« به نام او »

آموزش شبیه سازی دو بعدی فوتبال

نویسنده : محمد علی میرزایی - علی یعقوبی آزاد

مرجع روبوکاپ ایران

www.iranrcss.com

جلسه چهاردهم

مباحث این جلسه :

۱- چگونه مثل یک فرد روبوکاپ کار فکر کنیم ؟

۲- آموزش هوش مصنوعی، جستجوی خصمانه

آموزش چگونه مثل یک فرد روبوکاپ کار فکر کنیم؟

در جلسه ی سیزدهم در این باره صحبت کردیم که همه چیز به خود فرد بستگی دارد . به ادامه ی این مبحث توجه کنید :

همیشه سعی کنید برای هر حرکت بازیکنتان دلیلی بیابید ؛ یعنی همیشه بگید چرا در این سیکل بازیکن من پاس داد ؟ شما نباید سرسری از برخی مسائل بگذرید . دقت راز موفقیت در همه جا ست . شما باید در همه چیز دقیق باشید و سعی کنید همیشه برای هر مساله ای دلیلی منطقی بیابید .

به بازی تیم های دیگر توجه کنید . باید همیشه بازی تیم های دیگر را زیر نظر بگیرید : نقطه ضعف ، نقطه قوت ، چینش بازیکنان ، الگوریتم های بکار رفته و .... بازی هر تیم می تواند بهترین استادها برای شما بار علمی بیشتری به ارمغان بیاورد . تاکتیک های به کار گرفته شده در تیم های مختلف می تواند جرقه ای برای ایده سازی و الگوریتم نویسی شما شود . مو به مو باید بازی تمام تیم ها را بررسی کنید . همه ی تیم ها چیزی برای بدست آوردن دارند . از تیم اول جهان گرفته تا تیم آخر . همه و همه برای شما حکم یک استاد را دارند که باید خودتان از او طلب علم کنید .

برای خودتان هدف تعیین کنید . هدف نهایی تان را به اهداف کوتاه مدت تبدیل کنید . کسی که هدف نداشته باشد ، شکست میخورد پس همیشه با هدف کار کنید .

برنامه ریزی داشته باشید . همیشه با برنامه پیش بروید و بدون برنامه کار نکنید . تیمی که متشکل از حداقل ۲ نفر است ، احتیاج به برنامه دارد . برای تیمتان رهبر انتخاب کنید و همیشه به حرف های او عمل کنید . برنامه و تقسیم کار دو تا از مهمترین عوامل موفقیت یک تیم است .

همیشه مساله ی گم شدن جدیدترین نسخه تیم وجود دارد پس برای تیم هایی که مینویسید به این صورت کد بزنید :

YourTeamName.N۱.N۲.N۳

N۱ تا N۳ اعدادی هستند که از صفر آغاز میشوند و به صورت زیر افزایش می یابند :

N۱ : با هر بار جمع کردن کانل تیم و یکپارچه شدن کد تیم یک عدد اضافه می شود .

N۲ : با هر بار شرکت تیم در مسابقات یک عدد اضافه میشود .

N۳ : با هر بار انتشار تیم یک عدد اضافه می شود .

تا بدین جا ما در مورد چگونگی ساختار و تفکر یک تیم صحبت کردیم . امیدوارم تا بدین جای آموزش ها مفید بوده باشد . از جلسه ی بعد به جنبه های دیگر شبیه سازی دو بعدی نظیر : معرفی مسابقات مختلف ، شیوه ی نوشتن گزارش فنی ، شیوه ی نوشتن یک تیم به صورت اصولی و .... خواهیم پرداخت .

از مختصر بودن این جلسه عذرخواهی میکنیم. مقالات آینده بسیار جامع تر خواهند بود. هرگونه سوالی و یا انتقاد و پیشنهادی داشتید با ما در میان بگذارید . ما با نظرات شما پیشرفت خواهیم کرد .

باتشکر

گروه شبیه سازی دوبعدی مرجع روبوکاپ ایران

جستجوی خصمانه

فهرست :

- **بازیها چیستند و چرا مطالعه میشوند؟**
- **انواع بازیها**
- **الگوریتم minimax**
- **بازیهای چند نفره**
- **هرس آلفا- بتا**
- **بازیهای قطعی با اطلاعات ناقص**
- **بازیهایی که حاوی عنصر شانس هستند.**

بازی ها چیستند و چرا مطالعه میشوند؟

بازیها حالتی از محیطهای چند عاملی هستند :

- هر عامل نیاز به در نظر گرفتن سایر عاملها و چگونگی تأثیر آنها دارد.
- تمایز بین محیطهای چند عامل رقابتی و همکار
- محیطهای رقابتی، که در آنها اهداف عاملها با یکدیگر برخورد دارند، منجر به مسئله های خصمانه میشود که به عنوان بازی شناخته میشوند.

