Departamento de Sistemas Informáticos y Computación
Escuela Técnica Superior de Ingeniería Informática
Universitat Politècnica de València

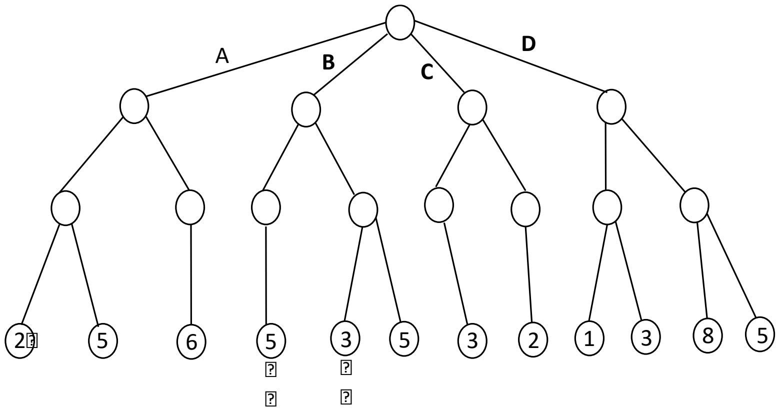
SOLUTIONS TO COLLECTION OF EXERCISES
INTELLIGENT SYSTEMS

Block 1: Adversarial Search

November 2022

MULTIPLE CHOICE QUESTIONS

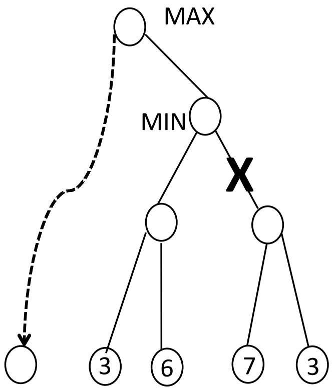
1) Given the following game search space, which is the best move for the root MAX node if we apply an alpha-beta algorithm?



A. Branch A
B. Branch B
C. Branch C
D. Branch D

---

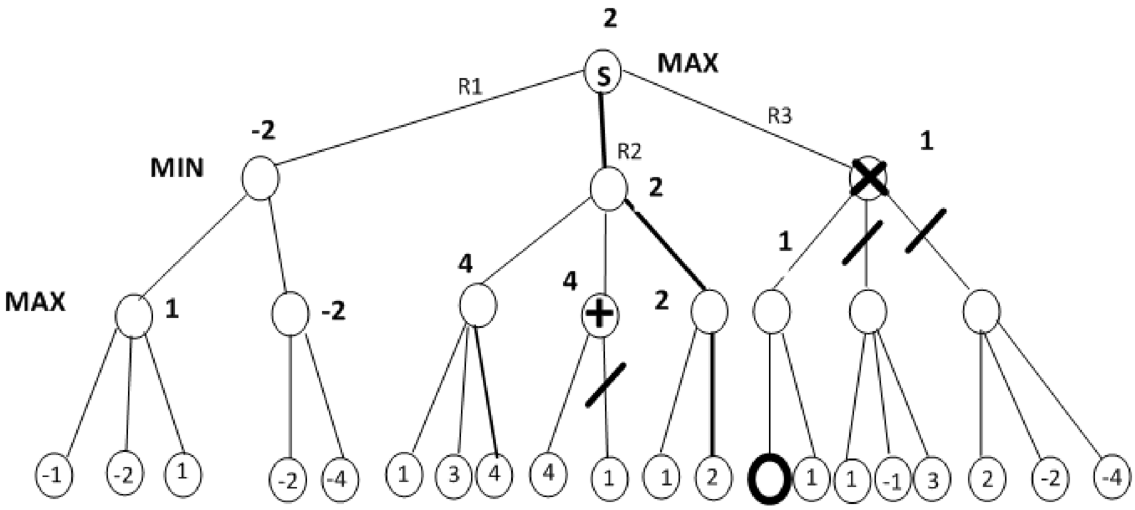
2) Given the following partial tree of an alpha-beta algorithm, which provisional backed up value should the node MAX have for that the cut-off is produced?



A. 7
B. Higher or equal than 6
C. Lower or equal than 6
D. Lower than 3

---

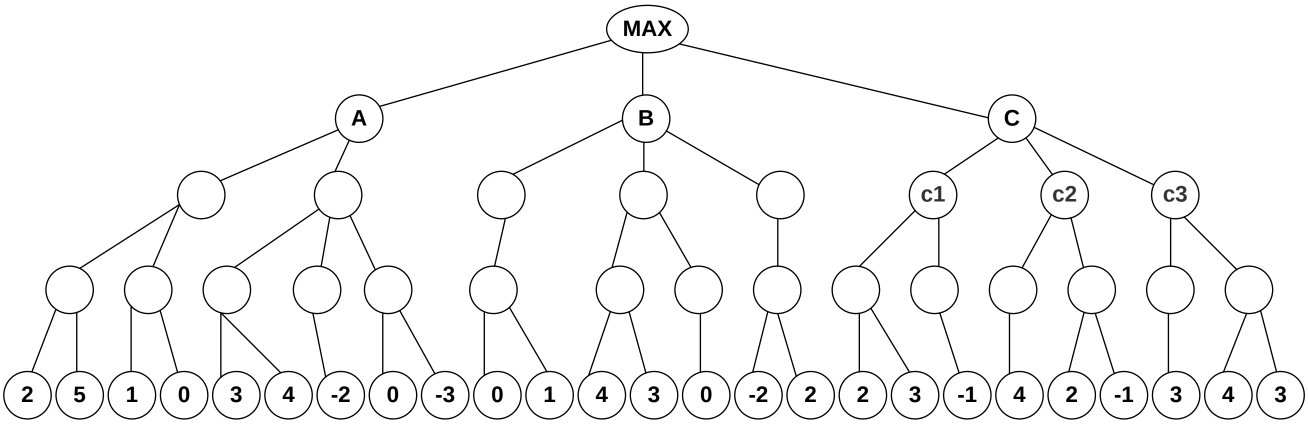
3) Given the search space of a game shown in the figure, which value should the terminal node in bold type have?



A. [-∞, 1]
B. 1
C. [2, +∞]
D. It is not possible to determine the value of the terminal node with the available data

---

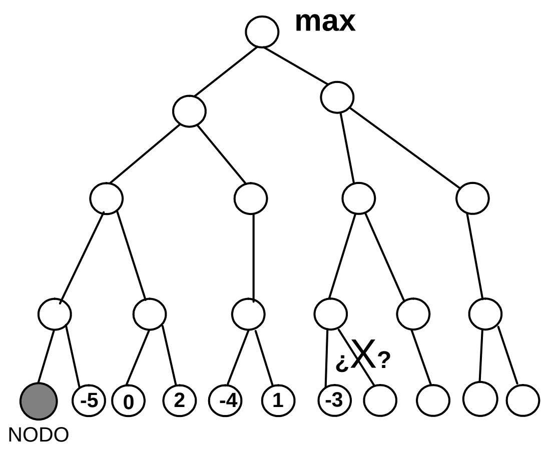
4) Given the game tree of the figure, which is the best move for MAX (root node) if we apply an alpha-beta procedure?



A. Branch A
B. Branch B
C. Branch C
D. Branch A or branch B

---

5) Given the game tree of the figure and assuming we apply an alpha-beta procedure:

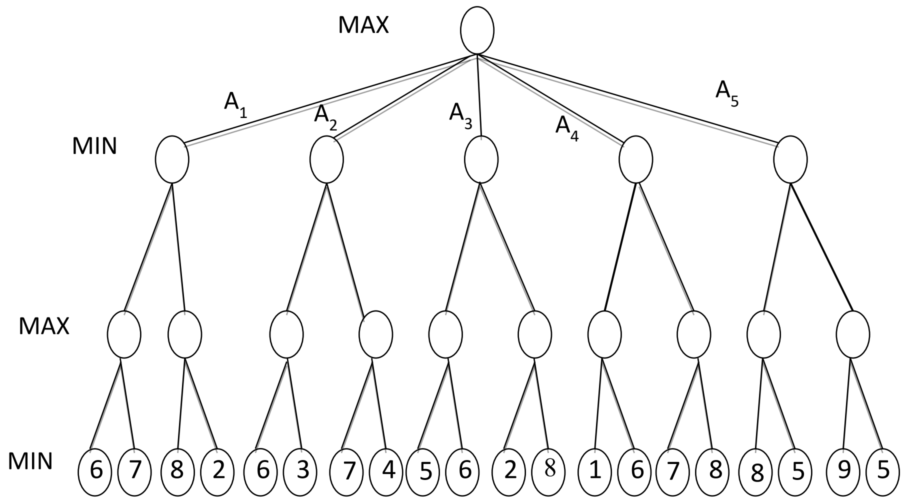


Which value should the shadowy node (NODO) have to provoke the cut-off shown in the figure?

A. Any value will produce the cut-off
B. Lower than -3
C. Higher than -3
D. The cut-off can never happen (or none of the above answers)

---

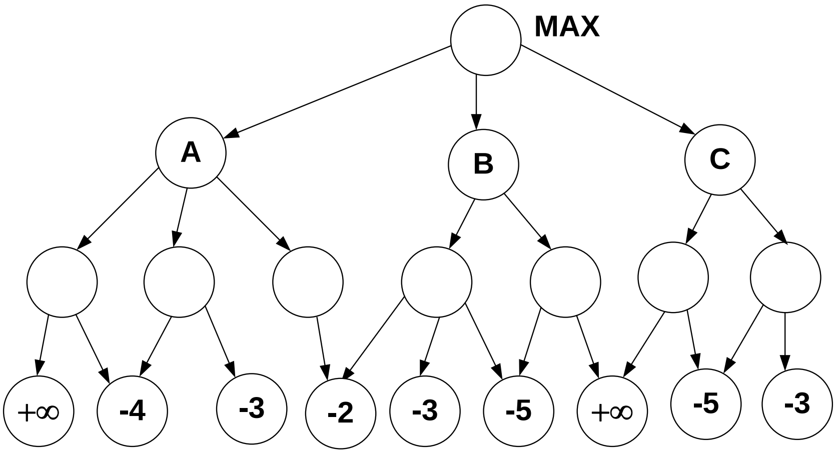
6) If we apply and α-β algorithm for the game tree of the figure, which is the best move for MAX?



A. Either branch A1 or A4
B. Branch A4
C. Branch A5
D. Either branch A1 or A2

---

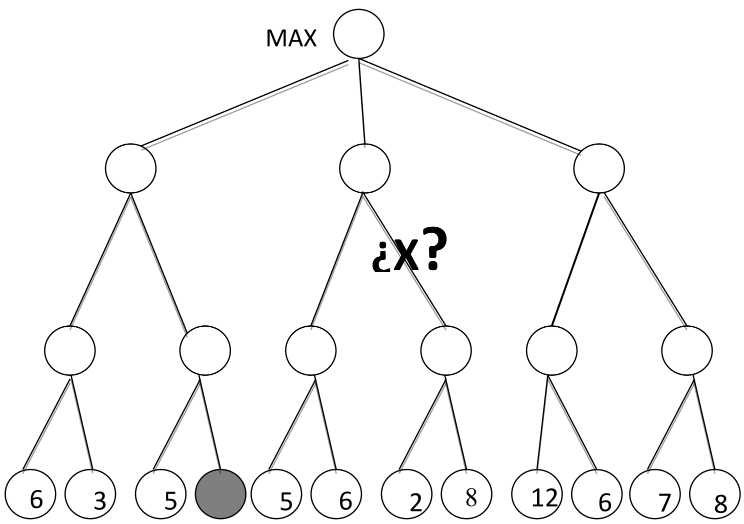
7) Given the game tree of the figure, where MAX is the initial player and assuming we apply an alpha-beta procedure, mark the CORRECT statement:



A. MAX will choose any of the three branches because all of them lead to a winning position for MAX
B. MAX will choose the branch A
C. MAX will choose the branch B
D. MAX will choose the branch C

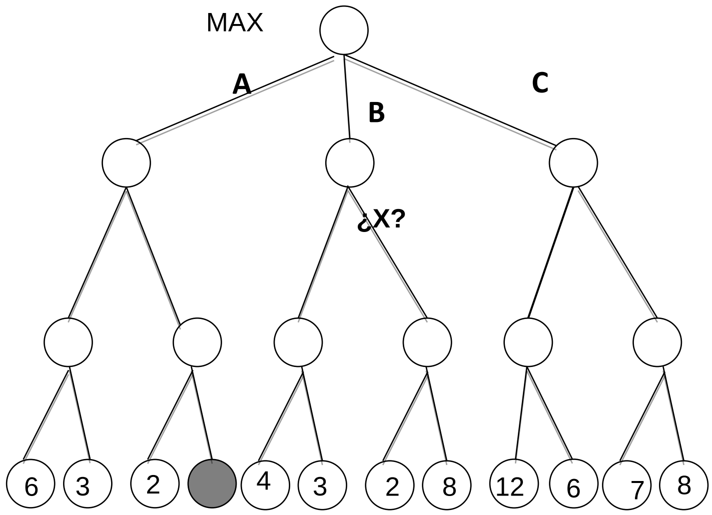
---

8) Assuming we apply an alpha-beta procedure to the game tree of the figure, which is the value that the shadowed node should have in order to get the cut-off of the figure?



A. Any value
B. A value lower than 6
C. A value higher or equal than 6
D. The cut-off would never be produced (none of the above answers)

9) Given the game tree of the figure and assuming we apply an alpha-beta algorithm:



which is the value that the shadowed node should have in order to get the cut-off of the figure?

A. Any value
B. A value lower than 3
C. A value higher or equal than 4
D. The cut-off would never be produced (or none of the above answers)

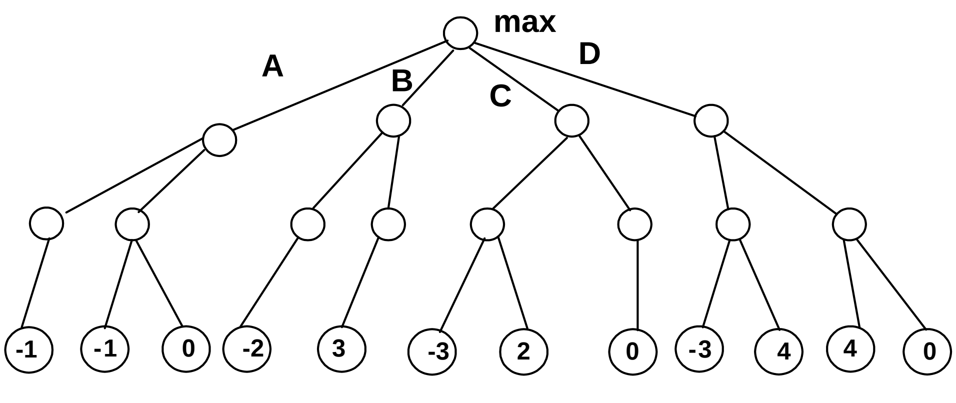
---

10) Given the game tree of the above figure and assuming the cut-off is produced, after applying an alpha-beta algorithm:

A. MAX will choose branch A
B. MAX will choose branch B
C. MAX will choose branch C
D. MAX will choose either branch A or B

---

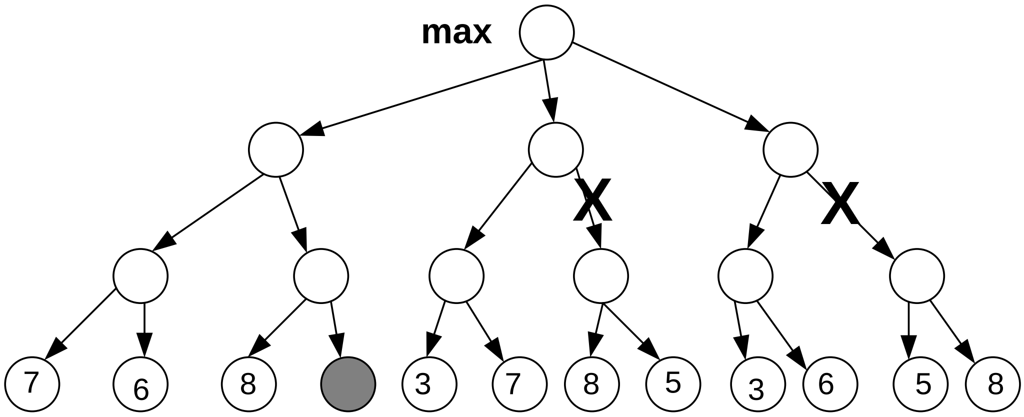
11) Assuming we apply an ALPHA-BETA algorithm on the game tree of the figure, how many nodes do we save with compared to MINIMAX?



