Inteligencia Artificial
Tema 4. Teoría de Juegos

José Manuel Fuertes García
jmf@ujaen.es

Departamento de Informática
Universidad de Jaén

20 de marzo de 2018

Objetivo

Determinar y resolver los problemas que surgen cuando tratamos de planear en un mundo donde otros agentes planean contra nosotros.

Índice

Juegos y agentes

- Un agente debe tomar decisiones, frecuentemente en entornos con incertidumbre.
- En muchos casos, existen otros agentes cuyas decisiones afectan a las nuestras.
- Muchas de estas situaciones pueden ser modelizadas como juegos.

Juegos vs. problemas de búsqueda

- Los juegos están basados en búsqueda.
- Adversarios "impredecibles" $\Rightarrow$ la solución es una estrategia que especifica un movimiento para cada réplica posible del adversario.
- Limitaciones en el tiempo de respuesta $\Rightarrow$ aproximación, en lugar de búsqueda del óptimo global.

Se trata de construir sistemas que sean capaces de tomar decisiones en un entorno adverso.

Juegos

Tipos de juegos

- En base a la información de que disponen:
  - Con información perfecta, deterministas: *ajedrez, damas*
  - Con información perfecta, con incertidumbre (azar): *póker, backgammon*
  - Con información incompleta, con incertidumbre (azar): *juegos de cartas*
- Movimientos alternativos *(parchís)* o simultáneos *(piedra, papel, tijera)*.
- Una única jugada *(pares o nones)*, o varias *(tres en raya)*.

- El **objetivo del agente** es encontrar la estrategia que maximice las expectativas de éxito.
- La Teoría de Juegos permite analizar las posibilidades y escoger la mejor opción en cada momento.

Uso de la Teoría de Juegos

En el diseño de agentes

- Supone que el contrincante emplea una estrategia racional.
- Permite seleccionar una estrategia adecuada:
  - Que maximice la probabilidad de éxito.
  - Que impida a los contrincantes predecir nuestro comportamiento.

En el diseño de mecanismos

- Es posible modelizar un entorno de forma que el *bien común* se maximice cuando cada agente maximice su propia utilidad. Ejemplos:
  - Diseño de protocolos de enrutamiento en Internet, de forma que cada ruta tenga un incentivo para actuar maximizando el rendimiento global.
  - Construcción de sistemas multiagentes inteligentes que resuelvan problemas complejos de forma distribuida, sin que cada agente sepa qué parte del problema global está resolviendo.

Índice

Juegos con movimientos alternativos e información perfecta

Características de este tipo de juegos

- Juegos bipersonales.
- Los jugadores mueven alternativamente.
- La ventaja para un jugador es desventaja para el otro.
- Los jugadores tienen toda la información sobre el estado del juego.
- Hay un número finito de estados y decisiones.
- No interviene el azar.

- Ejemplos: 3 en raya, ajedrez, damas, otelo, nim, etc.
- Se analizan mediante técnicas de búsqueda.
  - Valores minimax.
  - Poda *alfa-beta*.

Elementos de un juego

- Jugadores: agente (al que llamaremos `MAX`) y agente adversario (al que llamaremos `MIN`).
- Estados: situaciones por las que puede pasar el juego.
- Estado inicial: describe el comienzo del juego (posición e indicación del jugador que mueve).
- Movimientos: operadores que se aplican a los estados, cambiando las situaciones de juego.
- Función sucesor: devuelve una lista de pares (*movimiento, estado*).
- Estado(s) final(es), representan el final del juego, indicando qué jugador gana.
- Test terminal: determina si se ha alcanzado algún estado terminal (final del juego).
- Función de utilidad: da un valor numérico a los estados terminales, respecto a MAX (p.e. +1, 0 ó -1).