چرا مطالعه میشوند؟

- قابلیتهای هوشمندی انسانها را به کار میگیرند.
- ماهیت انتزاعی بازی ها

- حالت بازی را به راحتی میتوان نمایش داد و عاملها معمولا به مجموعه کوچکی از فعالیتها محدود هستند که نتایج آنها با قوانین دقیقی تعریف شده اند.

انواع بازی ها

تصادفی	قطعی	
تخته نرد	**شطرنج ریورسی**	**اطلاعات کامل**
پوکر		**اطلاعات ناقص**

یک نمونه بازی

بازی دو نفره: Min و Max

- **اول Max حرکت میکند و سپس به نوبت بازی میکنند تا بازی تمام شود.**
- **در پایان بازی، برنده جایزه و بازنده جریمه میشود.**

بازی به عنوان یک جستجو:

- **حالت اولیه: موقعیت صفحه و شناسه های قابل حرکت**
- **تابع جانشین:لیستی از (حالت,حرکت) که معرف یک حرکت معتبر است.**
- **آزمون هدف:پایان بازی چه موقع است؟(حالتهای پایانه)**
- **تابع سودمندی: برای هر حالت پایانه یک مقدار عددی را ارائه میکند. مثلا برنده(۱+) و بازنده(۱-)**

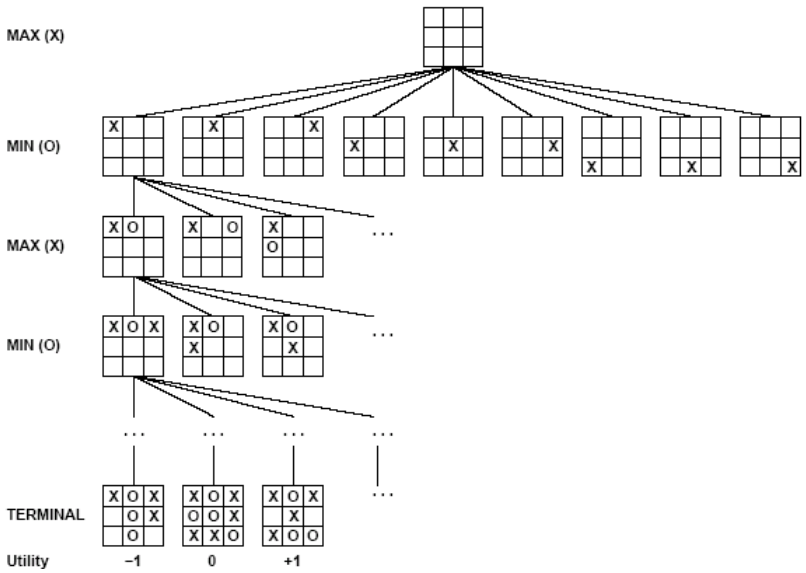
حالت اولیه و حرکات معتبر برای هر بازیکن، درخت بازی را برای آن بازی ایجاد میکند.

الگوریتم؛

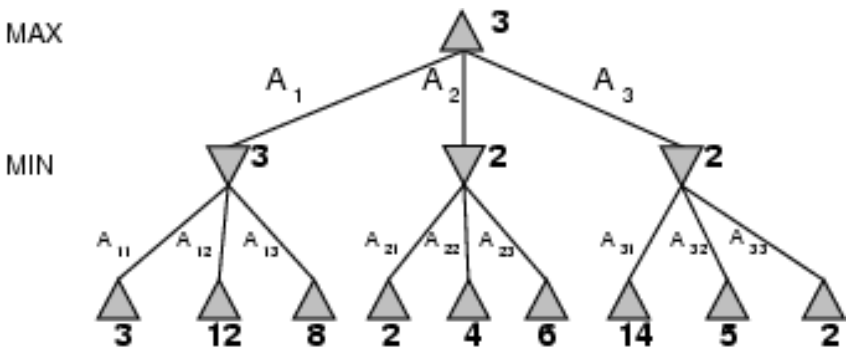
- **بازیکن: انتخاب بهترین حالت**
- **حریف: انتخاب بهترین موقعیت برای خودش یا بدترین وضعیت برای بازیکن**

