بخش پـنـجم

تـئـوري بـا زي

جستجوي رقابتي (تئوري بازي)

از مباحث ويژه هوش مصنوعي است اگرچه بازيهايي كه در هوش مصنوعي مطرح ميشوند بسيار پيچيده هستند اما تئوري درسي بيشتر به گروهي از بازيهاي خاص نظير شطرنج يا بازي x-o (tic-tac-toe) تعلق دارد اين بازيها دو نفره هستند و بازي تيمي در آن مطرح نيست. نوبت بازي چرخشي است يعني براساس قوانين بازي نوبت عوض ميشود در اين بازيها تعريف دقيقي براي برد، باخت و احتمالاً تساوي وجود دارد. بازيكن همواره در تمامي شرايط بازي تسلط دارد يعني هيچ حركتي نيست كه از چشم بازيكن دور بيفتد. مهمترين محدوديت حذف عوامل شانسي نظير سكه، تاس يا ورق ميباشد. در اين تئوري بازيكن خودي به نام max و بازيكن رقيب به نام min خوانده ميشود در تئوري بازي معمولاً گراف فضاي حالت را به صورت درخت در نظر ميگيريم. بازيهايي مثل شطرنج داراي درخت بسيار بزرگي هستند و ميتوان حدود ۳۵ به توان ۱۰۰ گره براي بازي توليد كنند. در اين شرايط امكان توسعه تمام درخت بازي و بررسي تمامي مسيرهاي برد و باخت وجود ندارد تنها راه حل آن است كه درخت بازي را تا سقف معيني بسط داده و سپس در همان سطح بينابين تصميمگيري شود كه كدام مسير بهتر است.

براي اين امر لازم است كه تابعي داشته باشيم كه يك گره از درخت بازي را دريافت نموده و شانس برد آن وضعيت را، به صورت عددي، معين كند. اين تابع را تابع امتياز يا payoff يا تابع سودمندي utility ميناميم. در واقع اين تابع شبيه به تابع هيوريستيك است اما داراي يك تفاوت اساسي با تابع هيوريستيك خواهد بود. در

بازي مهم نيست كه شما در كوتاهترين راه حل بازي را به اتمام برسانيد بلكه مهم اين است كه بازي را ببريد. براي اين منظور تابع امتياز بايد برآيند شانس برد شما را بر عليه شانس برد رقيب نشان دهد.

مثلاً بازي x-o را در نظر بگيريد در اين بازي هدف اشغال يك سطر يا ستون و يا يك قطر توسط هر يك از دو رقيب است. بديهيست تابع امتياز مناسب، بايد شانس برد مهره حريف را از شانس برد مهره خودي كم كند.

اين بدان معني است كه شانس برد خودي بيشتر است اگر حاصل، مقداري مثبت باشد. اما اگر عدد منفي باشد يعني شانس برد رقيب بيشتر است و اگر صفر باشد يعني طرفين در وضعيت متعادلي قرار دارند.

در اين جا يك فرض ساده شده است و آن اين است كه رقيب براي تصميم‌گيري از الگوريتمي مشابه الگوريتم خودمان استفاده مي‌كند.

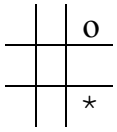
در واقع اين مسئله درست نيست و معمولاً رقيب ممكن است طرز تفكر و سبك خاصي را براي اين بازي داشته باشد. اما در اين بحث وارد مقوله شناسايي الگوريتم هوشمند رقيب نخواهيم شد. پس كافي است تابع امتياز را براي اين بازي تعريف كنيم تابع پيشنهادي مي‌تواند تعداد سطرها، ستون‌ها و يا قطرهايي باشد كه توسط مهره خودي يا رقيب شانس اشغال داشه باشد.