Ejemplo de juego: Nim

- Situación inicial: una pila con N fichas.
- Jugadas: tomar 1, 2 ó 3 fichas de la pila.
- Objetivo: obligar al adversario a coger la última ficha.
- Ejemplo de jugada con $N = 9$:
  1. Jugador 1 coge 3 fichas (quedan 6)
  2. Jugador 2 coge 1 ficha (quedan 5)
  3. Jugador 1 coge 2 fichas (quedan 3)
  4. Jugador 2 coge 2 fichas (queda 1)
  5. Jugador 1 coge 1 ficha (no quedan fichas)
  6. Por tanto, gana el jugador 2.

Elementos del juego en Nim

- Estados: Número de fichas que quedan en la mesa.
- Estado inicial: Número de fichas al comienzo.
- Un único estado final: 0.
- El estado final es ganador para un jugador si es su turno.
- Movimientos: tomar $1, 2$ ó 3 fichas.
- Función de utilidad (para el estado final): 1 si le toca a `MAX` y -1 si le toca a `MIN`.

Ejemplo de juego: tres en raya

En un tablero 3x3 un jugador posee fichas "X" y otro fichas "O". En cada turno el jugador coloca una ficha en el tablero. Gana el que consigue colocar tres de sus fichas en línea.

X	O	
	X	
	X	O

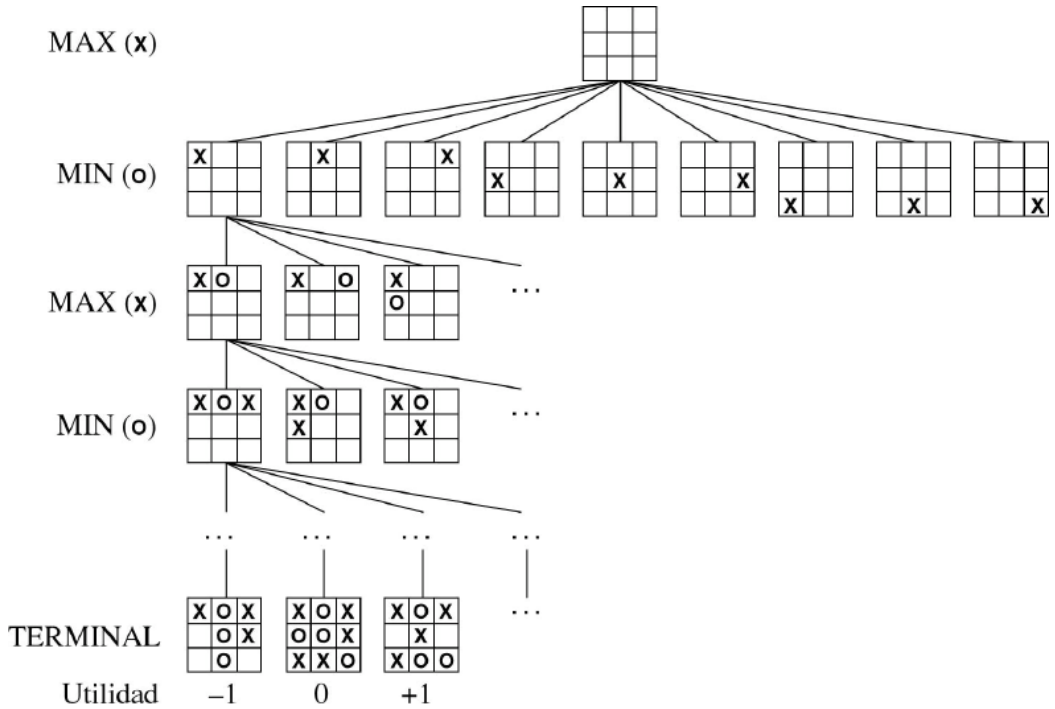
X	O	X
O	X	O
X	X	O

Elementos del juego en 3 en raya

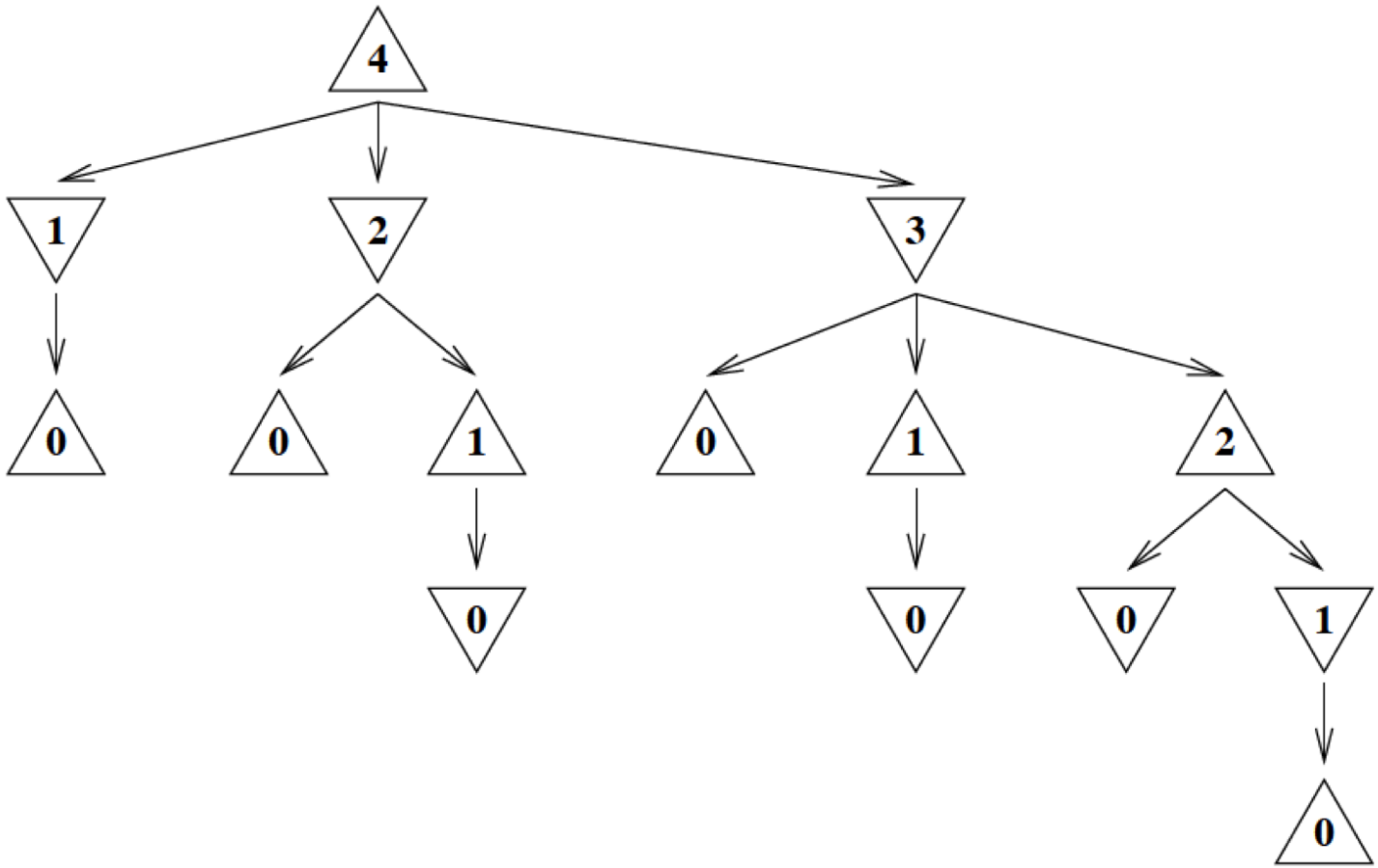
- Estados: tablero + ficha que se pondrá a continuación.
- Estado inicial: tablero vacío + ficha de salida.
- Estados finales: tableros completos o con línea ganadora.
- Estados ganadores para un jugador: estados finales en los que no le toca poner.
- Movimientos:
  - 9 movimientos posibles, uno por casilla.
  - Colocar ficha en $i, (i = 0, \ldots, 8)$.
- Aplicación de movimientos:
  - Aplicable si la casilla no está ocupada.
  - Estado resultante: colocar la ficha que toca en la casilla especificada.
- Función de utilidad: 1 si es ganador para `MAX`, 0 si es tablas y -1 si es ganador para `MIN`.