بازیکن: ماکزیمم حالت

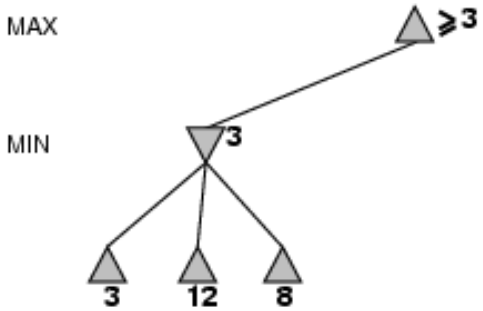
حریف: مینیمم حالت

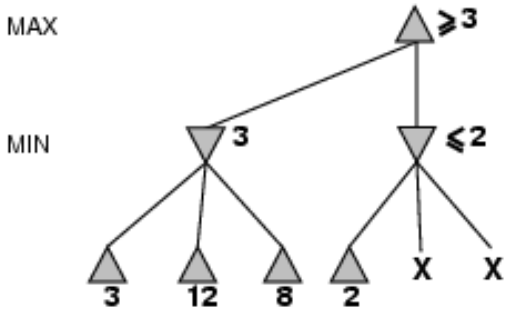


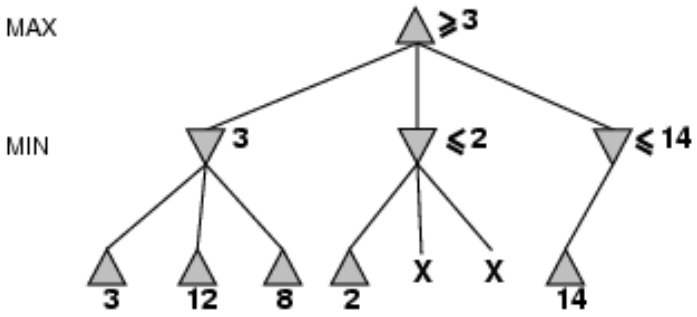
الگوریتم minimax

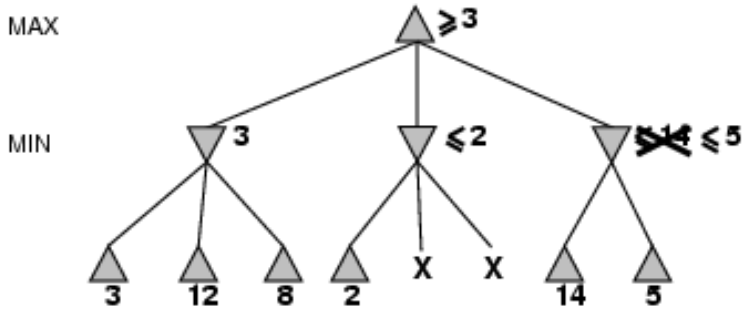


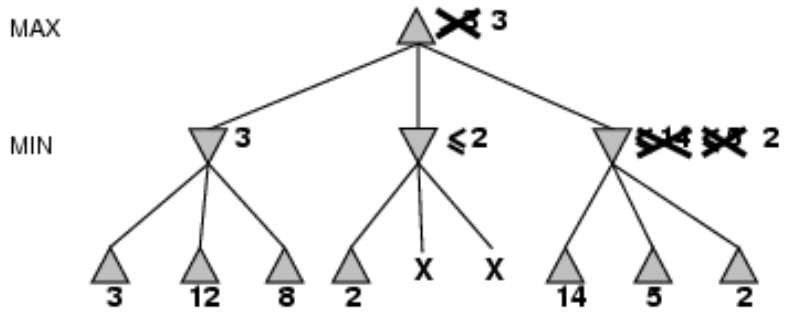
یک نمونه بازی











کامل بودن: بله (اگر درخت محدود باشد)

بهینگی: بله

پیچیدگی زمانی: $O(b^m)$

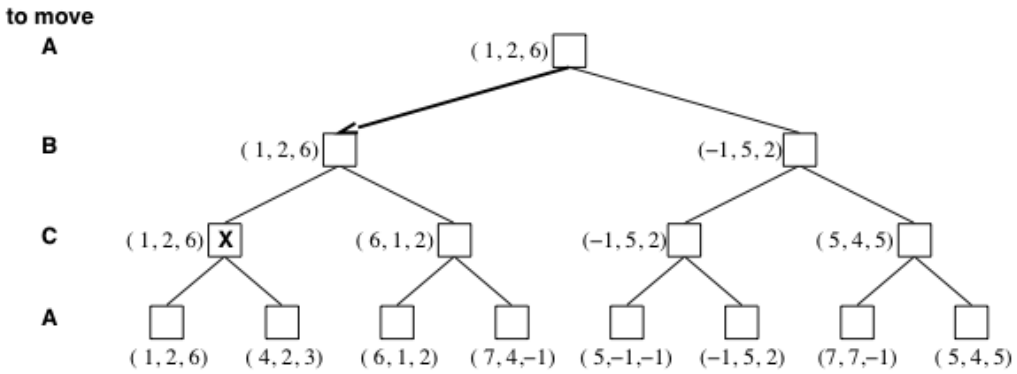
پیچیدگی فضا: $O(bm)$

بازیهای چند نفره

تخصیص یک بردار به هر گره، به جای یک مقدار

بازیهای چند نفره معولاً شامل اتحاد رسمی یا غیر رسمی بین بازیکنان است.

- اتحاد با پیشروی بازی ایجاد و از بین میرود.
- بازیکنان بطور خودکار همکاری میکنند، تا به هدف مطلوب انحصاری برسند.