تابع امتياز = امتياز مهره x - امتياز مهره o

تابع امتياز =۴-۲=۲

	o	o
	*	
*		

تابع امتیاز = ۵-۵=۰



استراتژي minimax

اين الگوريتم بر مبناي جستجوي اول عمق محدود است. درخت بازي در اين الگوريتم تا عمق معيني مانند L بسط داده مي‌شود. سپس به كمك تابع امتياز، برگ‌هاي درخت، امتياز خود را خواهند گرفت. در اين درخت توليد شده دو گروه گره داريم كه هر گره يك سطح كامل را اشغال مي‌كند. در سطح گره‌هاي max نوبت حركت با ما است در صورتي كه در سطح گره‌هاي Min نوبت حركت با رقيب ما است. نام اين الگوريتم هم از اين دو نوع گره برداشته شده است. هر گروه max به دنبال يافتن حداكثر امتياز است در صورتي كه گره‌هاي رقيب در صدد كمينه ساختن مقدار تابع ارزيابي هستند.

اين استراتژي را مي‌توان به صورت ۵ مرحله زير خلاصه كرد:

۱ـ توليد درخت بازي تا عمق معين شده (به صورت اول عمق)

۲ـ تعيين امتياز گره‌هاي برگي درخت توليدي از طريق تابع امتياز

۳ـ از برگ به سوي ريشه، سطح به سطح برمي‌گرديم. اگر در سطح max باشيم بين فرزندان بيشترين امتياز به پدر

نسبت داده مي‌شود و اگر در سطح min باشيم کمترني مقدار به پدر نسبت داده مي‌شود.

۴ـ عمليات مرحله ۳ از گره‌هاي برگي شروع شده و سطح به سطح تا ريشه ادامه پيدا مي‌کند.

۵ـ ريشه در نهايت عددي را پيدا مي‌کند که با مقدار يکي از فرزندان برابر است. پس حرکت پيشنهادي بعدي از ريشه به سمت همان گره خواهد بود.

نکته ۱: اگر عمق درخت حداکثر a و b حرکت قانوني در هر نقطه وجود داشته باشد پيچيدگي زماني $o(b^m)$ خواهد بود. از آنجا که جستجو به روش عمقي است پس حافظه مورد نياز آن تابع خطي از m و b مي‌باشد.

نکته ۲: هرچقدر عمق بیشتری را بسط داده و بعد تابع امتیاز را بروی برگ ها اعمال کنیم، حرکت بهتری را انجام خواهیم داد.

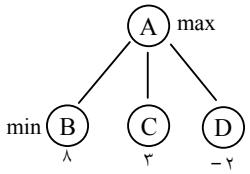
مثال: فرض کنید در درخت زیر، نوبت بازی A است(خودی) و برای بررسی وضعیت بازی، یک سطح بعد را تولید میکنیم

فرض مقادیر تولیدی تابع امتیاز، مقادیری بین [۱۰ و ۱۰-] است که:

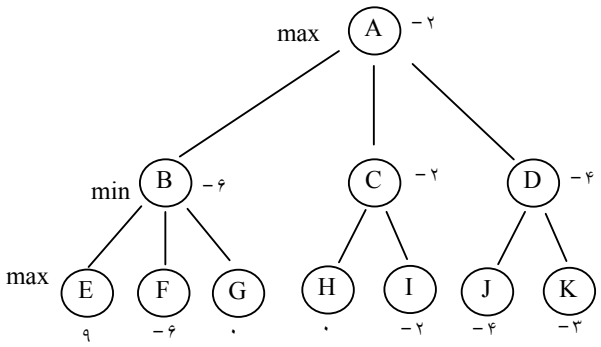
۱۰ برد

۱۰- باخت

۰ مساوي



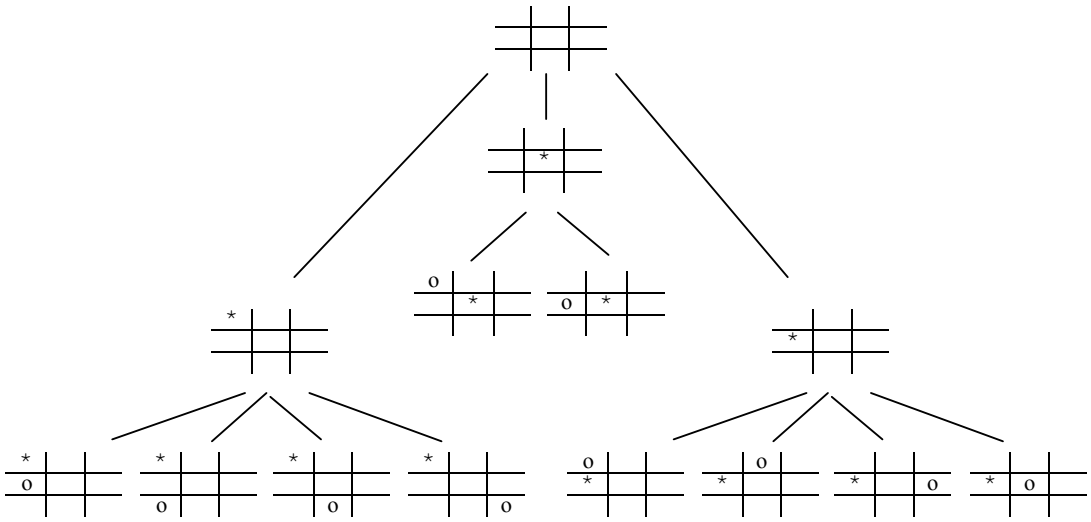
ابتدا یک سطح بعد را تولید نموده (نودهای B،DوC) و گره های برگی را در تابع امتیاز قرار میدهیم. با توجه به مقادیر بدست آمده،نود A که یک نود MAX است، نود B را بعنوان حرکت بعدی انتخاب میکند. از آنجا که عمق کمی را بسط داده ایم، یک سطح دیگر را بسط میدهیم:



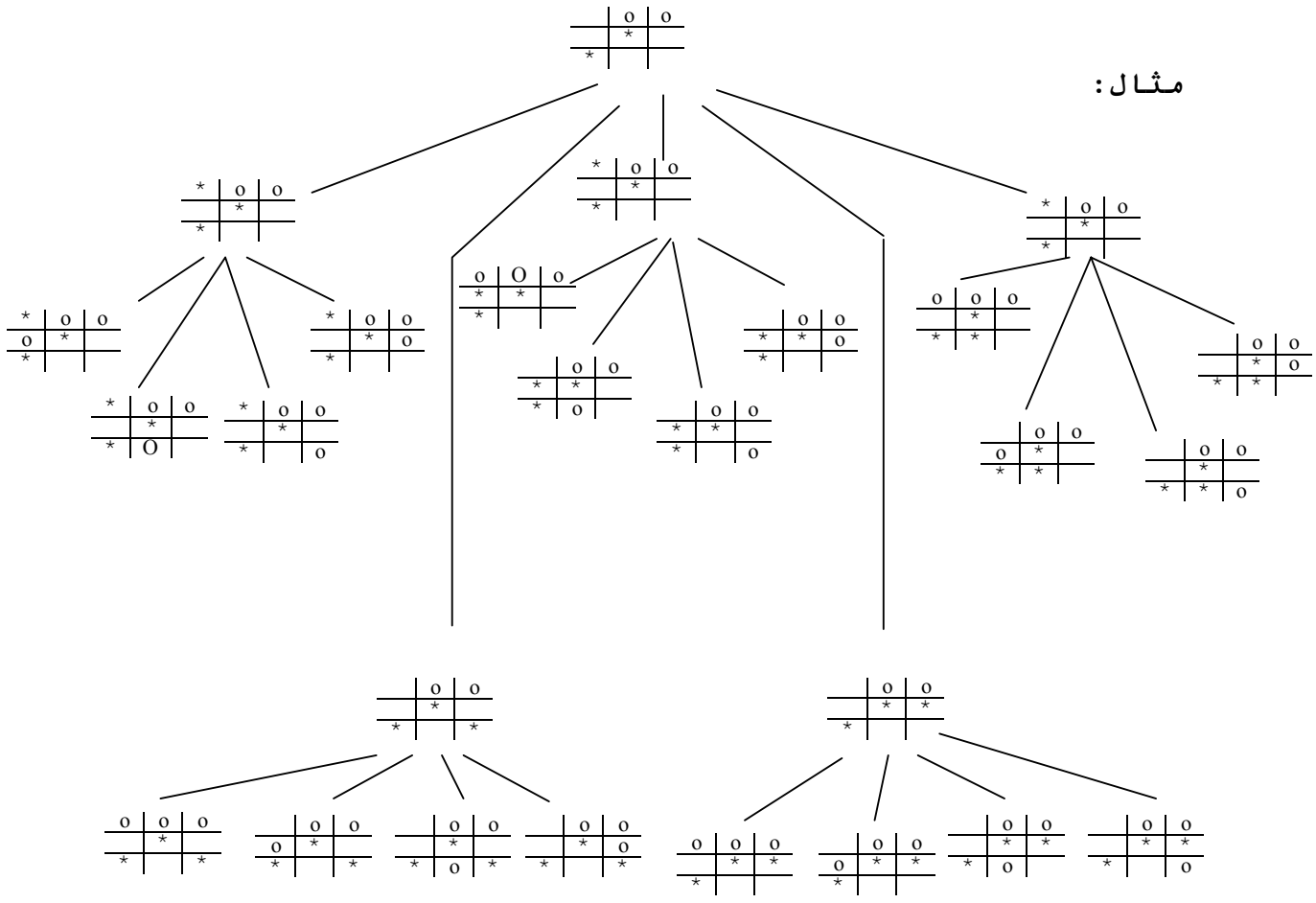
در این حالت، تا دو سطح بعدی درخت را بسط داده و امتیاز گره های برگی را محاسبه میکنیم. در این وضعیت حرکت ما باید به سمت نود C خواهد بود. مسلماً اگر ما حرکت B را انجام داده باشیم رقیب ما از بین حرکت‌هاي G،FوE نود F را انتخاب مي‌کند چرا که هدف رقیب کمینه ساختن مقدار تابع ارزیابي (امتیاز) است. پس با انتخاب حرکت B در وضعیت بدتري قرار مي‌گیریم. حریف همیشه از بین فرزندان خود کمترین مقدار را انتخاب مي‌کند ولي لایه max