A. 3
B. 4
C. 5
D. 6

---

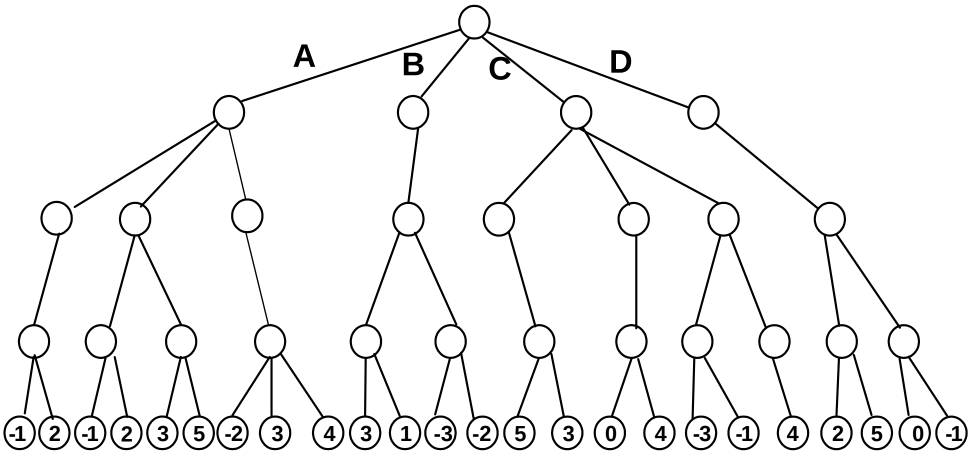
12) Assuming we apply an ALPHA-BETA procedure on the game tree of the figure, which value should the shadowed node have to provoke the shown cut-offs?



A. Equal or higher than 7
B. Equal or higher than 8
C. Equal or lower than 7
D. Any value would provoke the cut-offs

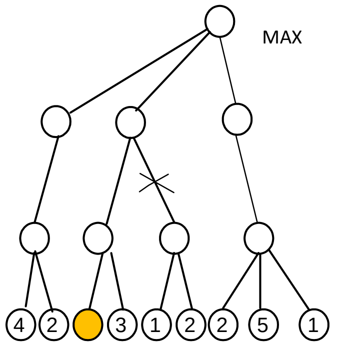
---

13) Which branch of the game tree of the figure below will be chosen if we apply the MINIMAX algorithm?



A. A
B. B
C. C
D. D

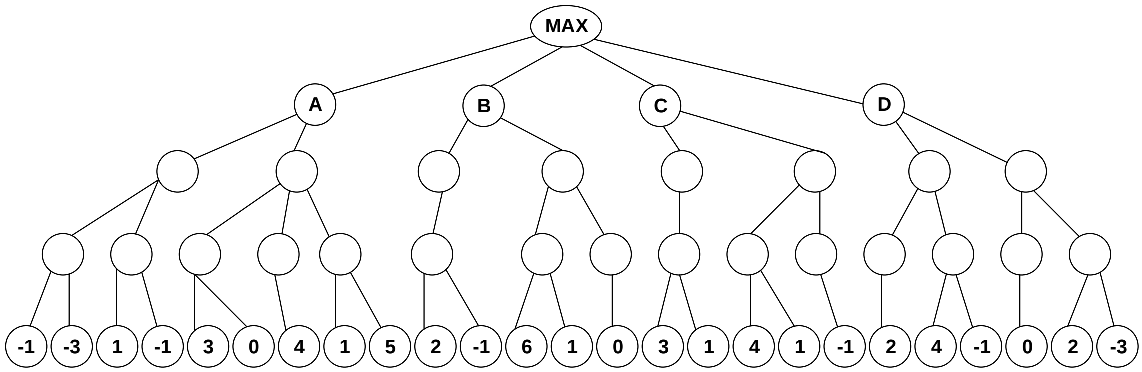
14) Which values should the shadowy node have so that the cutoff of the figure is always produced?



A. Any value in [-∞ 4].
B. Any value.
C. Any value in [4 +∞].
D. The cutoff can never happen.

---

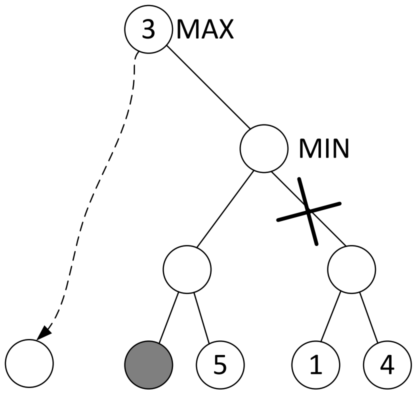
15) Show the branch that will be selected after applying the α-β pruning to the game tree of the figure:



A. A
B. B
C. C
D. D

---

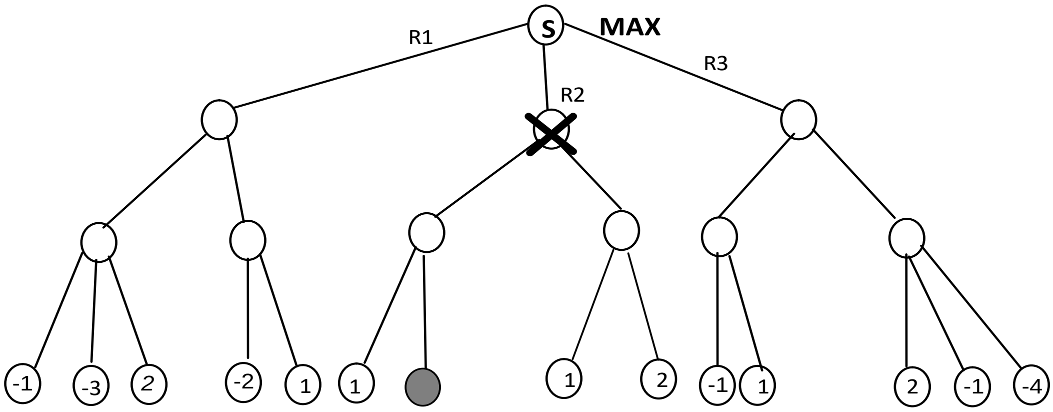
16) Which provisional value should the shadowy mode have in order to get the cutoff shown in the figure?



A. [-∞ , 2]
B. [-∞ , 3]
C. [-∞ , 5]
D. The cutoff is not feasible

---

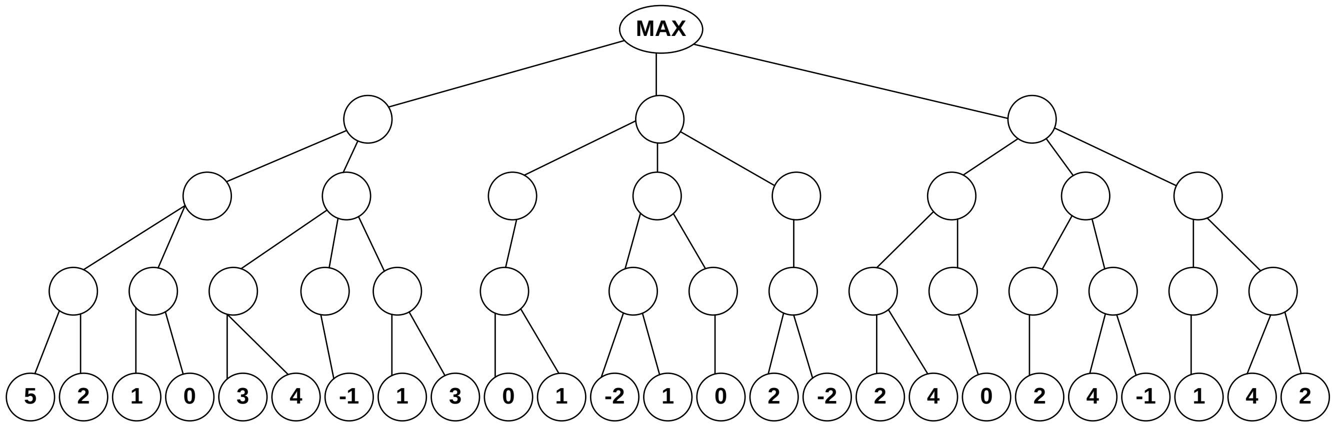
17) The figure below shows a game search tree. If we apply an alpha-beta procedure, which value should the shadowy node take on so that the cutoff in branch R2 is produced?



A. Any value in [- ∝, 1]
B. Any value in [1, +∝]
C. The shadowy node can only take on the value 1.
D. The cutoff of the figure cannot be produced.

---

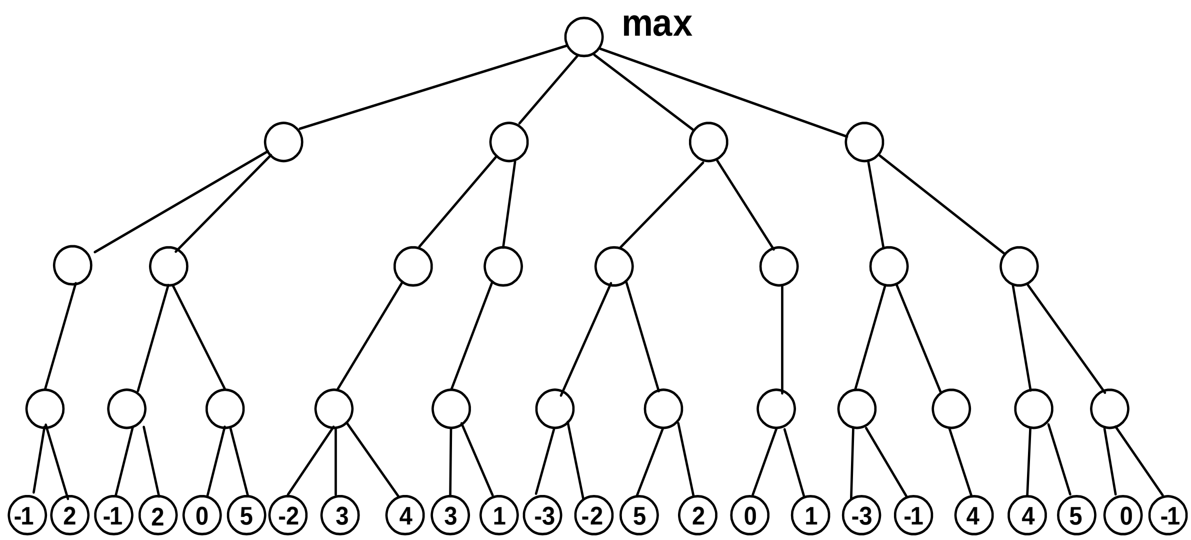
18) Given the below game search tree and assuming we apply an alpha-beta procedure, how many terminal nodes do not need to be generated?



A. 13
B. 15
C. 16
D. 17

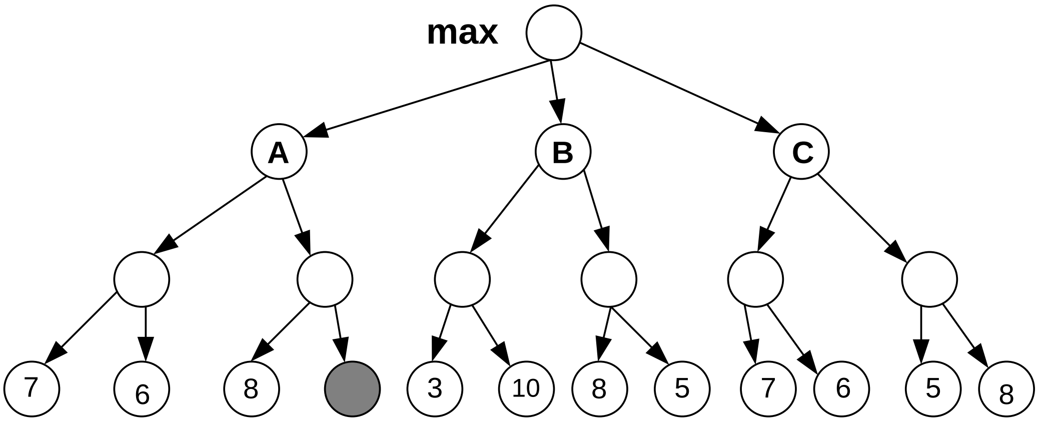
---

19) Show how many terminal nodes would be generated if we apply an alpha-beta procedure to the game tree of the figure:



A. 13
B. 12
C. 11
D. 14

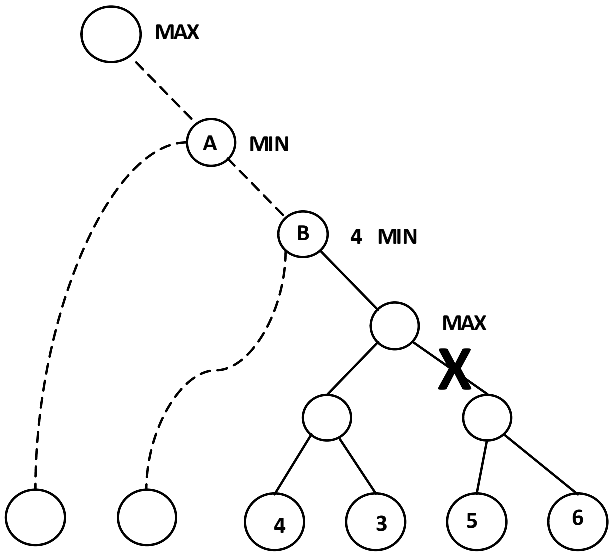
20) Given the game search tree below, and assuming we apply an alpha-beta procedure, show the **CORRECT** answer:



A. If the shadowed node takes on a value <=8, an alpha cut-off will be produced in both node B and node C.
B. If the shadowed node takes on a value >=10, an alpha cut-off will be produced in both node B and node C.
C. Regardless the value of the shadowed node, an alpha-beta cut-off will be always produced in node B.
D. Regardless the value of the shadowed node, an alpha-beta cut-off will be always produced in node C.

---

21) In the partial alpha-beta search of the figure below, the node B has a provisional backed-up value of 4. Which provisional value should node A have so that the effective cut-off shown in the figure is produced?



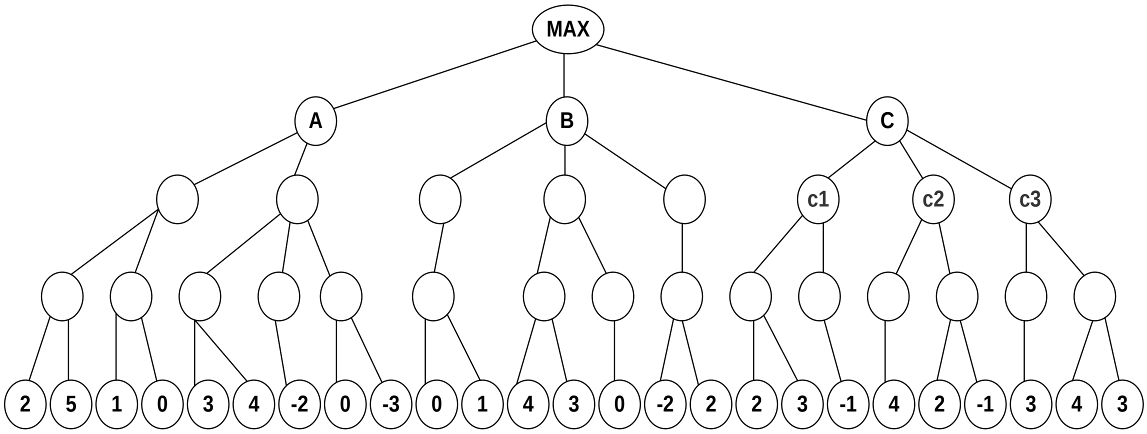
A. The cutoff of the figure can never be produced.
B. A value lower or equal than 3.
C. A value higher or equal than 3.
D. A value lower than 3.

22) Let n1 and n2 be the only two children nodes of a MAX node n in a game tree. We assume that node n1 is explored first and then node n2. Show the **CORRECT** answer:

A. The definite value of node n is the maximum value between the definite value of n1 and n2 only when n1 and n2 are terminal nodes.
B. When the value of n1 is backed up to its parent n, the node n can have a previously backed up value.
C. When the value of n1 is backed up to its parent n, a β cut-off can be produced in node n
D. None of the above answers is correct.