هرس آلفا-بتا

در الگوریتم MaxMin:

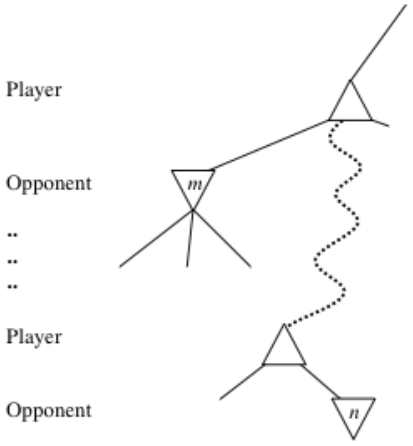
تعداد حالتهای بازی که باید بررسی شوند، بر حسب تعداد حرکتها، توانی است.

- راه حل: محاسبه تصمیم الگوریتم، بدون دیدن همه گره ها امکانپذیر است.

انشعابهایی که در تصمیم نهایی تأثیر ندارند را حذف میکند.

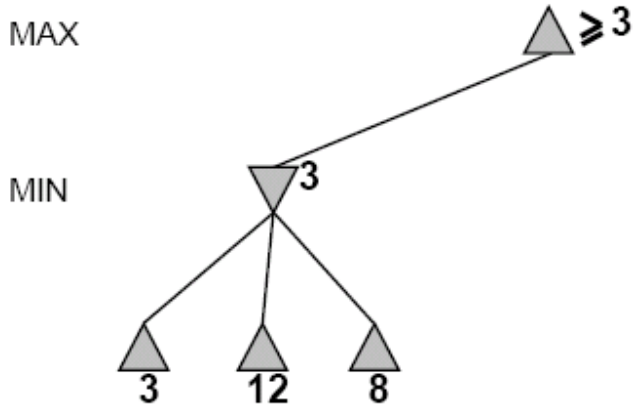
- آلفا: مقدار بهترین انتخاب در هر نقطه انتخاب در مسیر Max تاکنون

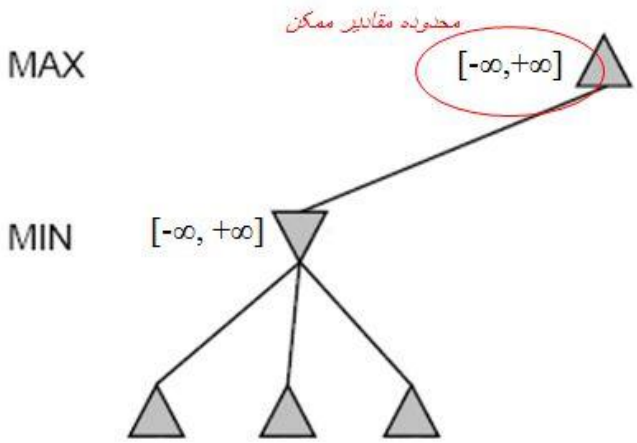
- بتا: مقدار بهترین انتخاب در هر نقطه انتخاب در مسیر Min تاکنون
- تعداد گره هایی که باید بررسی شوند به $O(b^{d/2})$ تقلیل میابد.
- فاکتور انشعاب مؤثر به جای b برابر با جذرb خواهد بود.
- پیش بینی آن نسبت به minimax دو برابر است.