همیشه بیشترین مقدار را به خود میگیرد پس در موقعیت A بهترین انتخاب C است.

مثال:



مـثـال :



استراتژي قطع (هرس) $\alpha - B$: (Alpha - Beta Pruning)

اين تكنيك به روي يك درخت استاندارد minimax انجام مي‌شود. در اين روش از بسط دادن شاخه‌هايي كه احتمال رسيدن به جواب بهينه در آنها وجود ندارد جلوگيري مي‌كنيم. در اين روش از جستجوي كل درخت بازي اجتناب نموده و قسمتي از درخت بازي را هرس مينمائيم.

در اين استراتژي هر گره max مقدار موقتي به نام α و هر گره min مقدار موقتي به نام β را به خود مي‌گيرد كه اين مقادير در حين جستجو در درخت توليد شده و ممكن است تغيير پيدا كند. اين مقادير به طور موقت از فرزندان به پدر نسبت داده خواهد شد. دو اصل زير همواره برقرار است:

۱ـ مقدار αي گره‌هاي max هيچ گاه كاهش پيدا نمي‌كند.

۲ـ مقدار βي گره‌هاي min هيچگاه افزايش پيدا نمي‌كند.

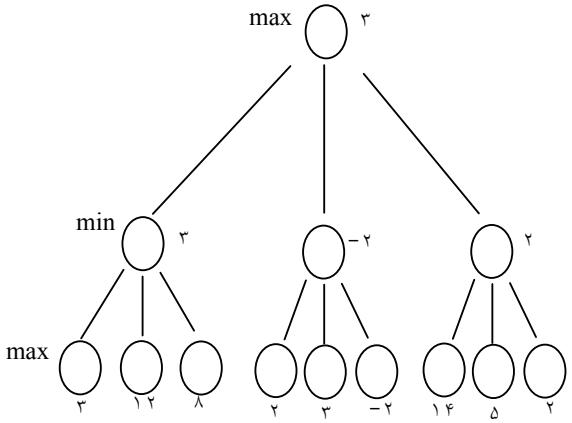
در اين روش دو قانون زير را اعمال مي‌كنيم:

۱ـ چنانچه αي يك گره از βي پدرش بيشتر باشد ساير فرزندان آن گره پدر را قطع مي‌كنيم.