---

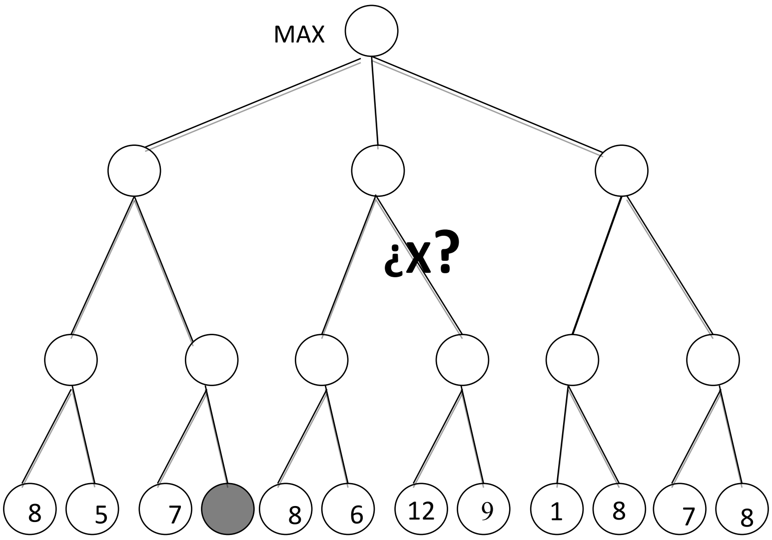
23) Which branch will be selected when we apply the α-β algorithm to the game tree of the figure?



A. Branch A
B. Branch B
C. Branch C
D. Branch A or B

---

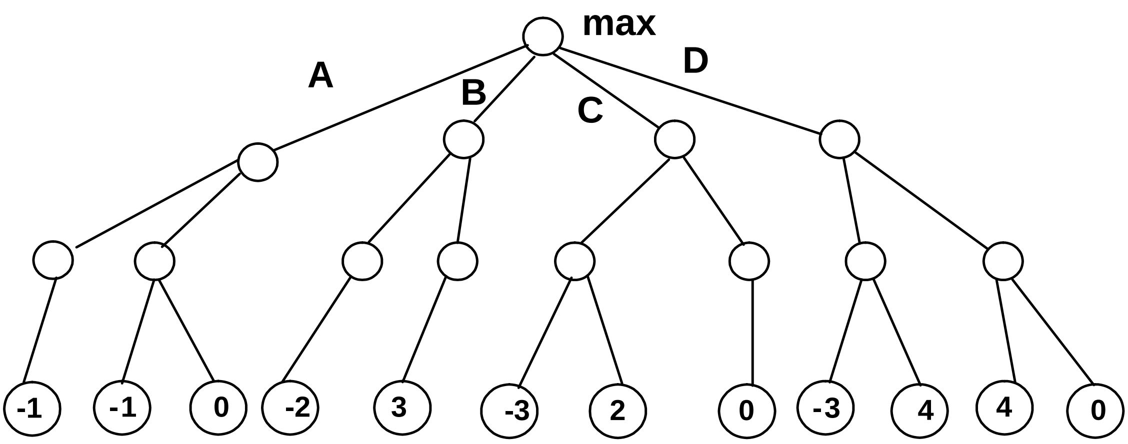
24) Given the game search of the figure below, and assuming we apply an alpha-beta procedure, show the value that the shadowed node should have in order to get the cut-off shown in the figure.



A. Any value of the shadowed node would produce the cut-off.
B. A value lower than 8.
C. A value equal or higher than 8.
D. The cut-off can never happen (or none of the above answers is correct).

---

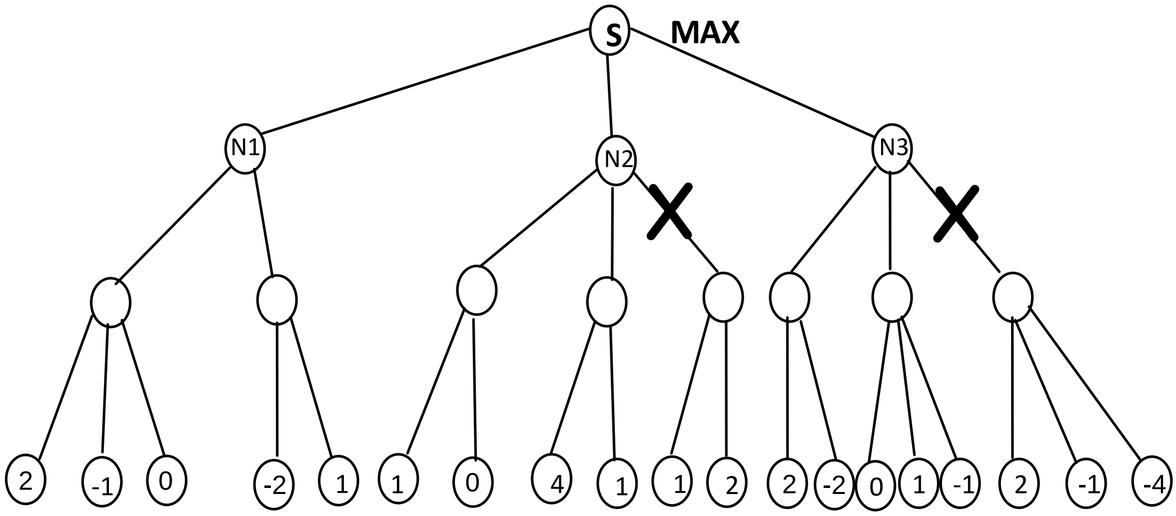
25) Given the game search space of the figure, if we apply an alpha-beta procedure, how many nodes are not needed to be generated compared to the application of a MINIMAX algorithm?



A. 3
B. 4
C. 5
D. 6

---

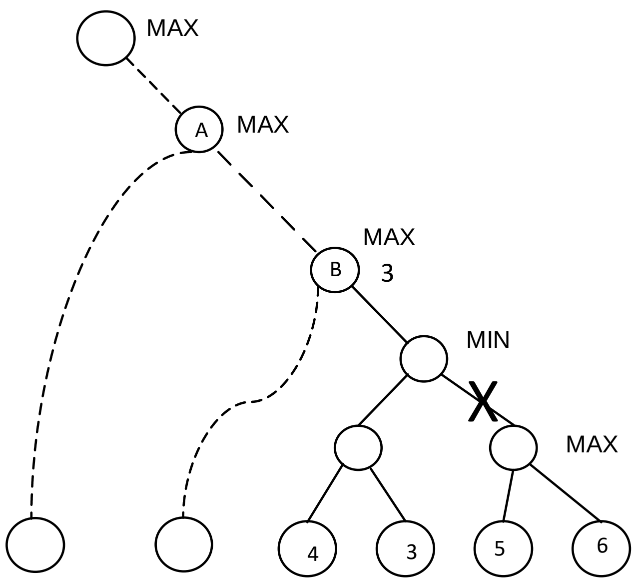
26) Given the game tree of the figure in which an alpha-beta procedure has been applied, show the **CORRECT** answer:



A. A cut-off is produced in node N2, which would also prune the middle branch of N2
B. A cut-off in N2 is not produced and so the right branch of N2 would not be pruned
C. A cut-off is produced in node N3, which would also prune the middle branch of N3
D. A cut-off in N3 is not produced and so the right branch of N3 would not be pruned

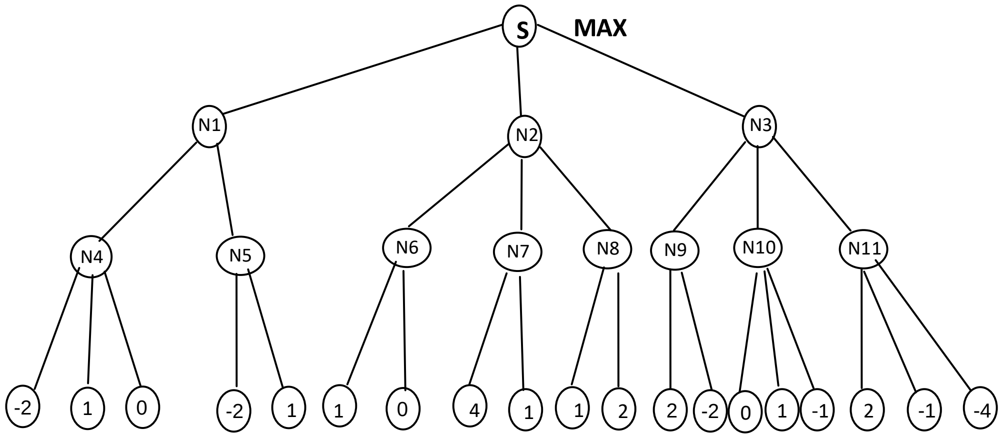
---

27) Given the partial game tree of the figure resulting from the application of an alpha-beta procedure, which provisional value should node A have so that it will provoke the cut-off the figure?



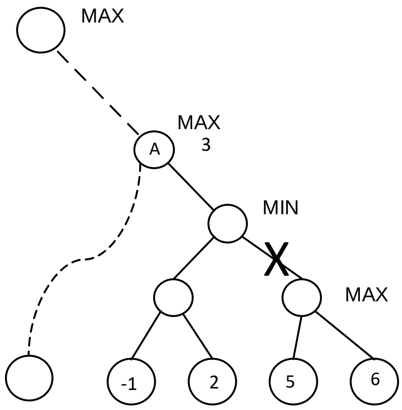
A. Every value within the interval [-α, 3] would provoke the cut-off
B. Every value within the interval [3, +∞] would provoke the cut-off
C. Every value within the interval [4, +∞] would provoke the cut-off
D. The cut-off can never happen

28) Assume we apply an alpha-beta procedure to the game space of the figure. Show the **CORRECT** answer:



A. A cut-off is produced in node N5
B. A cut-off is produced in node N6
C. A cut-off is produced in node N7
D. A cut-off is produced in node N10

29) Given the partial alpha-beta of the figure below, show the CORRECT answer:



A. The cutoff of the figure can never be produced.
B. If the value -1 would be replaced by 4 then the cutoff would be produced
C. If the value 2 would be replaced by 4 then the cutoff would be produced
D. None of the above answers is correct.

EXERCISES
(open answer questions)

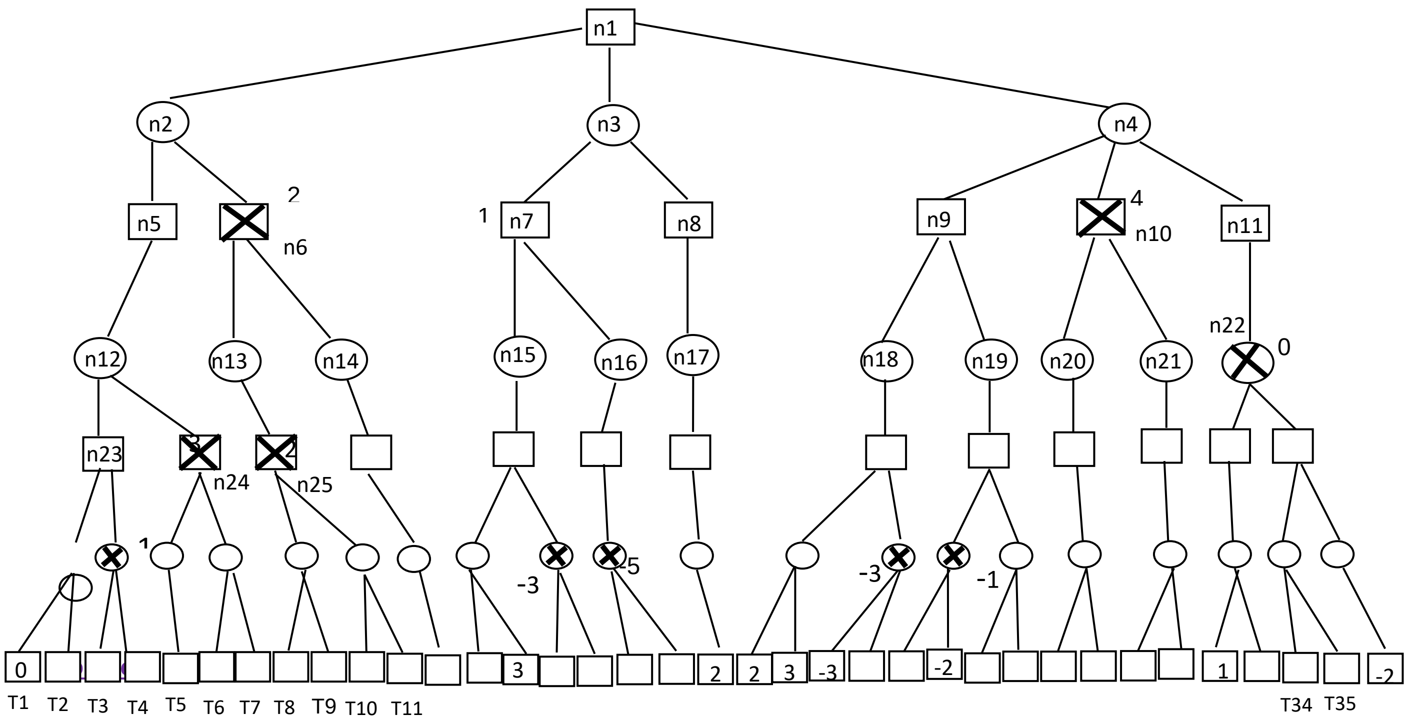
Exercise 1

The following tree shows a partial problem-solving from a depth-first α-β procedure where nodes have been examined in left-to-right order. The tree only shows the static scores of some of the tip nodes.

- Nodes in squares are MAX nodes, nodes in circles are MIN nodes
- There are 36 tip nodes (T1, T2, ..., T36). Only the static scores of some of them are shown.
- Values in the rest of nodes are the final backed up values
- The picture shows all the effective cut-offs, i.e. the cut-offs that prevent from generating the second successor in the nodes that have X's drawn through them (non-effective cut-offs are not shown)

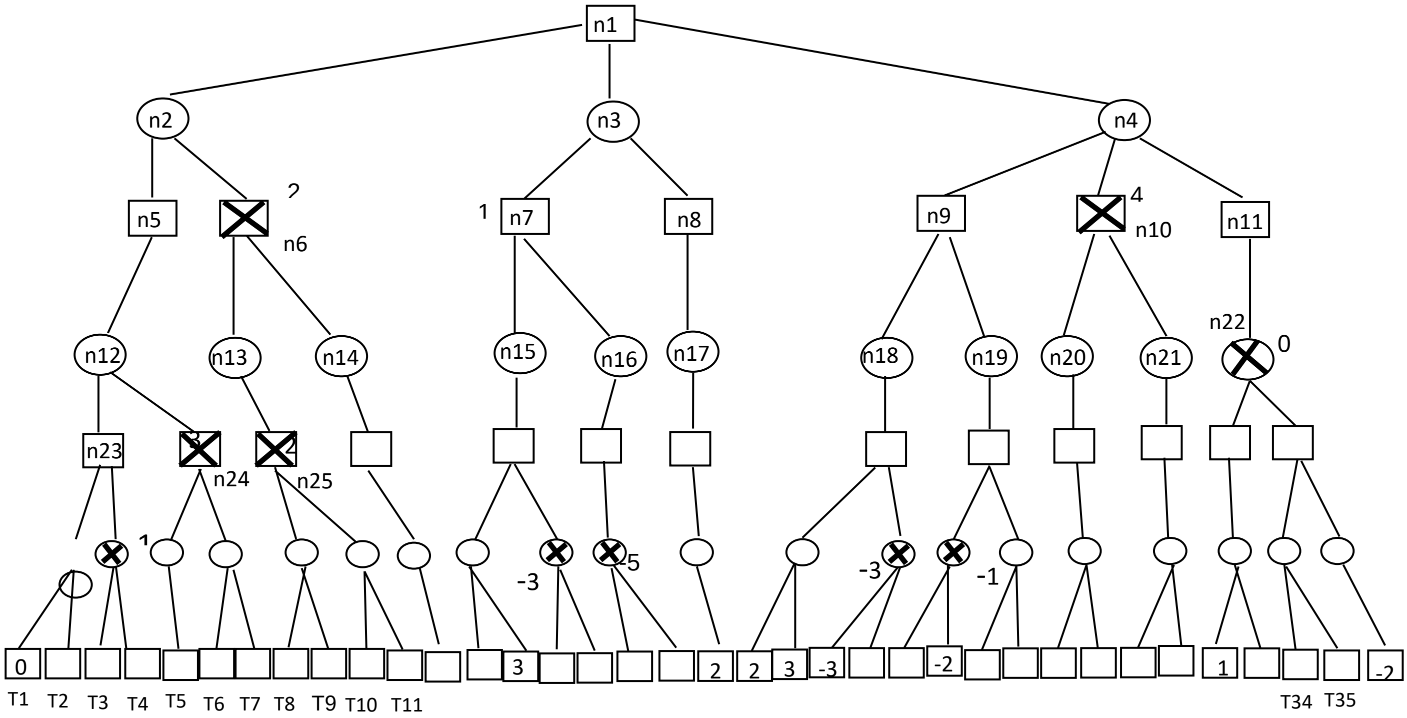
Give brief and justified explanations to the following questions:

1) Is it possible to know the value - otherwise a range of values- of the start node MAX? If so, show this value (or range of values).
2) Is it possible to know the branch selected by MAX? If so, show which of the three branches.
3) Is it possible to know which tip node (or tip nodes) give the final value to the start node MAX? If so, show which one(s)
4) Is it possible to know the static scores (or a range of scores) of tip nodes T8 and T9? If so show these scores (or range of scores)
5) Is it possible to know the static score of terminal node T16? If so show this value.
6) Do you think the application of the α-β procedure achieves the optimal reduction in the tree? Why? Otherwise, indicate how we could achieve an optimal α-β search.