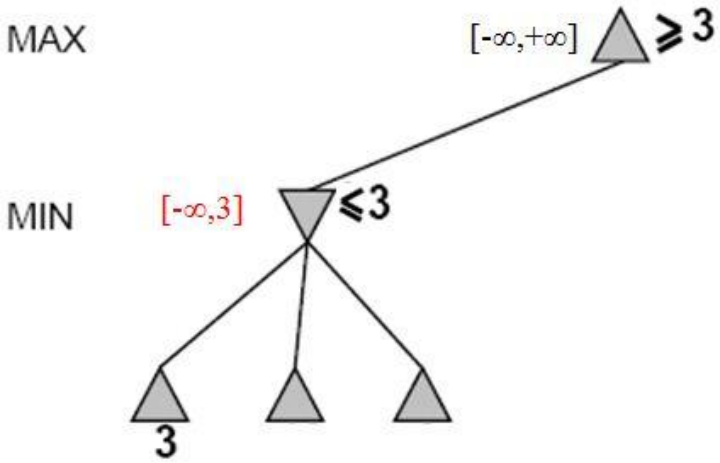


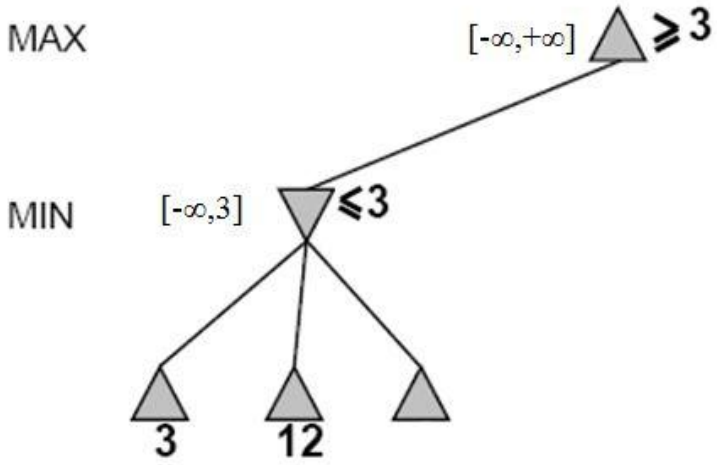
- گره n که هر جای درخت میتواند باشد، بررسی میشود.
- اگر بازیکن انتخاب بهتری داشته باشد.
  - در گره والد n
  - یا هر انتخاب بهتری تا کنون
- n هیچوقت در بازی واقعی قابل دسترس نخواهد بود.
- در نتیجه n هرس میشود.