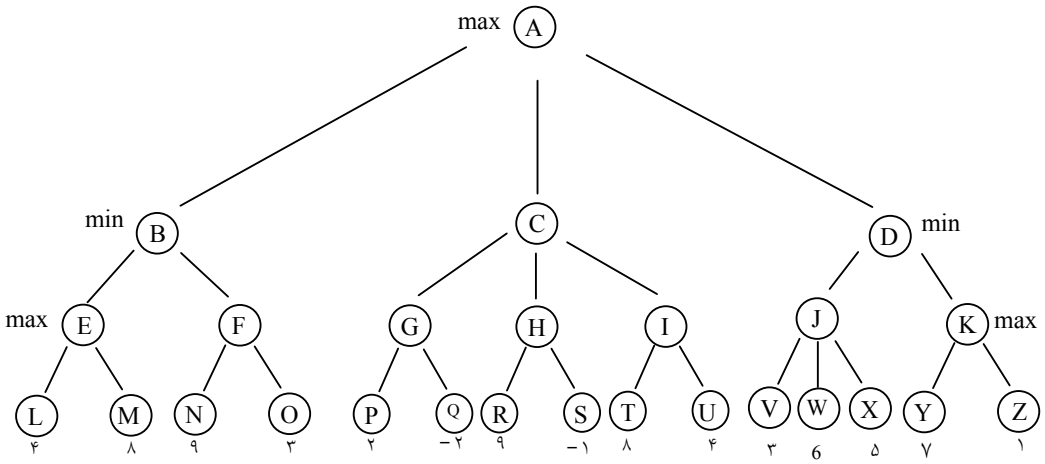
۲ـ چنانچه βي يك گره از αي پدرش كمتر باشد ساير فرزندان آن گره پدر را قطع مي‌كنيم.

توجه: ابتدا نودهاي max مقدار $-\infty$ و نودهاي min مقدار $\beta = +\infty$ را دارد در طول مسير α افزايش و β كاهش مي‌يابد.

مـثـال :



مـثـال


max
A
min
B
C
D
min
max
E
F
G
H
I
J
K
max
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
۴
۸
۹
۳
۲
-۲
۹
-۱
۸
۴
۳
6
۵
۷
۱

تمرین ۱

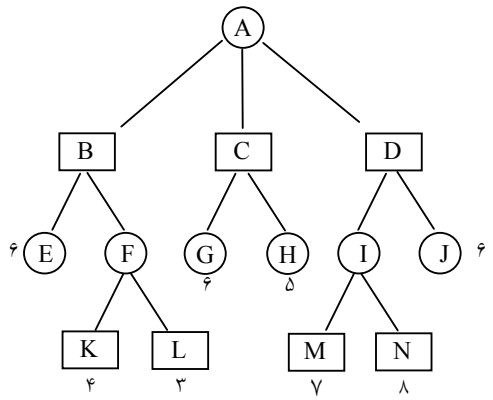
اگر با استفاده از روش جستجوي minimax درخت جستجوي مقابل پیمایش شود با استفاده از روش هرس α و β کدامیك از گره‌هاي این درخت ملاقات نخواهد شد (دوایر معرف گره‌هاي min و مربع‌ها معرف گره‌هاي max و اعداد کنار گره‌هاي برگ معرف ارزش این گره‌ها است) (کنکور ۸۳).

۱ـ $\{L,H,N\}$

۲ـ $\{L,N,j\}$

۳ـ $\{L,H,j\}$

۴ـ $\{H,N,j\}$



تمرین ۲

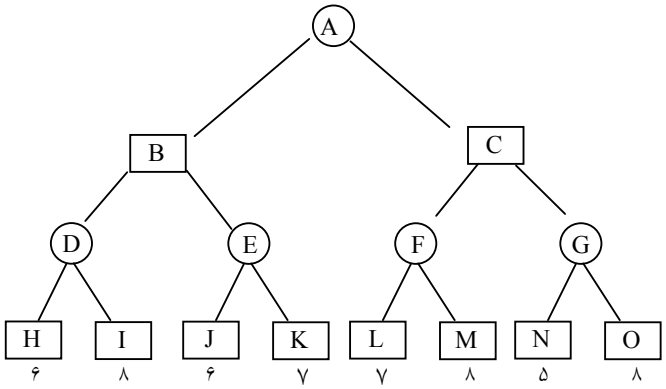
درخت مقابل در اثر جستجوي minimax ایجاد شده است گره‌هاي دایره گره min و گره مربع گره max هستند اعداد زیر گره‌هاي برگ ارزش آنها را نشان مي‌دهد. در صورت استفاده از روش هرس α و β کدامیك از گره‌هاي این درخت جستجو نخواهد شد (کنکور ۸۱).