Solution:

Ponemos nombres a algunos de los nodos para hacer más fácil el seguimiento del algoritmo. Comenzamos la aplicación del algoritmo por la rama más a la izquierda.



El nodo MIN padre de T1 y T2 tendrá un valor <= 0 [-∝, 0]. Este rango de valores se vuelca en el nodo MAX n23 como valor provisional. Al subir el valor de T3 (-1) se produce un corte que sólo puede venir provocado por el nodo MAX n23; por tanto el rango de valores de n23 será [-1,0]. Este rango valores se vuelca en n12 que provoca el corte de n24. El rango [-1,0] se vuelca a n5 y n2. El corte de n25 lo provoca los valores de n2. Los nodos terminales T8 y T9 tendrán que tener valores 2 y >= 2 (o >=2 y 2), de modo que se vuelca el valor 2 a su nodo MIN antecesor que a su vez es el valor que se vuelca a n25. El valor 2 se vuelca hasta n6 donde se produce un corte por el nodo n2. El valor de n2 [-1, 0] se vuelva provisionalmente en n1.

La segunda rama está prácticamente estudiada. El nodo n7 tiene un valor de 1 que se vuelca provisionalmente en n3. Como el valor que se propaga por el segundo hijo de n3 es el valor 2, n3 tomará el valor 1 como definitivo que cambia el valor provisional de n1 a 1.

Tercera rama. Al nodo MAX sucesor de n18 llega un valor provisional de 2, que es el que provoca el corte del nodo con valor –3. El valor 2 será definitivo en este nodo MAX, se vuelca a n18 y provisionalmente a n9. El valor definitivo de n9 no es conocido porque no se conoce el valor definitivo volcado por la rama de n19. No obstante, sabemos que tendrá que ser un valor >= 2 por tratarse de un nodo MAX. Un análisis más detallado nos dice que tiene que ser un valor >= 3 porque sino habría un corte, provocado por el valor 2 de n9, en el nodo hermano al nodo donde se produce el corte de –1. El valor >= 3 se vuelca como provisional a n4.

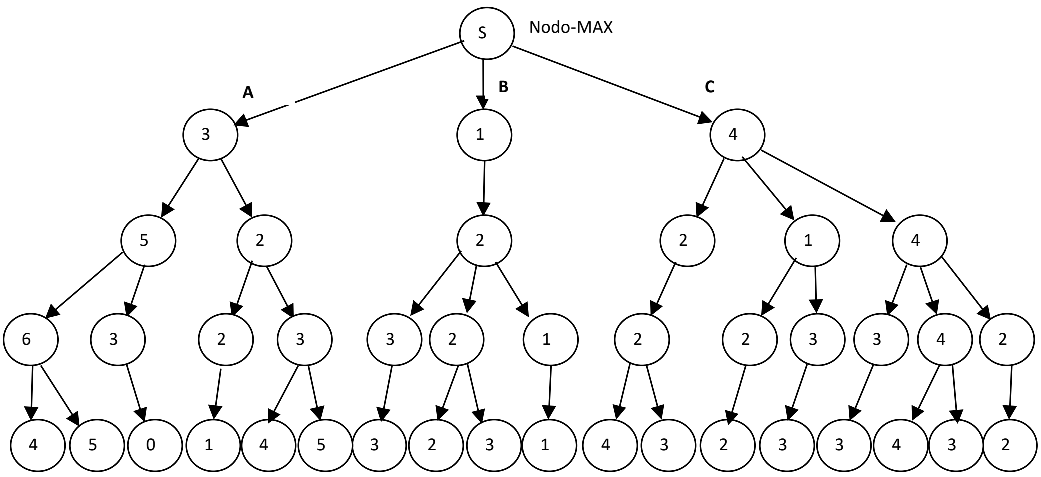
El corte de n10 lo provoca justamente el valor provisional de n4 así que en estos momentos sabemos que el valor provisional de n4 es [-∞, 3]. El corte de n22 lo provoca el valor provisional 1 de n1. Será el valor 0 el que se propague hasta n4. Por tanto el valor definitivo de n4 es 0.

1-2) Y por consiguiente el valor final de n1 es 1, que se lo da la segunda rama

2. Concretamente el nodo terminal T13 es el que le da el valor final al nodo MAX inicial..
3. Los valores de T8 y T9, tal y como se ha explicado arriba, serán 2 y >= 2 respectivamente o al revés.
4. No es posible conocer el valor de T16 porque ese nodo no se genera por el corte producido en su padre.
5. No, no es un α-β óptimo. Para que así fuera debería haberse estudiado antes la rama del medio y volcar primero el valor final asumido por el nodo inicial MAX (1), lo cual permitiría podar gran parte de la rama izda. de n1. Concretamente, si se estudia antes la rama del medio, se produciría un corte en el nodo sucesor

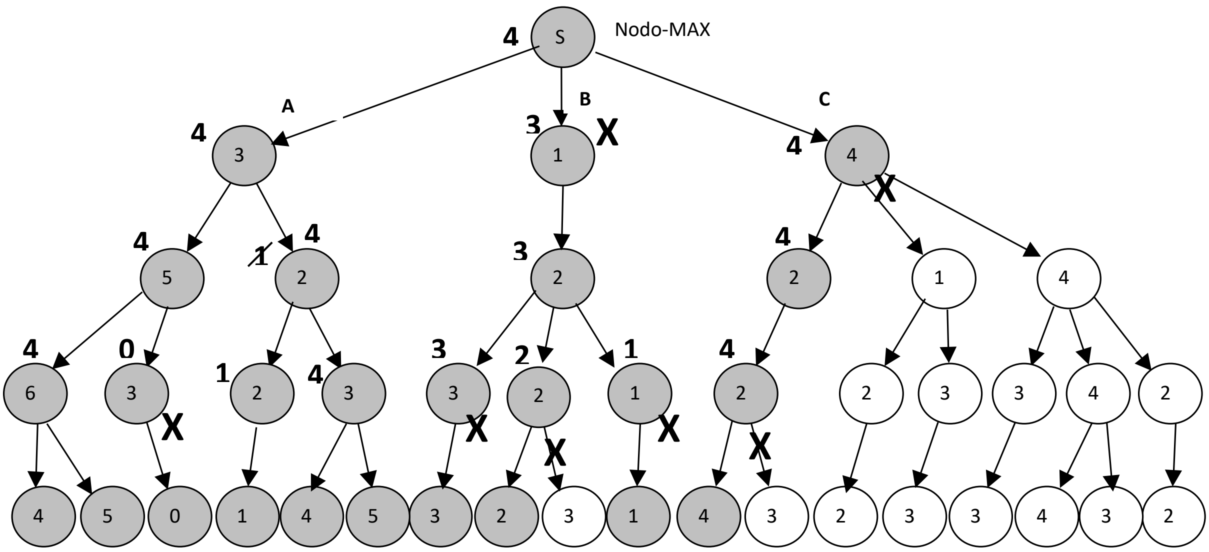
a la izda. de n23, en n12 y en n2, evitando así la generación de 10 nodos de la rama más a la izda. del nodo MAX inicial.

Exercise 2



Assuming an alpha-beta procedure with depth-first expansion is applied over the game tree (starting by the leftmost successor nodes), indicate which is the best move for MAX (A, B or C). Show clearly the cuts-offs the alpha-beta procedure makes and the initial branch selected by the start node MAX.

Solution:



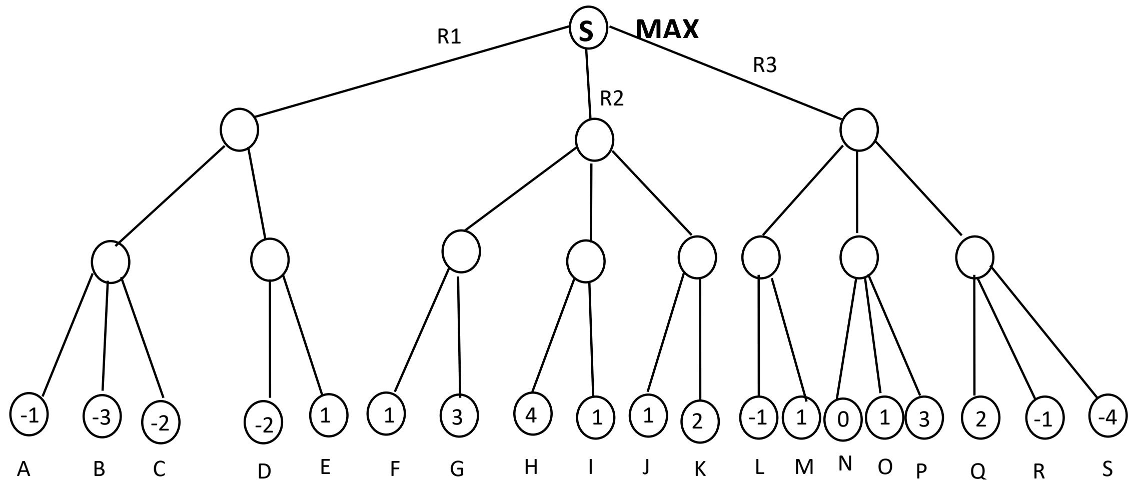
La mejor jugada será la rama A, ya que, aunque se produce una meseta con la rama C (igual valor final volcado en los hijos de S), por debajo de la rama A no hay corte, con lo que el valor final volcado coincidirá con el que se obtendría con MINI-MAX, mientras que en la rama C el valor final volcado es una cota superior (si desarrollase exhaustivamente la búsqueda el valor final volcado podría ser menor, como efectivamente sucedería) del valor que se obtendría en el caso de aplicar MIN-MAX.

Exercise 3

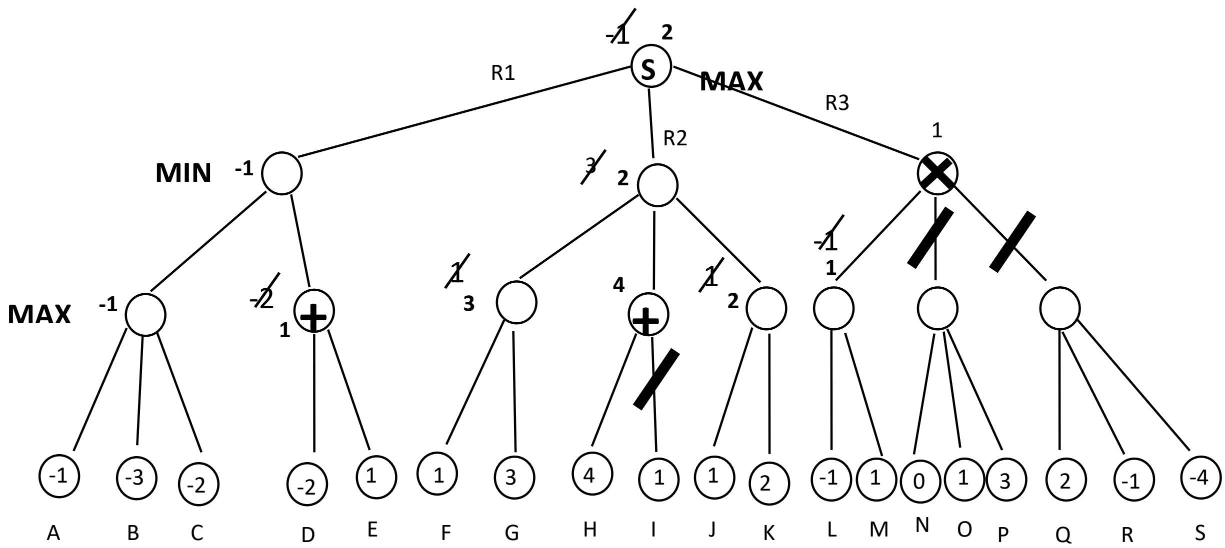
Apply the alpha-beta algorithm with depth-first expansion to the following game tree to find the values of the nodes. Assume that nodes are always considered left-to-right.

a) Show the backed up values. Indicate with symbol **X** the alpha cut-offs and with symbol **+** the beta cut-offs. Show all the cut-offs, effective and non-effective; in the former indicate which branches are pruned. Finally show the best move for MAX.

b) Assuming that nodes are always expanded from the left to right, which values should have the terminal nodes to result in alpha-beta doing the maximum number of cut-offs possible? Show which terminal nodes should change their value and the new value they would take on. Justify your answer.



Solution:



El primer corte que se produce en el nodo padre de los nodos D y E es no efectivo porque se produce cuando se vuelva el valor 1 del nodo E. El siguiente corte beta de la rama del medio se produce porque el padre MIN tiene un valor beta (3) menor que el de su hijo MAX (4). El corte alfa de la rama de la derecha se produce al volcar el valor 1 en un nodo MIN teniendo su antecesor MAX un valor de 2.

b) Un alfa-beta óptimo se obtiene cuando se producen el máximo número de cortes alfa-beta, podando el mayor número posible de nodos. Dicho de otro modo, un alfa-beta es óptimo cuando se producen todos los cortes posibles en el desarrollo del algoritmo (es decir, la respuesta a toda pregunta de corte es SI).

Hay varias formas de conseguir un alfa-beta óptimo para el ejemplo. En este caso, iremos tomando los valores de los nodos terminales tal cual aparecen en el árbol hasta que veamos un corte que no se produce y cambiaremos el valor correspondiente del nodo terminal para que el corte se genere.

- Los nodos A, B y C no cambian de valor ya que no es posible que se produzca un corte porque los nodos superiores aún no tienen valor volcado.

- Al volcar el valor del nodo D no se produce corte pero podría generarse uno ya que su padre MIN ya tiene un valor (-1). Por tanto, si el valor del nodo **D ∈ [-1, +∞]** se hubiera producido un corte beta efectivo, podando de este modo el nodo E. En esta primera rama no es posible conseguir ningún corte más ya que el nodo MIN no tiene valor hasta que su primer hijo se estudia por completo; por otra parte, el nodo MAX superior aún no tiene valor por lo que no es posible obtener un corte.

- En la segunda rama el corte más efectivo que se podría haber producido sería si al volcar el primer valor al nodo MIN se produce un corte alfa con el valor provisional que ya tiene el nodo MAX inicial (-1) (similar a lo que ocurre en la rama de la derecha). Para que esto suceda los valores de los nodos **F y G tendría que** ∈ **[-∞, -1**]. De este modo se hubiera volcado el valor -1 al nodo padre MIN que a su vez hubiera producido un corte con el valor -1 provisional del nodo MAX superior.

- Llegados este punto, el nodo MAX tiene un valor provisional de (-1). Los nodos L y M tienen que ser estudiados para conseguir un primer valor en el nodo MIN y así poder producirse un corte con el valor del nodo MAX inicial. Si cambiamos **M por un valor [-∞,-1]** el nodo MAX padre se quedaría con un valor -1 que subiría al nodo MIN que a su vez provocaría un corte alfa con el valor del nodo MAX inicial.