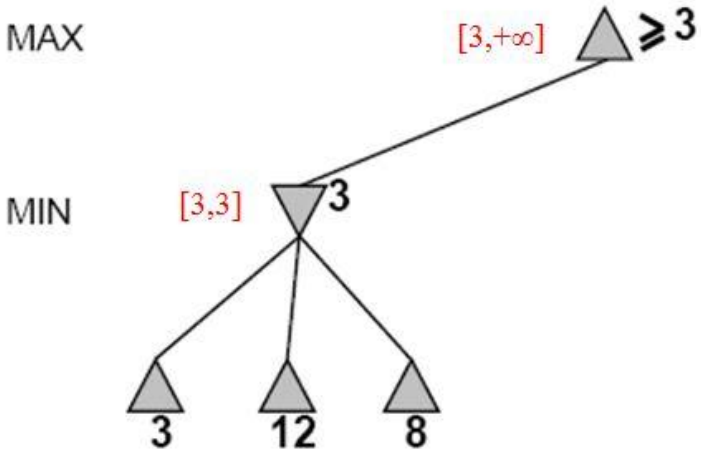
مثال: هرس آلفا-بتا

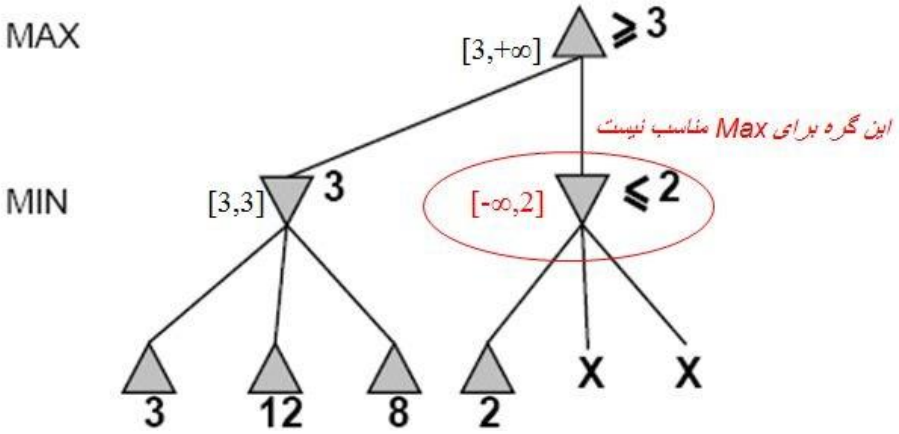


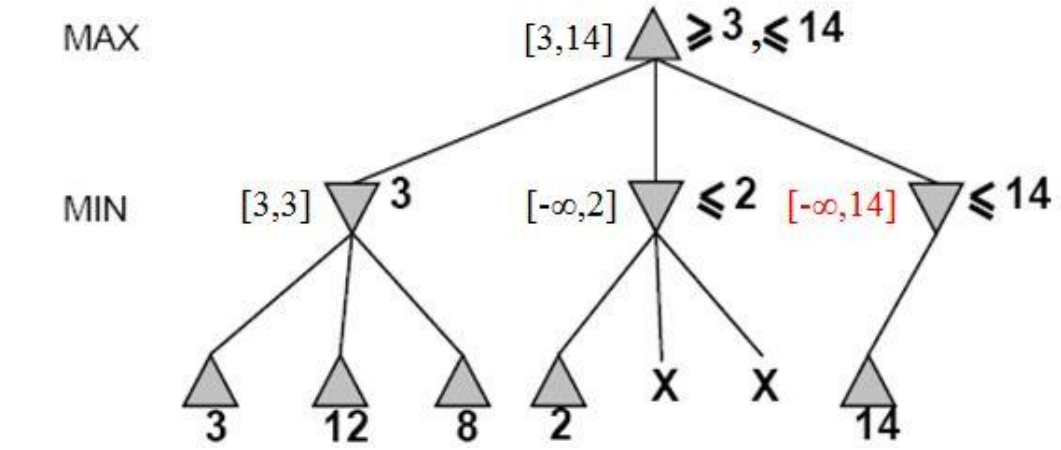


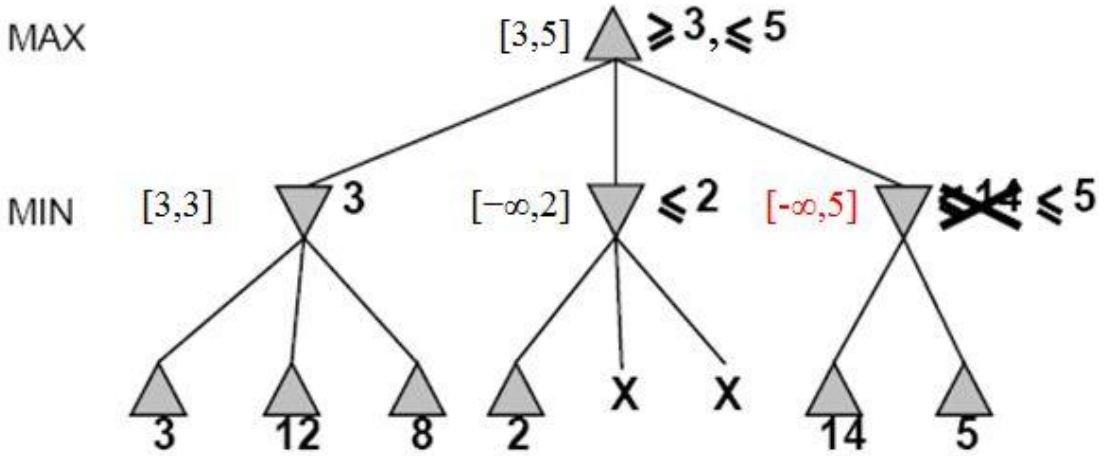


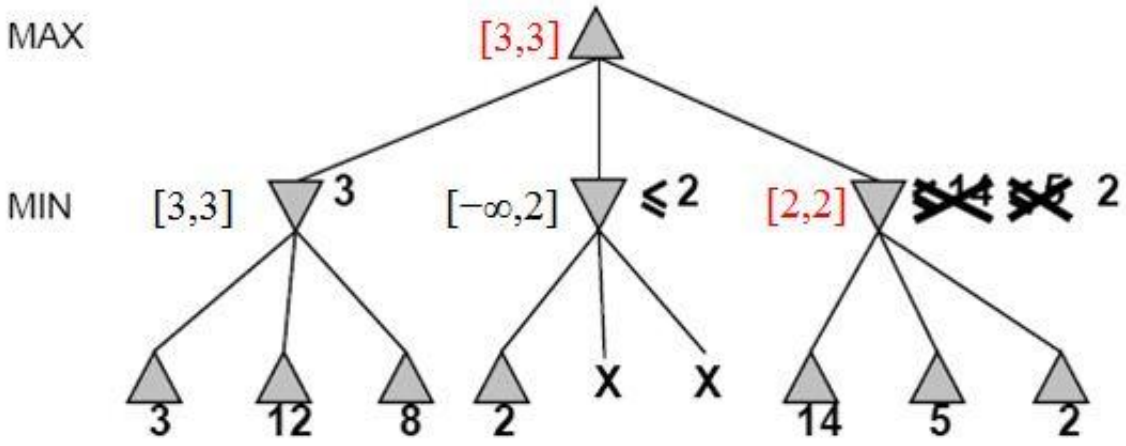

MAX
[-∞,+∞]
≥3
MIN
[-∞,3]
≤3
3
12

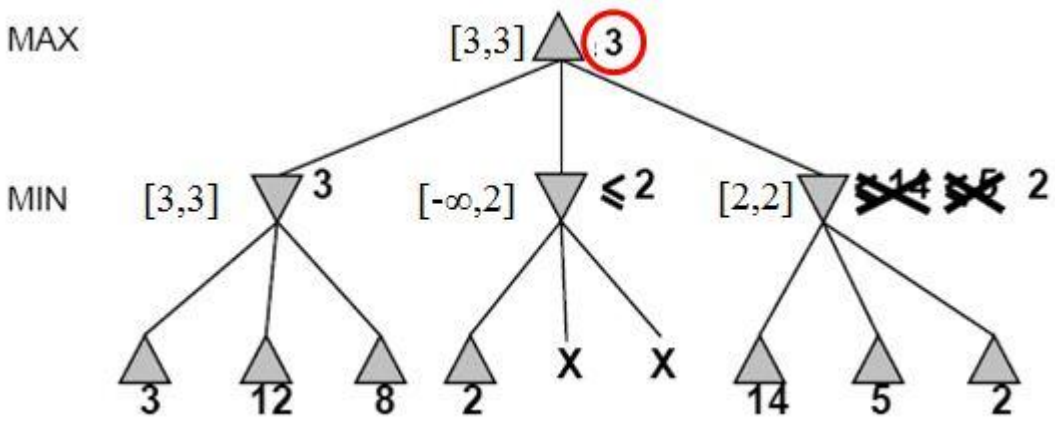

MAX
[3,+∞]
≥3
MIN
[3,3]
3
3
12
8


MAX
[3,+∞]
≥3
این گره برای Max مناسب نیست
MIN
[3,3]
3
[-∞,2]
≤2
3
12
8
2
X
X


MAX
[3,14]
≥3,≤14
MIN
[3,3]
3
[-∞,2]
≤2
[-∞,14]
≤14
3
12
8
2
X
X
14


MAX
[3,5]
≥3,≤5
MIN
[3,3]
3
[−∞,2]
≤2
[-∞,5]
≤5
3
12
8
2
X
X
14
5


MAX
[3,3]
MIN
[3,3]
3
[−∞,2]
≤2
[2,2]
2
3
12
8
2
X
X
14
5
2



بازیهای قطعی با اطلاعات ناقص

معایب الگوریتم های پیشین

الگوریتم minimax کل فضای جست و جوی بازی را تولید میکند.

الگوریتم آلفا-بتا با وجود هرس درخت، اما کل مسیر حالتهای پایانه، حداقل برای بخشی از فضای حالت، باید جست و جو شود.

- این عمق عملی نیست، زیرا حرکات باید در زمانی معقول انجام شود.

شانون(۱۹۵۰) :

برای کمتر شدن زمان جست و جو و اعمال تابع ارزیابی اکتشافی به حالتهای جستجو، بهتر است از گره های غیر پایانه به گره های پایانه پرداخته شود.

