می‌تواند تا ۴۰۰۰ نیز باشد. لذا حداکثر میزان پیش‌روی احتمالاً ۳ لایه خواهد بود.

بازیهای دارای عامل شانس

- یکی از مزیتهای هرس آلفابتا آن است که این روش شاخه هایی از درخت که احتمال وقوع آنها با فرض انجام بهترین بازی، بسیار پایین است را در نظر نمی گیرد و تنها بر حالتهایی با امکان وقوع بالا تمرکز می کند.
- در بازیهای دارای تاس، هیچ ترتیبی از حرکات با احتمال وقوع بالا وجود ندارد، چون لازمه وقوع آنها، آمدن تاسها به شکل مورد انتظار است.
- بنابراین یکی از مشکلات عمومی در مواجهه با عدم قطعیت، بالا بودن تعداد حالات ممکن می باشد.
- میتوان روشی مشابه هرس آلفابتا را به درختهای بازی که دارای گرههای شانس می باشند، اعمال کرد.
- در این روش بررسی گرههای Min و Max همانند گذشته است ولی می توان گرههای شانس را نیز هرس کرد.
- ۱-۱,۱-۲,۱-۳,۱-۴,۱-۵,۱-۶
- این حد بالا در واقع همان مقداری است که در روش آلفابتا برای هرس یک گره وانشعابهای آن بدان نیاز می باشد.

بازیهای دارای عامل شانس

- در نگاه اوّل چنین امری غیر ممکن مینماید، زیرا مقدار گره شانس در حقیقت برابر است با میانگین مقادیر انشعابهای آن تا زمانی که تمامی این مقادیر در نظر گرفته نشوند این میانگین می تواند هر مقداری باشد.

- ولی اگر برای مقادیر تابع سودمندی کرانهایی قائل شویم،می توان به کرانهایی برای مقدار میانگین نیز رسید.

- برای مثال اگر فرض کنیم تمام مقادیر سودمندی دارای مقداری بین ۳+ و ۳- باشند، مقدار گره برگی نیز دارای یک حد می باشد و در نهایت می توانیم یک حد بالا برای گره شانس بدون نگاه به انشعابهای آن اختصاص دهیم.

پیشرفته ترین برنامه های بازی

Chess: Deep Blue در سال ۱۹۹۷ قهرمان دنیای انسان ها (کاسپاروف) را شکست داد. Deep blue در یک ثانیه ۲۰۰ میلیون موقعیت را جستجو می کند و از توابع ارزیابی بسیار پیچیده ای استفاده می کند.

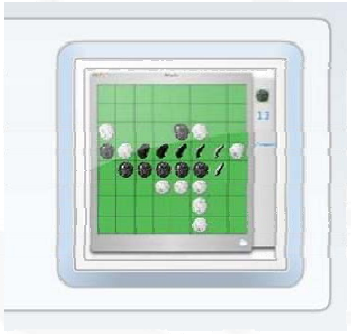
Othello: قهرمان های انسانی از رقابت با کامپیوترها (که خیلی خوب هستند) امتناع می ورزند.

Go: قهرمان های انسانی از رقابت با کامپیوترها (که بسیار بد هستند) امتناع می ورزند. در بازی Go فاکتور انشعاب بیشتر از ۳۰۰ می باشد، بنابراین اکثر برنامه ها برای ارائه حرکت های معقول از پایگاه های دانش الگویی استفاده می کنند.

شطرنج

Otello

Otello: قهرمانان جهان از مسابقه با کامپیوترها سر باز زده‌اند، زیرا شانسی برای برد نمی‌بینند.



GO

- برای اولین کسی که قادر باشد این بازی را طوری پیاده‌سازی کند که بتواند از یک انسان متوسط ببرد، یک میلیون دلارجایزه در نظر گرفته شده است.