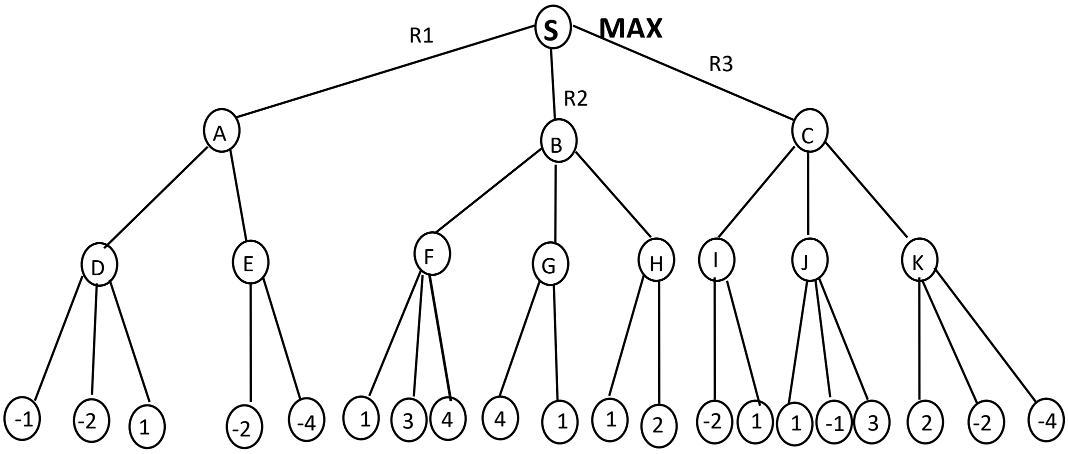
En resumen, los nodos que tendrían que cambiar son:

  - D ∈ [-1, +∞]
  - F,G ∈ [-∞, -1]
  - M ∈ [-∞, -1]

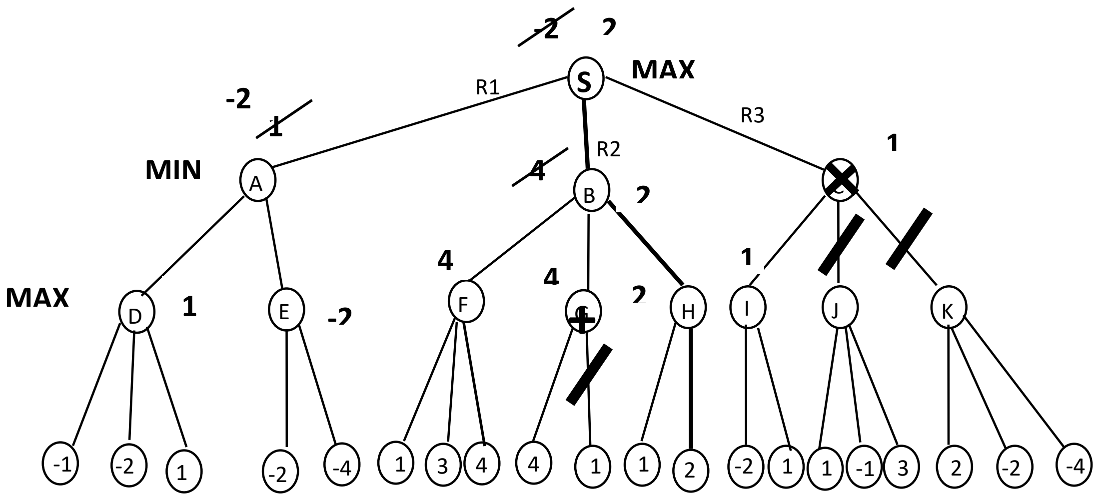
Exercise 4

Apply the alpha-beta algorithm with depth-first expansion to the game tree in the figure to find the values of the nodes. Assume that nodes are always considered left-to-right. Show the backed up values. Indicate with symbol **X** the alpha cut-offs and with symbol **+** the beta cut-offs. Show all the cut-offs, effective and non-effective; in the former indicate which branches are pruned. Finally, show the best move for MAX and indicate the number of pruned nodes.

We want to apply alpha-beta algorithm again but this time we want the algorithm application returns the maximum number of pruned nodes as possible. Apply again alpha-beta by selecting the order in which nodes at levels 1 and 2 are expanded (terminal nodes are always considered left-to-right). Show and justify the order you select nodes A,B,C, …, K.

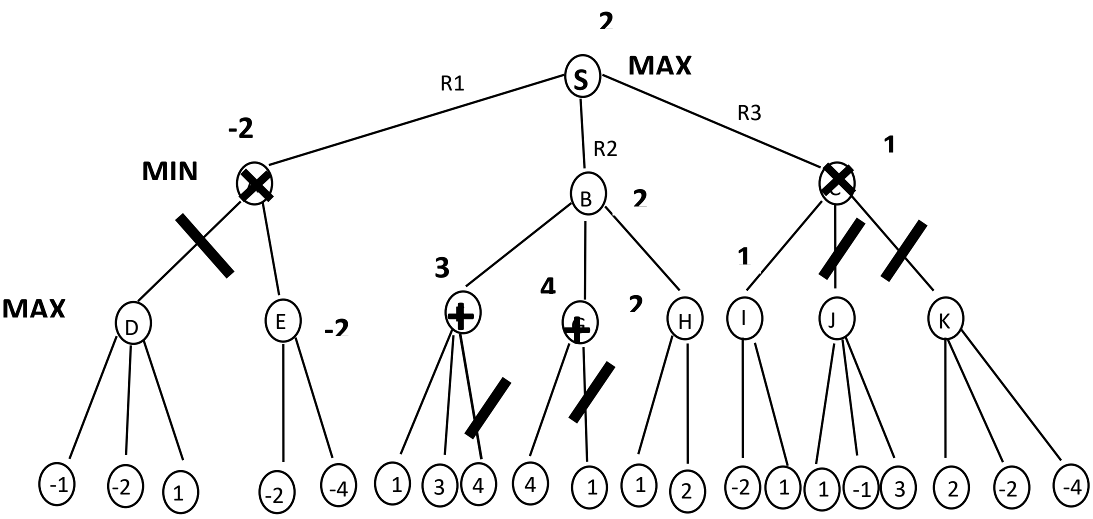


Solution:



Se produce un corte beta en el nodo G y un corte alfa en el nodo C, podando así sus respectivos hijos, en total 9 nodos. La mejor jugada de MAX sería la rama R2; concretamente, el nodo terminal que daría valor al nodo MAX raíz sería el hijo del nodo H con valor 2.

b) Para que se produzca el mayor número de nodos podados posibles, habría que volcar en primer lugar el valor del nodo terminal que maximiza el nodo raíz MAX, esto es, el hijo del nodo H con valor 2. De este modo, expandimos primero el nodo B y a continuación su hijo H.



En el siguiente paso, deberíamos estudiar el nodo F o G. No importa el orden en el que se estudien estos dos nodos, ya que producirán los mismos cortes, ambos producidos por el valor 2 del nodo B. Por tanto, el orden seguido hasta el momento es (B,H,F,G) o bien (B,H,G,F).

En el siguiente paso podemos estudiar el nodo A o el nodo C. De nuevo, el orden no influirá en el número de nodos podados porque en ambos casos se producirá una poda por el valor 2 provisional del nodo S. Estudiamos primero el nodo C. Ahora tenemos que elegir el nodo hijo de C que vamos a estudiar en primer lugar. El nodo J no produciría poda porque el máximo valor de sus nodos terminales es 3, que no cortaría con el valor 2 provisional del nodo S al volcarlo en el nodo C. El nodo K sí produciría poda al volcar el valor máximo 2 de sus nodos terminales, pero en dicho caso se podarían 7 nodos (los nodos I y J y sus respectivos hijos); por el contrario, si estudiamos primero el nodo I, se producirá una poda con respecto al valor 2 provisional del nodo S y podremos podar 8 nodos (los nodos J y K, los 3 hijos de J y los 3 hijos de K).

El orden escogido hasta el momento es (B,H,F,G,C,I).

En el último paso estudiamos el nodo A. El máximo valor que adquiriría D es 1 y el máximo valor que adquiriría E es -2. En ambos casos se produciría una poda en el nodo A con respecto al valor 2 provisional del nodo S. Por tanto, estudiamos en primer lugar el nodo E ya que producirá una poda de 4 nodos.

En resumen, se podan en total 14 nodos y el orden final es: (B,H,F,G,C,I,A,E) pudiendo alterar el orden entre F y G y el orden entre C y A.

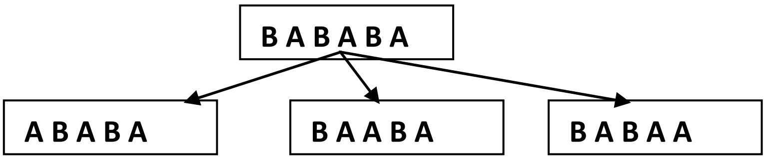
Exercise 5

Two players, A and B, want to play with some playing counters that are lined up in a row. There are two types of playing counters, A-counters and B-counters. Players A and B play alternatively (in turns).

- Player A's turn: Player A can take all B-counters that are between two A-counters or between an A-counter and the board end.

- Player B's turn: Player B can take all A-counters that between two B-counters or between a B-counter and the board end.

The next figure shows the possible moves for player A from the initial situation (B A B A B A):



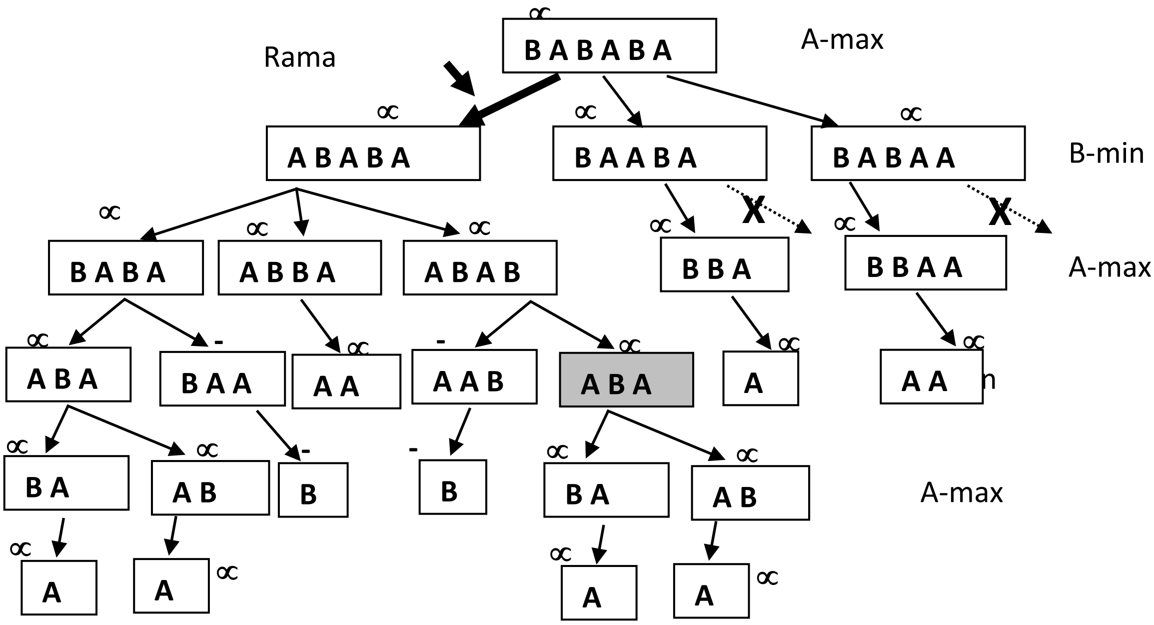
The game is over when one player has no more counters. The winner is the one who still has counters when the game is over. Apply an alpha-beta procedure from the initial situation in the figure [B A B A B A] assuming player A is the starting player and the tree is generated until reaching final states (states in which one of the two players wins). Show the best move for player A.

Important notes:

1. The tree must be generated such that successor nodes are ordered in the following way: the leftmost node will represent the state in which the leftmost counters of the opponent are removed.
2. Take into account that it is not necessary to generate the whole search tree since we are using an alpha-beta procedure. Show only those nodes that must necessarily be generated.

Solution:

El árbol que se explora es el siguiente, incluyendo los cortes α-β:



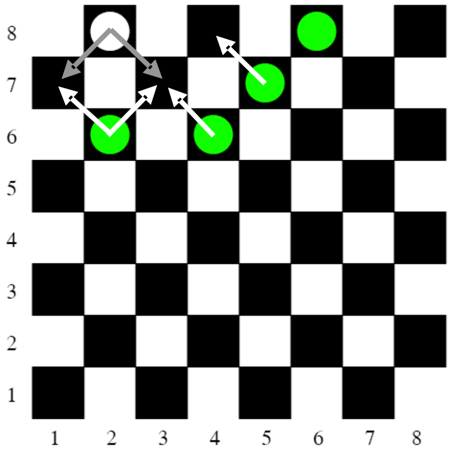
Adicionalmente, podría haberse considerado la repetición de nodos ya alcanzados en la búsqueda (sombreados en gris). También, podrían efectuarse más cortes si se aplica alfa-beta de forma que se tiene en cuenta que una evaluación provisional ∝ para un nodo alfa no es superable por ninguna otra, e inversamente para nodos beta.

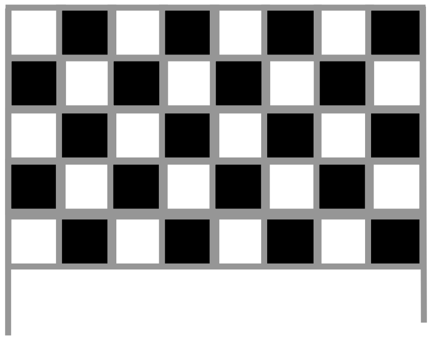
Exercise 6

Two people play the cat-and-mouse game over a chess board (8x8) using only the black tiles of the board. Cats play with black counters and the mouse plays with a white counter (see the figure below; cats are the shadowed counters and the mouse is the white one). Initially, cats are situated in the black tiles of the first row and the mouse is in any of the black tiles of the last row. The two players play in turns. The player who plays with cats can only move a counter at a time (a single cat), and the cat counter always moves forward to any of the two adjacent diagonal tiles in the upper row. The player who plays with the mouse can move the mouse counter forward or backward to any of the four adjacent diagonal tiles.

Cats win the game if they get to corner the mouse and prevent it from moving. The mouse wins if: i) it reaches a tile in a row lower than the row of the belated cat or in the same row of the belated cat AND ii) it is the mouse's turn to play (since cats can only move forward and consequently the mouse would get away)

Assuming the initial situation of the left figure and that the mouse plays first (MAX node):

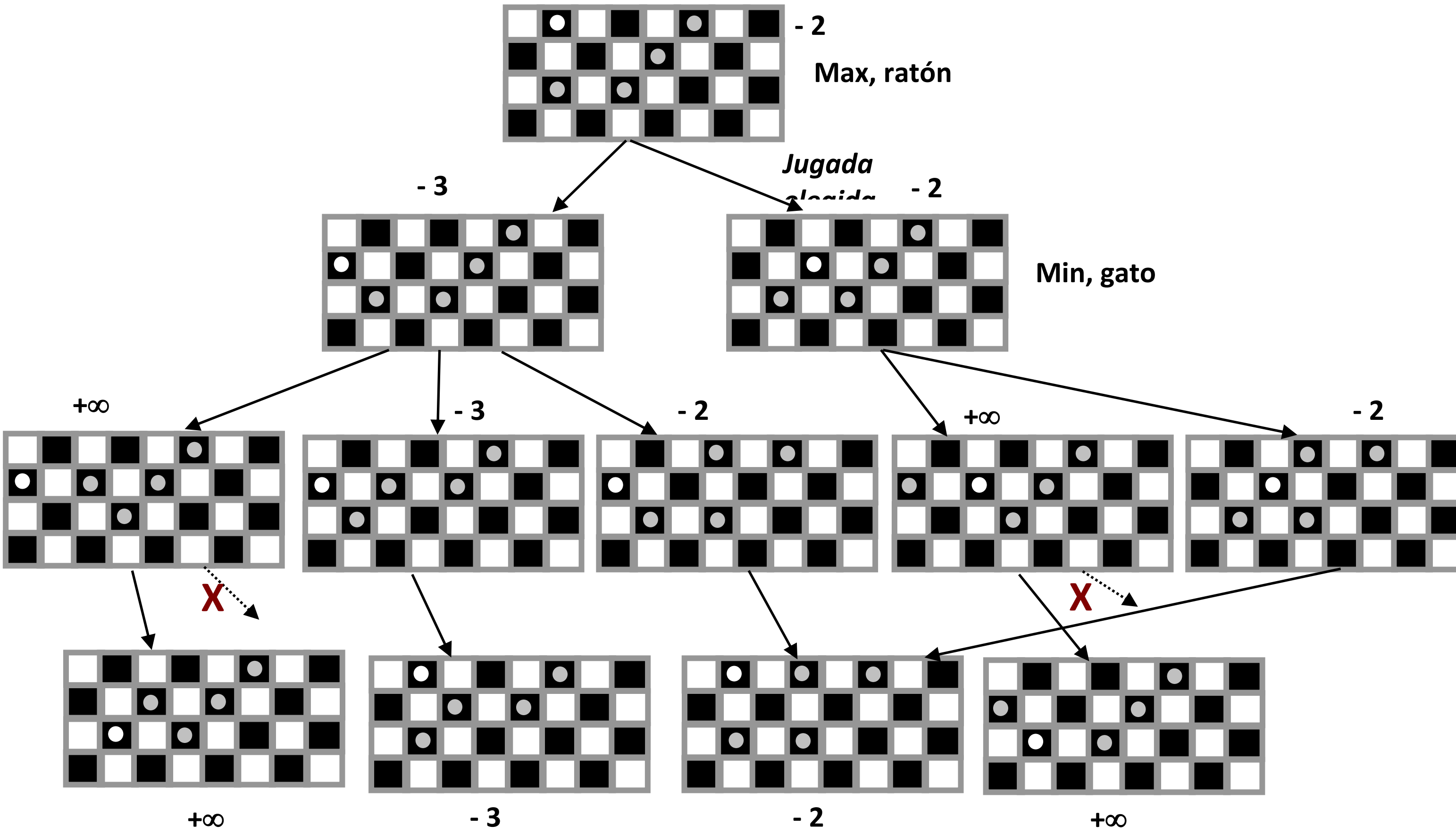




- Design a simple heuristic function such that it returns +∞ if the mouse wins and -∞ if the mouse loses
- Apply an alpha-beta algorithm in depth-first expansion up to a maximum depth level of 3. Show the cut-offs and the best initial move. Apply the heuristic function you have designed to evaluate the terminal nodes. Use the pattern in the right figure above to represent the nodes of the tree.
- NOTE: Assume the following conventions in the alpha-beta application according to the numbered rows and columns of the figure:
  - ***Mouse priority***: first, move the mouse to a lower row rather than an upper row and, second, move the mouse to a lower column rather than an upper column. For example, if the mouse is at tile (row=6, column=4) it will move first to tile (row=5, column=3)
  - ***Cat priority***: first, move the left-most cat; second, move the cat to a lower column, that is to the left diagonal tile. For example, if the left-most cat is at tile (row=2, column=2), it will move first to tile (row=3, column=1).