در شانون, minimax و آلفا-بتا به دو روش بطور متناوب عمل میکنند.

جایگزینی تابع سودمندی با تابع ارزیابی اکتشافی بنام EVAL

- تخمینی از سودمندی موقعیت ارائه میکند

جایگزین تست پایانه با تست توقف

- تصمیم میگیرد EVAL چه موقع اعمال شود.

تابع ارزیابی اکتشافی EVAL

تابع ارزیابی، ارائه تخمینی از سودمندی مورد انتظار بازی از یک موقعیت خاص

- توابع اکتشافی، تخمینی از فاصله تا هدف را بر میگرداندند.

اغلب توابع ارزیابی، **خواص** گوناگونی از حالتها را محاسبه میکنند.

- خواص روی هم رفته، کلاسهای هم ارزی یا دسته های مختلفی از حالتها را تعریف میکنند.
- حالتهای هر دسته، برای تمام خواص مقدار یکسانی دارند.

هر دسته حاوی چند حالت است که

- موجب برنده شدن
- موجب رسم شدن
- منجر به باختن

تابع ارزیابی نمیداند کدام حالت منجر به چه چیزی میشود، اما میتواند مقداری برگرداند که تناسب حالتها را با هر نتیجه نشان دهد.

مثال تابع EVAL :

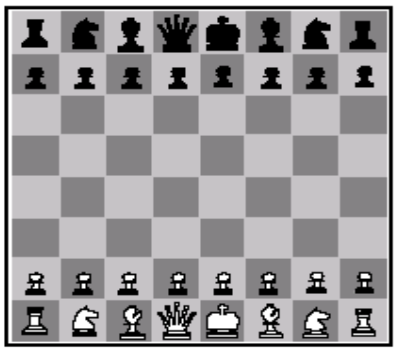
اغلب توابع ارزیابی, مقدار عددی جداگانه ای برای هر خاصیت محاسبه، سپس آنها را ترکیب میکنند تا مقدار کل بدست آید.

مثال در تابع بازی شطرنج:

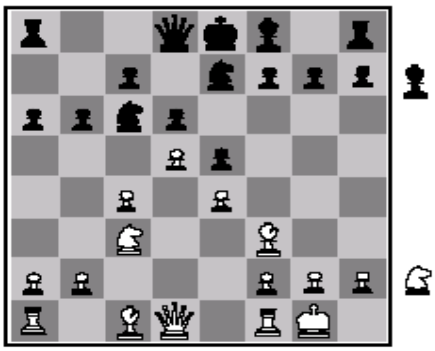
تعداد هر نوع قطعه در صفحه

مقادیر آن قطعات(۱ برای پیاده، ۳ برای اسب یا فیل،۵ برای رخ و ...)