El árbol de juegos

- Es una representación de todas las posibles situaciones que se pueden dar en el juego, a partir de un estado dado.
- Dos jugadores, `MIN` y `MAX`. Uno intenta minimizar la función de utilidad, y el otro la maximiza.
- Cada nivel corresponde a un jugador (el primero para `MAX`).



Un árbol de juegos para Nim (con $N = 4$)



El problema de decidir el movimiento adecuado

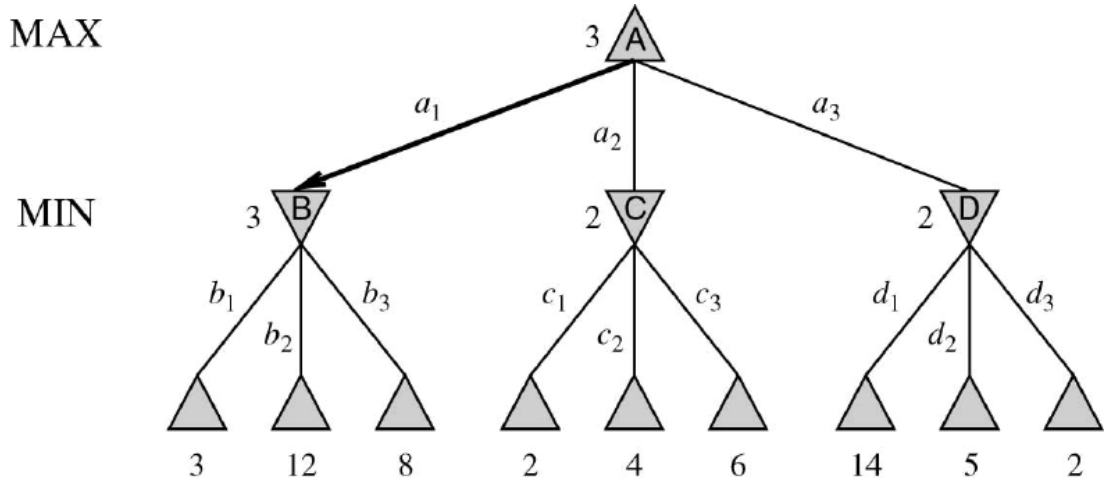
- En cada turno tanto `MAX` como `MIN` deben decidir qué movimiento hacer.
- Idea general:
  - En cada turno, construir el árbol de juego completo cuyo nodo raíz sea la situación actual, desarrollándolo hasta los estados finales.
  - Valorar los finales según la función de utilidad.
  - Propagar hacia arriba los valores de la función.
  - Elegir el movimiento que lleve al estado sucesor del actual con mejor valoración.
- La propagación se hace según el principio *minimax*.
  - `MAX` siempre escogerá lo mejor para `MAX` y `MIN` lo peor para `MAX`.
  - Un nodo de `MAX` toma el valor del sucesor con mayor valor.
  - Un nodo de `MIN` toma el valor del sucesor con menor valor.