Solution:

Por ejemplo, definimos **f(n)= - (nº gatos en filas inferiores al ratón)**. Será más positiva (menos negativa) cuanto mejor sea para el ratón. Asumimos +∞ (o -∞) si gana (o pierde) el ratón. El árbol que se genera es el siguiente, produciéndose los cortes indicados. Se indica la mejor jugada inicial.



Exercise 7

The figure (a) shows the initial state of a sliding tile puzzle that consists of a row of 5 squares, with 2 white tiles (W) at the left, 2 black tiles (B) at the right and an empty space in the middle. The figure (b) shows the final state we want to reach.

1	2	3	4	5
W	W		B	B

(a) Initial state

1	2	3	4	5
B	B		W	W

(b) Final state

The legal moves or actions, with costs, are as follows:

A1. If the empty space is next to a tile, the tile may move into the empty space
A2. A tile may hop over 1 tile (of either colour) into the empty space
A3. A tile may hop over 2 tiles (of either colour) into the empty space

Additionally, tiles W can only move towards the RIGHT, and tiles B can only move towards the LEFT.

Consider we have two players: player-W plays with W tiles, and player-B plays with B tiles. Players can make the moves specified in the problem formulation (A1, A2 and A3).

The objective of player-W is to place the two W tiles in positions 4 and 5. The objective of player-B is to place the two B tiles in positions 1 and 2. If a player cannot move any of his tiles at his turn (that is, player-W cannot move any tile to the right or player-B cannot move any tile to the left), the player loses his turn and the opponent moves. Apply an alpha-beta with depth-first expansion from the initial situation (WW_BB), assuming player-W is the starting player and the tree is generated until reaching final terminal states (states in which one of the two players wins). Show the best move for player-W. Justify your answer.

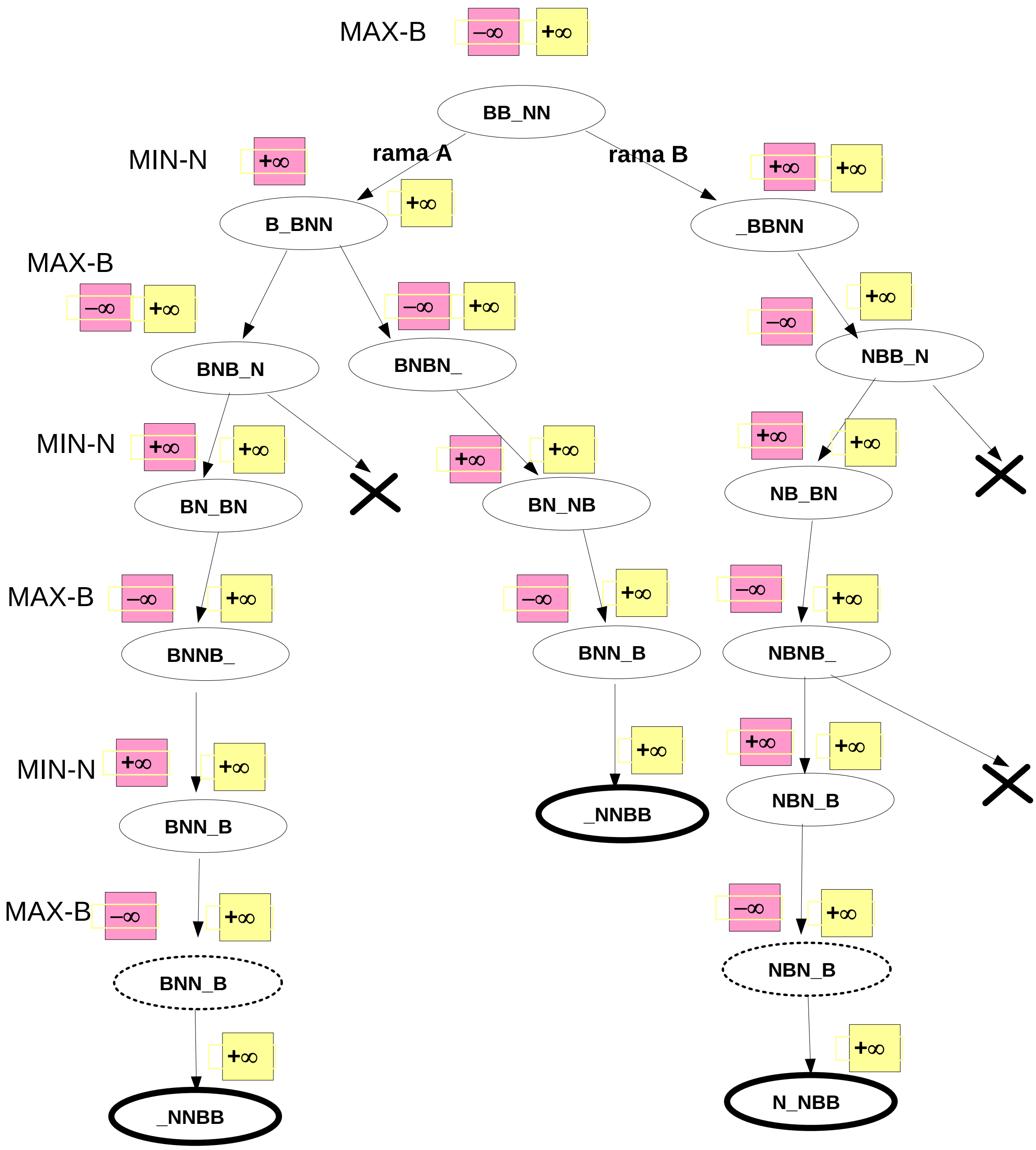
RECALL NOTE: The initial beta value of a MIN node is +∞, and the initial alpha value of a MAX node is -∞.

Solution:

En rosa aparecen los valores iniciales de MIN y MAX. En amarillo los valores volcados. Se muestran solo los cortes efectivos.

El jugador-B puede escoger cualquiera de las dos jugadas iniciales porque ambas le conducen a una situación ganadora. En ambas ramas se producen cortes pero los cortes de la rama B no son consecuencia del valor volcado de la rama A sino del valor inicial +∞ del nodo MIN lo que indica que el corte se produciría igualmente si se hubiera expandido antes por la rama B. Por tanto, en este caso ambas jugadas producen el mismo beneficio.

Los cortes que se producen en la figura son producto del valor +∞ del nodo MIN, pero dichos cortes se podrían producir igualmente siguiendo el razonamiento de que una evaluación provisional +∝ para un nodo alfa no es superable por ninguna otra. En este caso, y siguiendo este mismo razonamiento, podríamos haber obviado la expansión de la rama B.

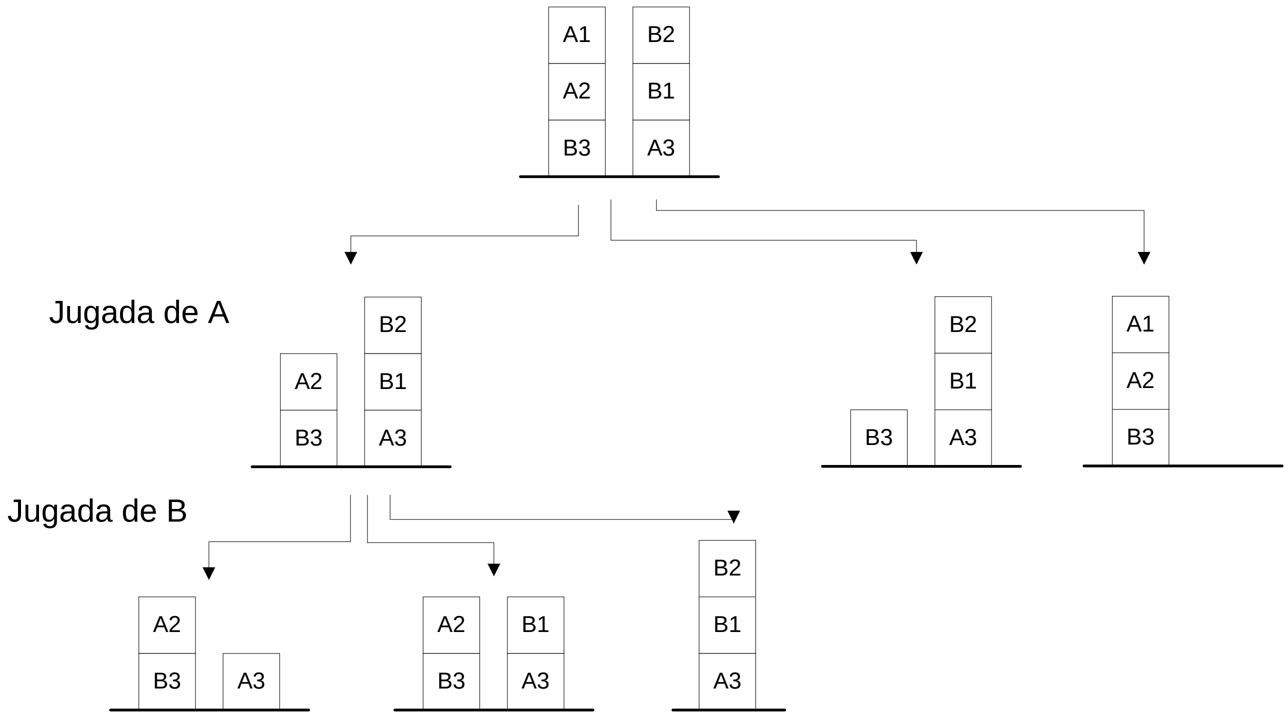


Exercise 8

Two players, A and B, are playing a game similar to the so-called Hackenbusch game. The two players play against each other and alternatively remove blocks from several heaps of blocks situated in front of them. The blocks in the heaps can be of two types, type-A or type-B. Player-A plays with blocks of type-A and player-B plays with blocks of type-B. In each move, a player chooses a block of his/her type and removes it from the heap. When the block is removed, all the blocks on top of it must be removed too, regardless the type of such blocks. The player who has no blocks to remove in his/her turn loses the game.

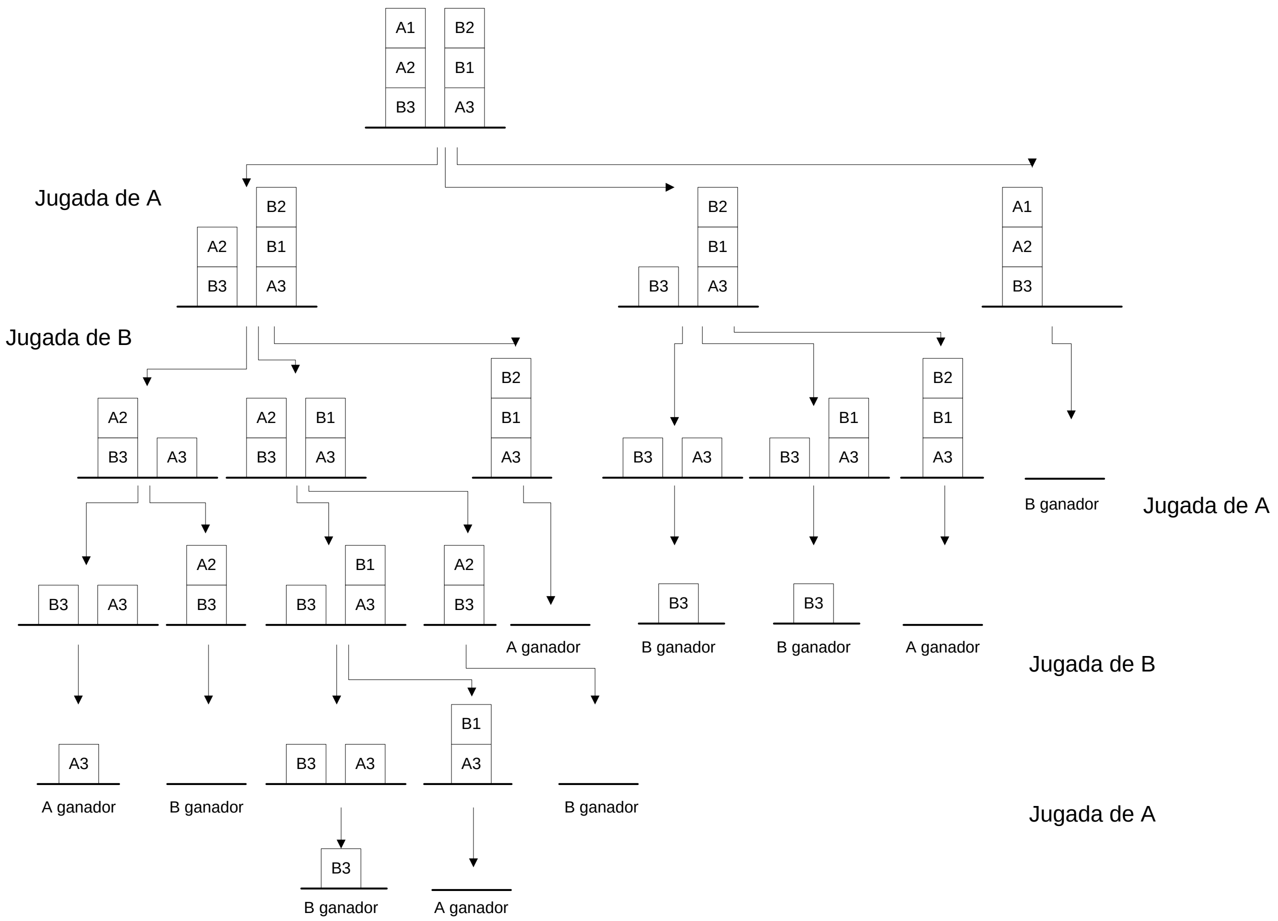
The figure below shows the initial situation of the game as well as the choices for the first move of player-A; it also shows the alternatives of player-B after the first move of player-A.

a) Develop the complete search tree for this game until reaching final terminal states
b) Apply an alpha-beta with depth-first expansion to determine the best move for player-A.
c) Reason about which choices has player-A of winning the game if player-B would apply the same procedure to decide his/her next move.