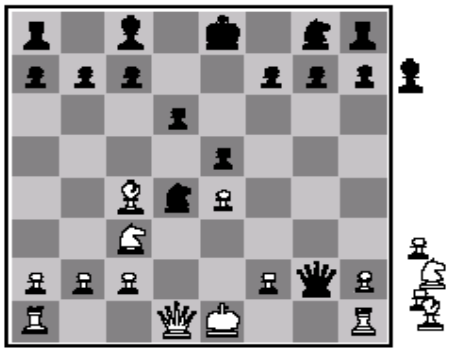
$$Eval(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$$



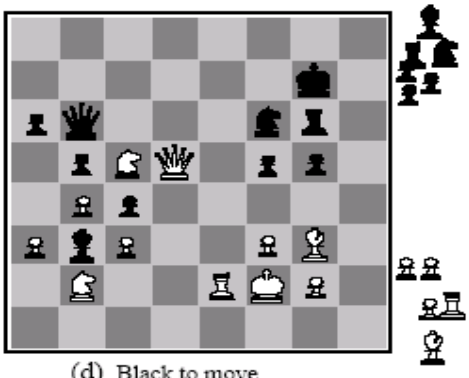
(a) White to move
Fairly even



(b) Black to move
White slightly better



(c) White to move
Black winning



(d) Black to move
White about to lose

ارزیابی تابع EVAL از مقدار پیروزی در دو موقعیت کاملا متفاوت

الف) سیاه، مزیت اسب و دو پیاده دارد و بازی را میبرد.

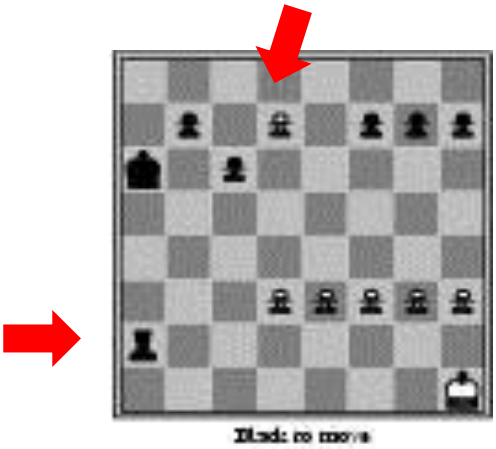
ب) پس از اینکه سفید، وزیر را در اختیار میگیرد، سیاه میبازد.

اثر افق

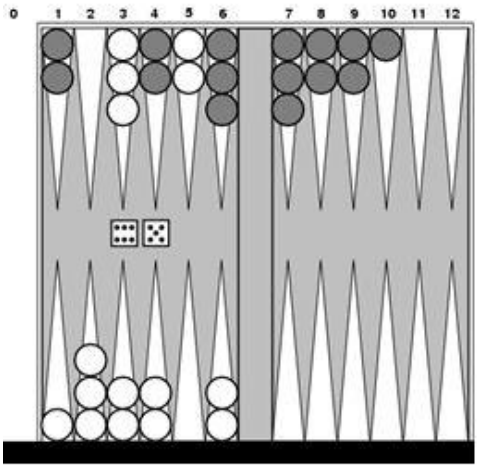
وقتی بوجود می آید که برنامه با اثری از رقیب مواجه شود که منجر به خرابی جدی گشته و اجتناب پذیر است.

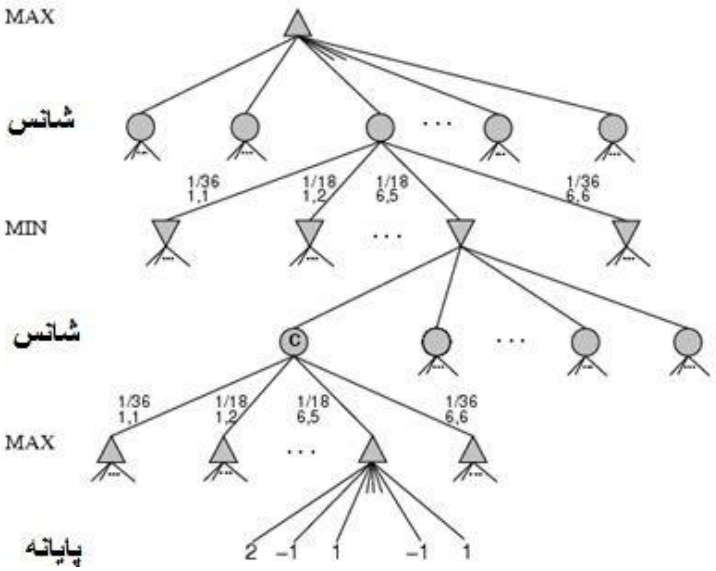
- مثال: شکل زیر؛

سیاه در اصل جلوست، اما اگر سفید پیاده اش را از سطر هفتم به هشتم ببرد، پیاده به وزیر تبدیل میشود و موقعیت برد برای سفید بوجود می آید.



بازیهایی که حاوی عنصر شانس هستند





در جلسه آینده به مبحث عامل های منطقی میپردازیم.

پایان