Valores *minimax* de un árbol de juegos

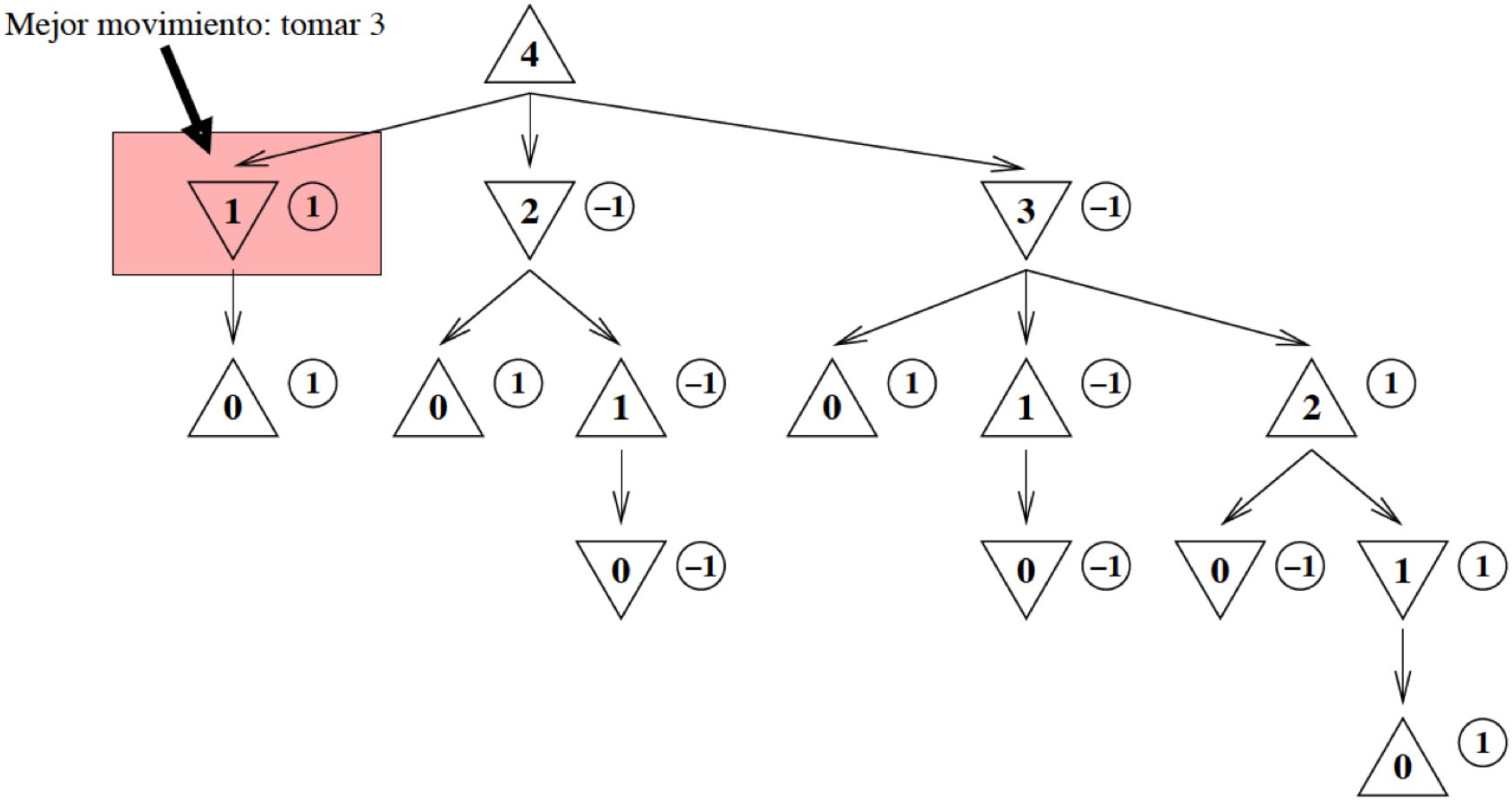
Algoritmo *minimax*

- Proporciona la jugada perfecta para juegos deterministas.
- Cálculo de los valores minimax: propagar el menor valor de los hijos en las capas `MIN` y el mayor en las capas `MAX`.

- La mejor jugada es aquella que proporciona un mejor valor *minimax*.
- Problema: Hay que explorar el árbol primero en profundidad para obtener sus valores minimax.