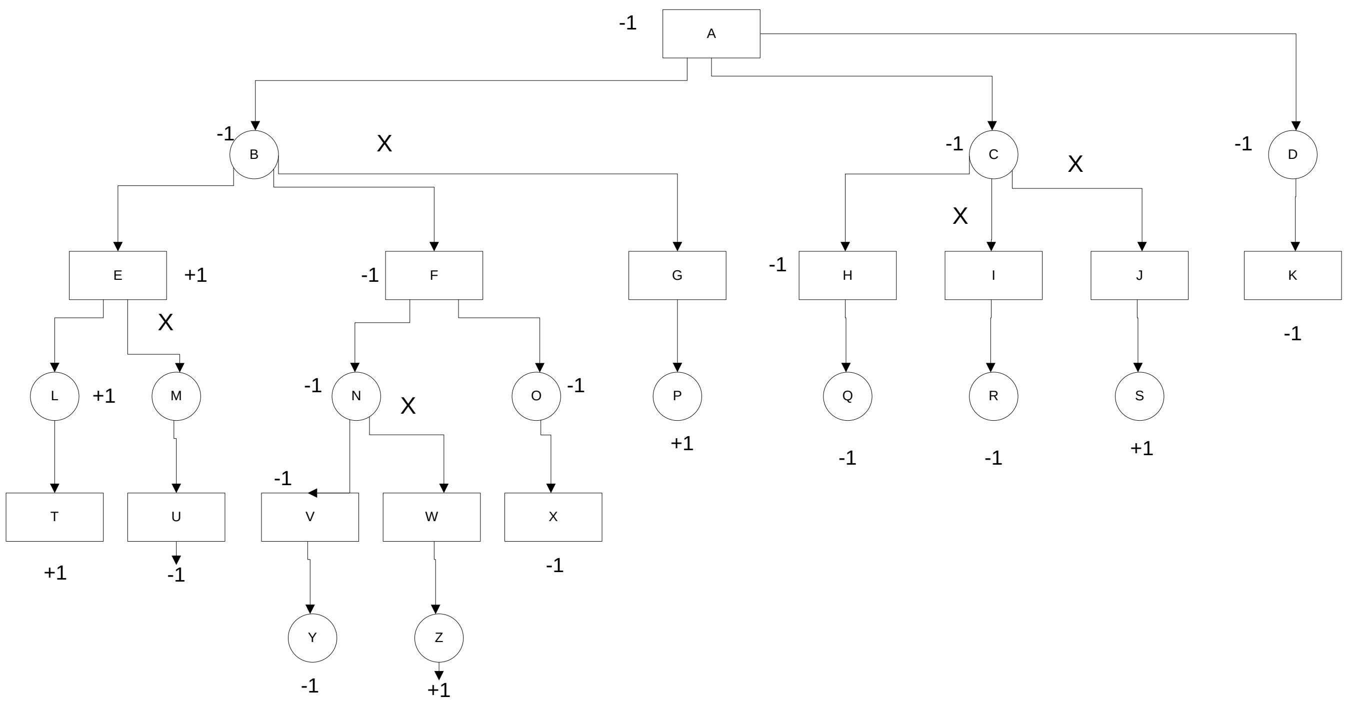


Solution:

a) Desarrollo del árbol de juego:



b) Minimax con poda alfa-beta:



Nota: Utilizamos los valores -1 y +1 equivalentemente a -∞ y +∞.

Comenzamos asignando todos los nodos con sus valores iniciales: -1 para nodos MAX (rectángulos) y +1 para nodos MIN (círculos), ya que son los únicos valores que se podrán volcar desde los nodos sucesores

El procedimiento empieza estudiando la rama de la izquierda. En este caso, el valor volcado al nodo L (+1) es el mismo que inicialmente tenía, por lo que no se modifica. Sin embargo, al volcar este valor +1 al nodo E, sí se modifica, ya que éste es un nodo MAX y su valor inicial era -1. Como B es un nodo MIN y E es un nodo MAX y ambos tienen el mismo valor (+1, que además es el máximo que E puede tomar), se poda la rama M-U. Se vuelca el valor +1 a B, que es el que inicialmente tenía.

Continuamos por la siguiente rama. El valor -1 del nodo Y se vuelca al nodo V, que no modifica su valor. Al subir el valor al nodo N, éste sí modifica el valor asociado, pasando a ser -1. La rama W-Z se poda ya que N es un nodo MIN con valor -1 y F es un nodo MAX con el mismo valor (también se podría considerar que -1 es el mínimo valor que puede tomar N). Y este valor -1 se vuelca en el nodo F (que no se modifica). Seguimos por el siguiente nodo terminal, X, que vuelca su valor en O. Igualmente, se sube -1 a F, que ya tenía este valor. B es un nodo MIN y actualmente tiene el valor +1, pero al subir el valor de F, se modifica a -1. Como B es un nodo MIN y ya tiene el menor valor que puede tomar, podamos la última rama sucesora de B (esta poda también se realiza si tenemos en cuenta que A tiene como valor -1 que es igual al valor del nodo B). El valor -1 de B se vuelca en A (aunque no se modifica).

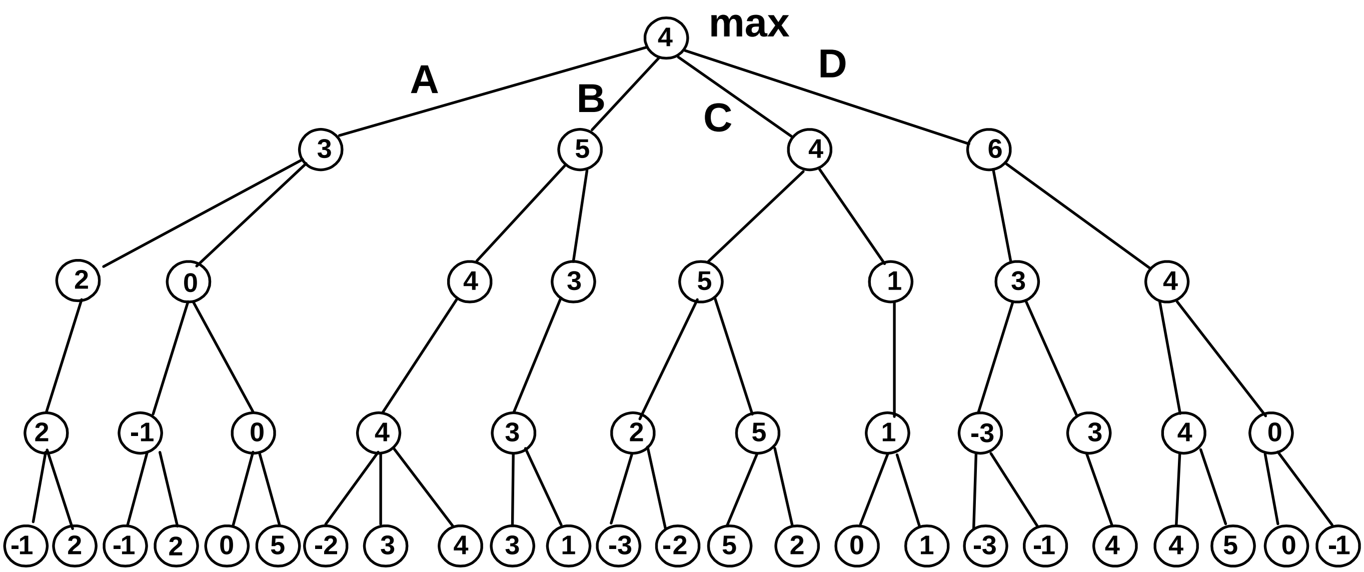
Estudiamos ahora la rama más a la izquierda de nodos sucesores de C. Q vuelca su valor en H y éste en C, que pasa a tener el valor -1. Como el valor del nodo C (MIN) es igual al valor del nodo A (MAX), se realiza una poda de ambas ramas (efectivamente, ya no es necesario estudiarlas, puesto que C tiene el mínimo valor que puede tomar).

Por último, el valor del nodo K se sube al nodo D, que modifica su valor. Aquí se produce una poda no efectiva, ya que este nodo no tiene otros sucesores aparte del ya estudiado.
A la vista del resultado del procedimiento Minimax con poda alfa-beta, no existe una jugada que claramente sea mejor para A. La jugada que corresponde al nodo D se descarta rápidamente, ya que ésta conduce a A a una derrota clara. La jugada que corresponde al nodo C también se descarta porque esta rama tiene varias podas, por lo que no se ha estudiado en profundidad. Así que se escogería como jugada la que corresponde al nodo B, cuyos nodos sucesores llevan a varias jugadas ganadoras para A.

c) A no tiene ninguna posibilidad de ganar, ya que todas las ramas nos indican que existe una combinación de jugadas en las que ganará B (siempre y cuando B escoja la mejor opción en cada caso).

Exercise 9

The following tree represents a game search space. The values inside the intermediate and terminal nodes denote the result of applying a particular evaluation function f(n).



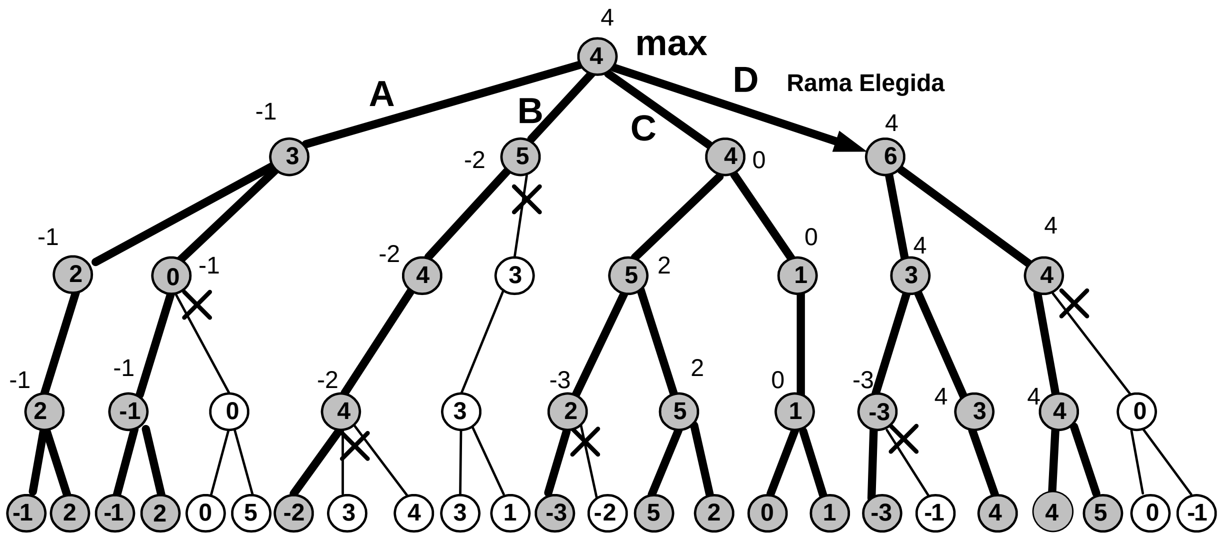
Apply an alpha-beta with depth-first expansion expanding first the left-most nodes.

- Show the best initial move for player MAX (A, B, C or D). Mark clearly the branches and nodes that have been generated. Show the final backed-up value of each node.
- Show the effective cut-offs.
- Say the total number of generated nodes (except the root node) and the total number of nodes that have been evaluated.

Solution:

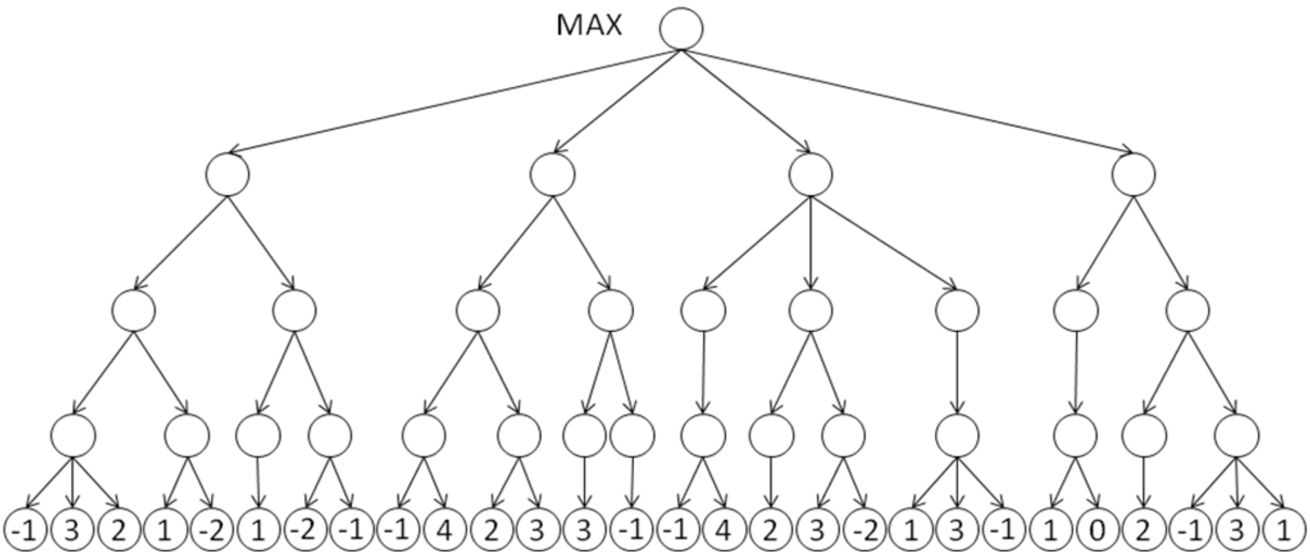
Expansión en profundidad (no se muestran cortes no efectivos).

En total se generan 34 nodos (excluyendo el raíz) y solo se evalúan los nodos terminales explorados (14 nodos). Ello daría un coste temporal de 34 + 14*2= 62 segundos



Exercise 10

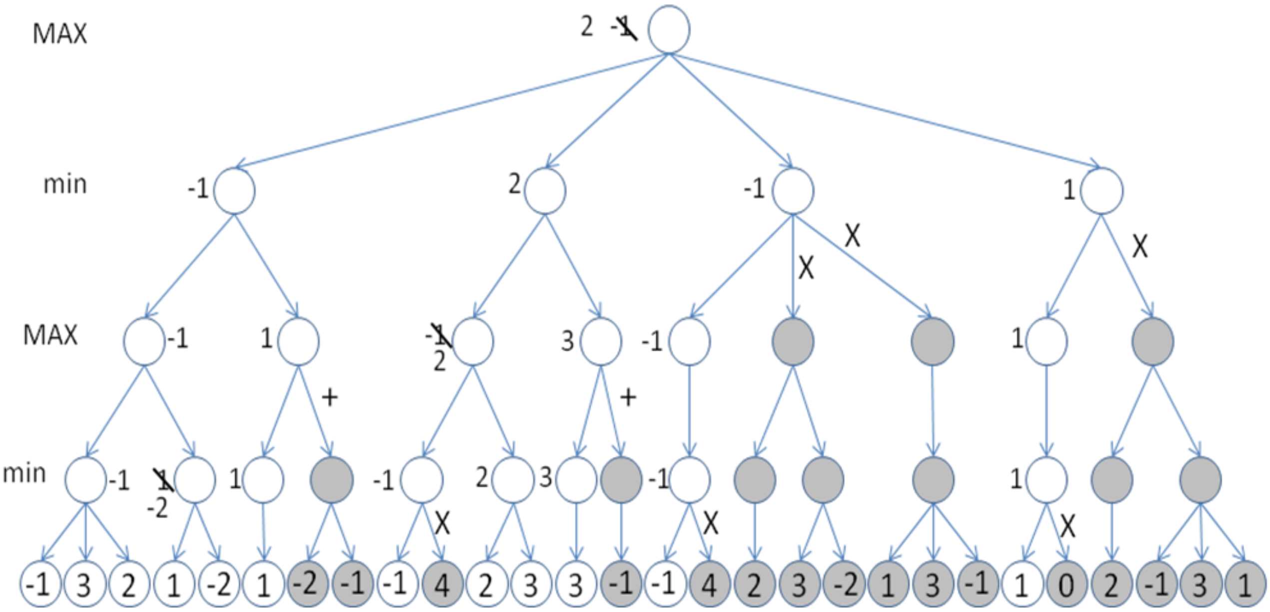
In the game tree shown below, the numbers inside the terminal nodes are the values resulting from applying an evaluation function f(n).



Apply an alpha-beta with depth-first expansion expanding first the left-most node. Mark with **X** the alpha cut-offs, and with **+** the beta cut-offs. Show the best initial move for the MAX player.