۱) گره‌هاي K و M

۲) گره‌هاي K و M و O

۳) گره‌هاي K و O و N و G

۴) کلیه گره‌هاي جستجو خواهند شد



تمرین ۳

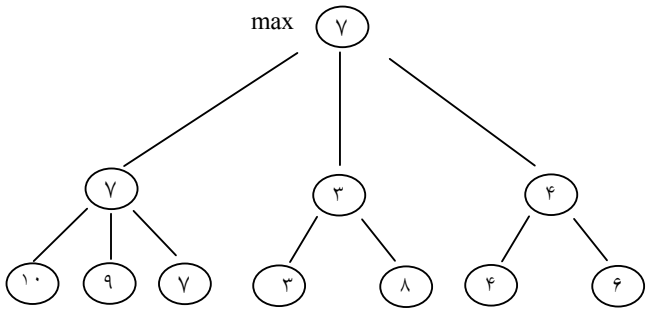
در درخت بازي زير (i,j) نمايش ارزش عنصر j ام از چپ به راست در سطح i ام است. مثلاً (۲ و ۳) نمايش عنصر دوم از سطح سوم با ارزش ۹ مي‌باشد. چه عناصري شامل $\alpha.\beta$ *pruning* مي‌شود.

۱ـ (۳ و ۳)

۲ـ (۵ و ۳) و (۷ و ۳)

۳ـ (۳ و ۲) و (۲ و ۲)

۴ـ (۶ و ۳) (۴ و ۳) و (۲ و ۳) و (۱ و ۳)



تمرین ۴

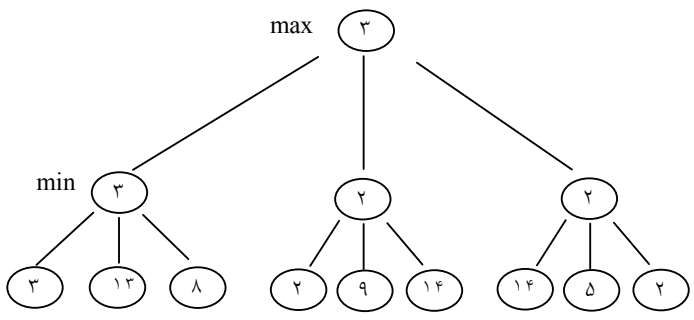
درخت minimax زير را در نظر بگيريد اگر از قطع α و β استفاده كنيد چند اتصال قطع مي‌شود.

۱) ۲ اتصال

۲) ۴ اتصال

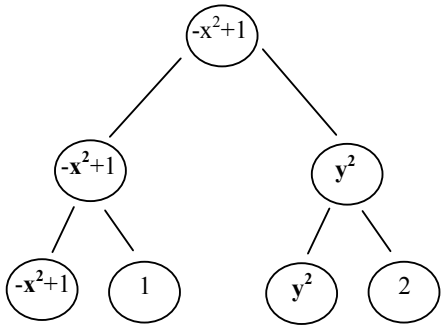
۳) ۶ اتصال

۴) هيچكدام



تمرین ۵

درخت بازي‌هاي دو نفره زير را در نظر بگيريد در صورتي كه بازي به صورت minimax انجام گيرد مقادير مجاز براي x و y چيست؟ x و y اعداد حقيقي و در آغاز بازي نوبت با max است.



تمرین ۶

اگر از الگوريتم minimax براي بازي زير استفاده كنيم ثابت كنيد كه x نقشي در تصميم‌گيري نهايي ندارد و اگر:

الف ـ n<m و نوبت بازي با max باشد.

ب ـ n>m و نوبت بازي با min باشد.