Solution:

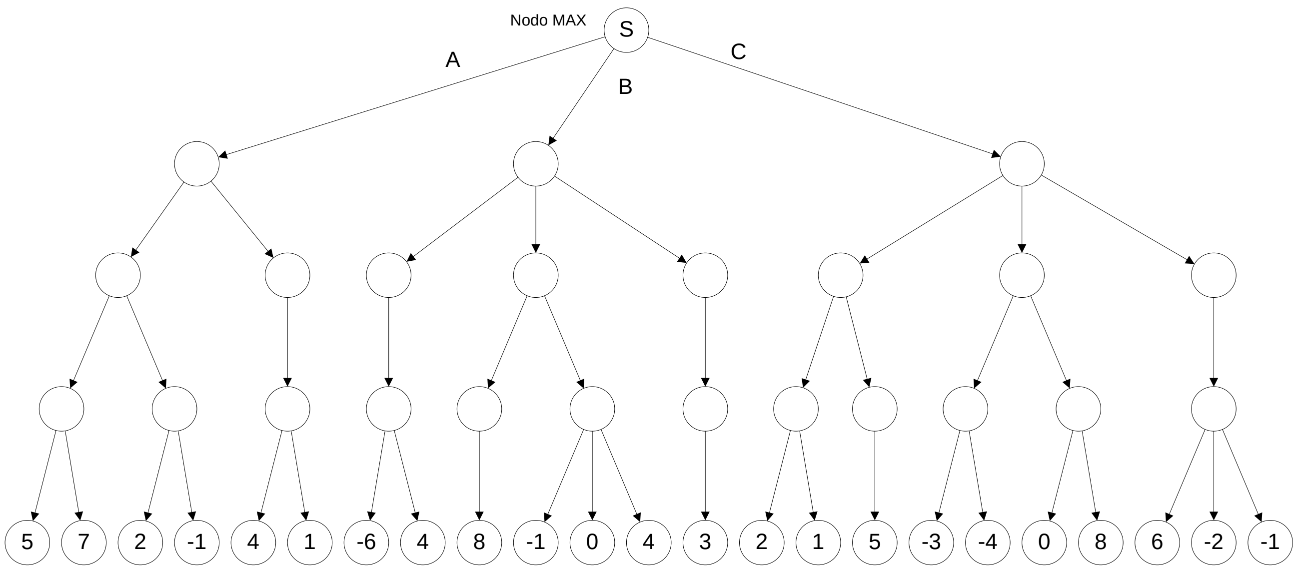
Expansión alfa-beta en profundidad



La rama que escoge el algoritmo es la segunda.

Exercise 11

The following tree represents a game search space. The values inside the terminal nodes denote the result of applying a particular evaluation function f(n).

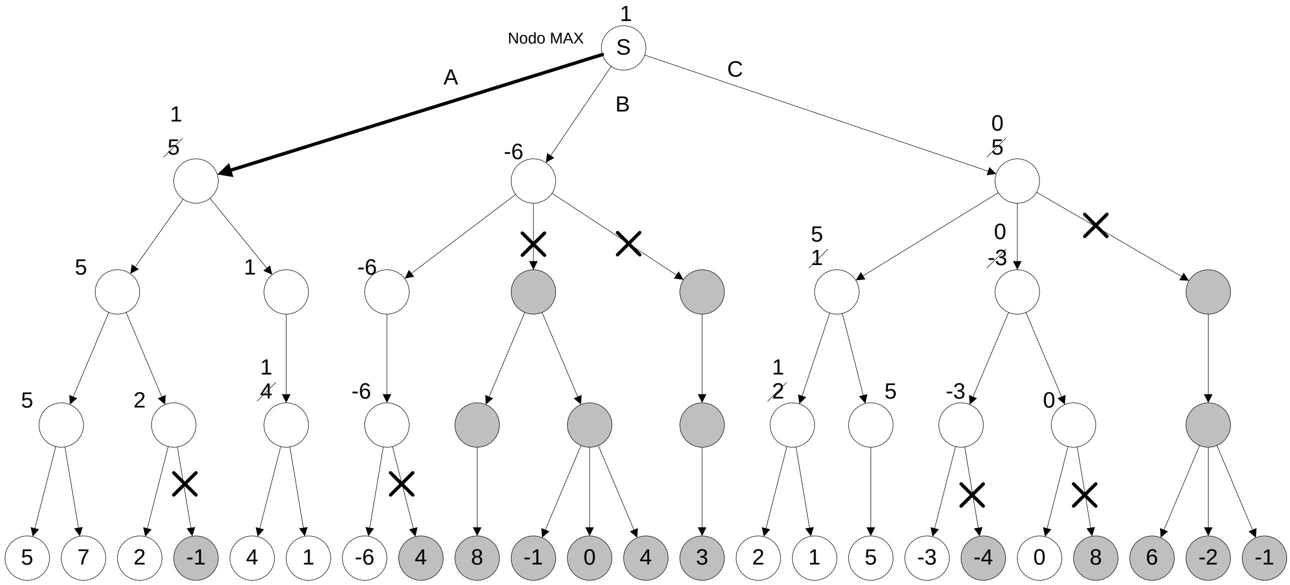


a) Apply an alpha-beta with depth-first expansion expanding first the left-most nodes. Show the best initial move for player MAX (A, B or C). Mark clearly the branches and nodes that are generated. Show the final backed-up value of each node. Show the effective and non-effective cut-offs.

b) The branch that MAX would select with the application of the MINIMAX procedure with no alpha-beta pruning, would it be the same branch? Justify your response.

Solution:

a)

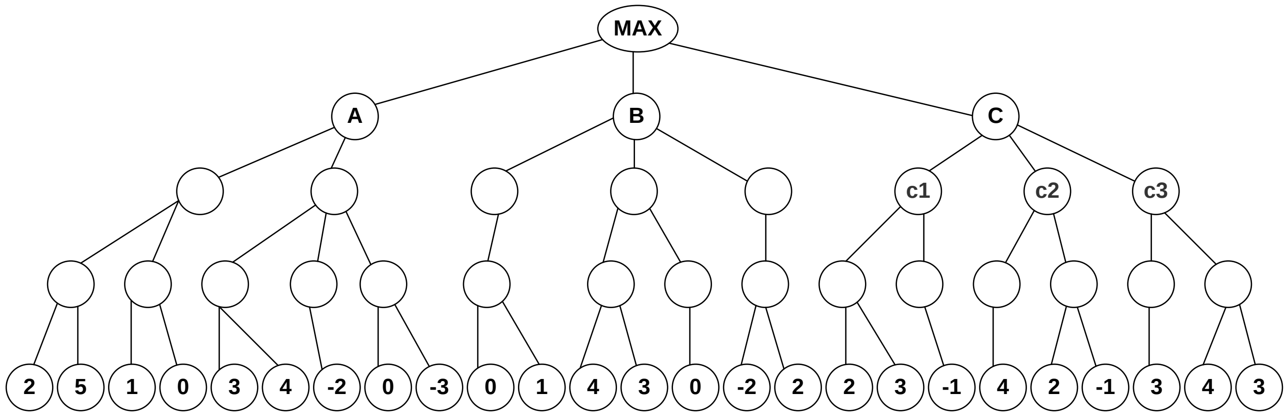


b) No difiere, ya que al aplicar el algoritmo MINIMAX sin poda alfa-beta, el valor de la segunda rama también sería -6, y en el caso de la tercera sería -2. Hay que tener en cuenta que si se aplica alfa-beta, los nodos Min pueden tomar valores mayores que los que tomarían si no se aplicara (debido a que no se exploran ramas que no pueden mejorar el valor del nodo Max que provoca el corte).

Exercise 12

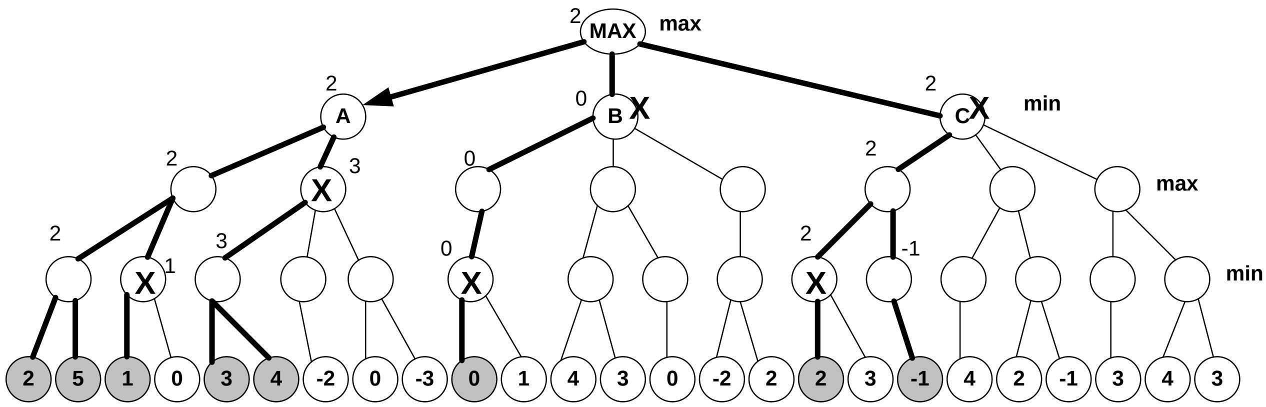
The figure below shows a game tree in which the values inside the terminal nodes represent the result of a particular evaluation function.

a) Apply an alpha-beta pruning expanding first the leftmost successor nodes. Show the best initial move for MAX (A, B or C), the final backed-up values, the cut-offs produced and the nodes which are pruned.
b) In case that MAX player chooses branch C, which move is the most recommendable for MIN next? Apply a MNIMAX procedure to determine the best move for MIN (c1, c2 or c3).

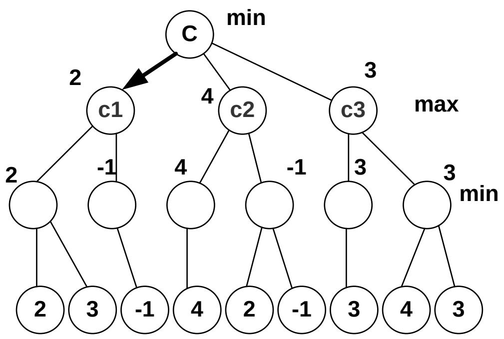


Solución

a) La rama elegida es A y el árbol resultante es:

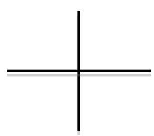


b) La aplicación de MINIMAX a la jugada C es (MIN elegiría la rama c1):



Exercise 13

Consider the 2x2 tic-tac-toe game (the figure below shows the board):



The game rules are:

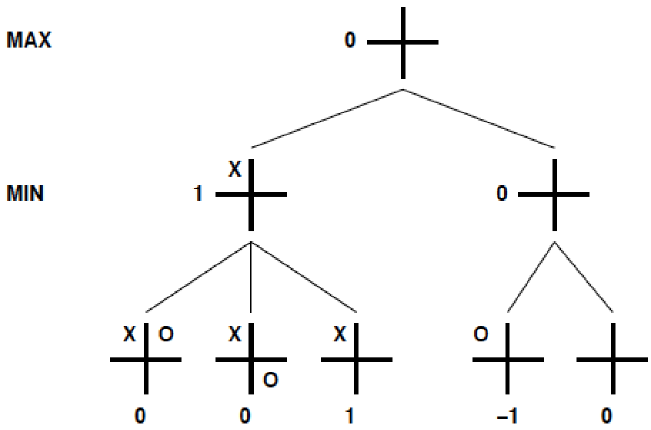
- MAX draws the symbol 'X' and MIN draws the symbol 'O' on a square at their respective turns
- Another choice for a player is to pass the turn, i.e. not to put an 'X' or an 'O'

As usual, we will assume that MAX plays first:

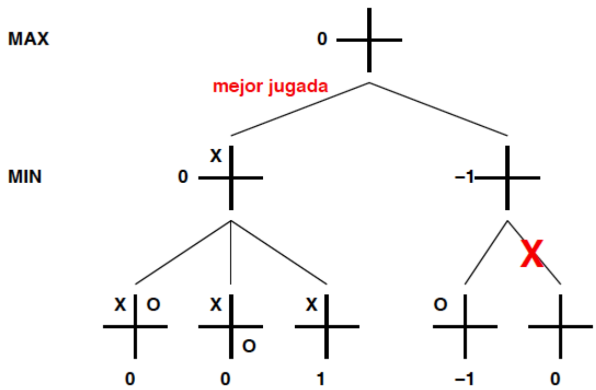
a) Draw the complete game tree up to depth 2. Do not draw symmetrical nodes to nodes already in the tree (resulting from rotations or reflections). The complete game tree must have 5 leaf nodes.
b) Let's assume the evaluation function is "f(n)=number of X minus number of O in node n". Show the f-value of all the nodes (leaf nodes and intermediate nodes) in the game tree.
c) Apply the alpha-beta algorithm to the tree in section (a) with the values of the leaf nodes obtained in section (b). Which is the best play for MAX?
d) Apply the MINIMAX algorithm to the tree in section (a) with the values of the leaf nodes obtained in section (b). Which is the best play for MAX?

Solution

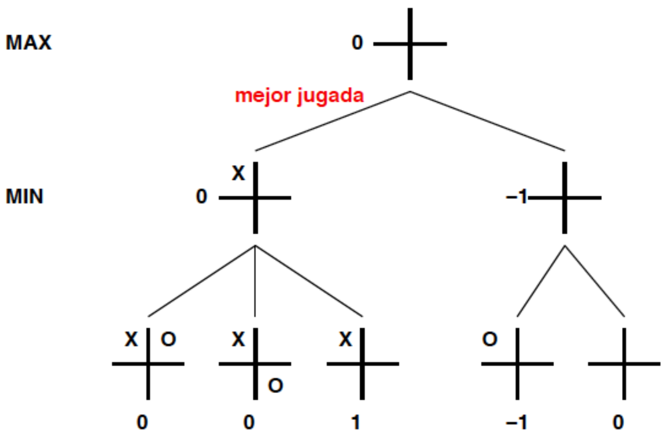
a) and b)



c)

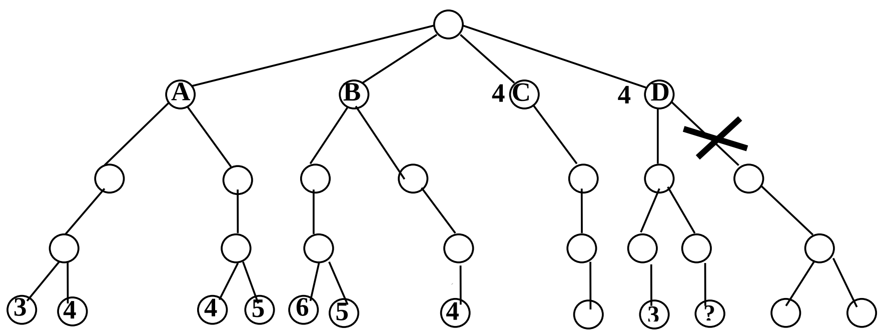


d)

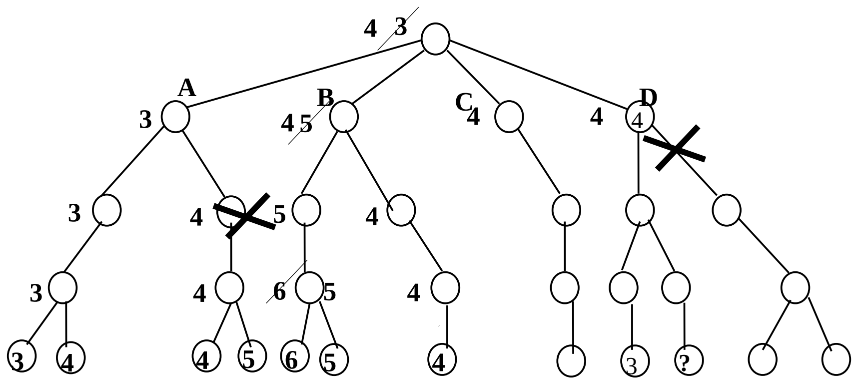


Exercise 14 (Exam 2013)

The tree below shows partially the result of applying an alpha-beta algorithm, from left to right, to a game search space. The figure occludes some of the backed up values as well as some of the cut-offs. Answer the following questions and justify your responses.



a) Apply alpha-beta on the two branches on the left (A and B) and show the backed-up values.



b) According to the resolution in section a), which branch (A, B, C or D) is the best play for MAX? why?
Se produce una meseta entre las ramas B, C y D, y por tanto cualquiera de ellas podría ser la mejor jugada. Sin embargo, si se ha producido alguna poda por debajo de una rama, los valores volcados podrían ser mayores que los obtenidos en caso de no producirse dichas podas.

La rama B será la primera en ser expandida, y no se producen podas, por tanto el valor volcado obtenido será el mismo que en el caso de utilizar MINIMAX y ésta será la mejor jugada en el caso de utilizar alfa-beta, es decir ‘la rama de mayor valor volcado que haya sido expandida en primer lugar’. Por debajo del nodo C tampoco se producen podas, por lo que la rama C también podrá ser considerada mejor jugada. En la rama D no podemos garantizar que el valor volcado no sea menor que en caso de desarrollar completamente la búsqueda por debajo de la misma.

c) Considering the cut-off below the node at level 1 in branch D, which would be the value of the node labeled as ‘?’?

El valor será 4, si aplicamos el algoritmo alfa-beta, tal como se indica en la figura, podemos comprobar que este es el único valor posible.