Un árbol minimax para Nim (con $N = 4$)



Algoritmo *minimax*

- Se basa en el valor *minmax* asociado a cada nodo del árbol.

$$\text{VALOR-MINIMAX}(n) = \begin{cases} \text{UTILIDAD}(n) & \text{si } n \text{ es un estado terminal} \\ \max_{s \in \text{Sucesores}(n)} \text{VALOR-MINIMAX}(s) & \text{si } n \text{ es un estado MAX} \\ \min_{s \in \text{Sucesores}(n)} \text{VALOR-MINIMAX}(s) & \text{si } n \text{ es un estado MIN} \end{cases}$$

- Realiza una exploración primero en profundidad completa del árbol de juegos.
- Ofrece tiempos poco prácticos en juegos reales.

Propiedades de *minimax*

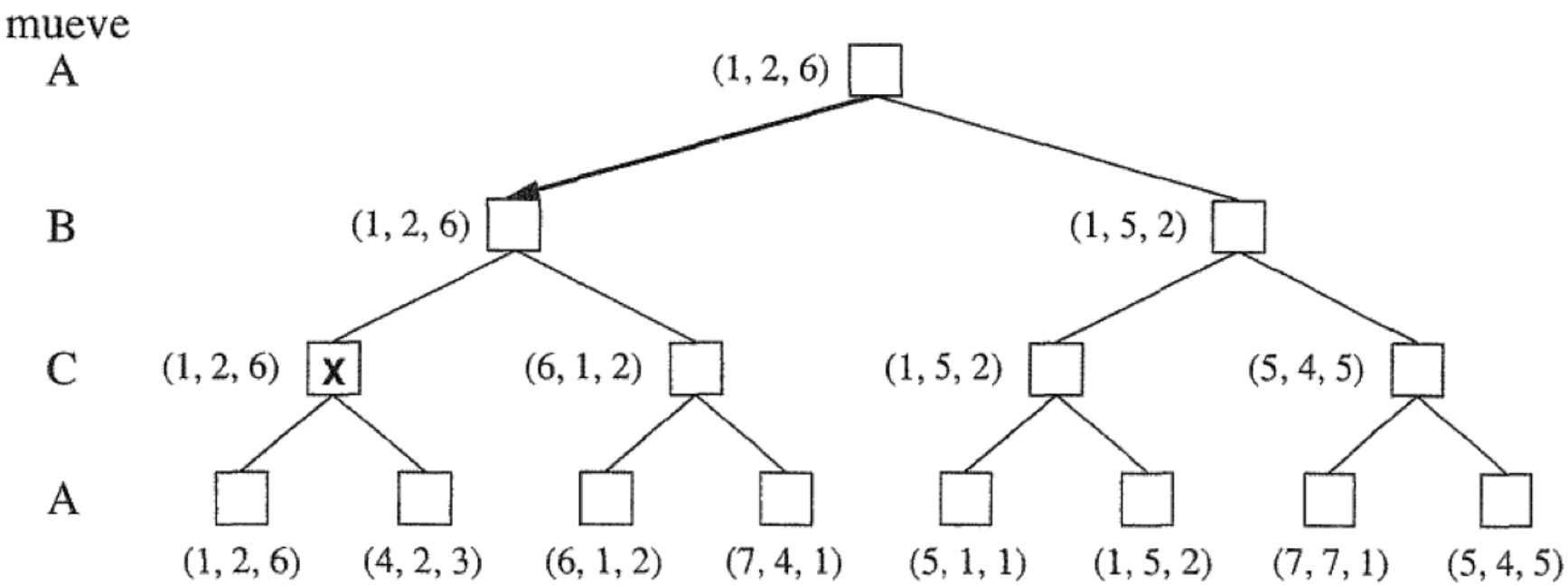
- ¿Completo?: Sólo si el árbol es finito.
- ¿Óptimo?: Si, contra un oponente óptimo. En otro caso...
- Complejidad en tiempo: $O(b^m)$
- Complejidad en espacio: $O(bm)$ (exploración primero en profundidad)

Para el ajedrez: $b = 35, m = 100$ en juegos "razonables" $\Rightarrow$ la solución exacta es completamente infactible. Pero, ¿necesitamos explorar todos los caminos? Idea:

- No expandir el árbol hasta los estados finales sino sólo hasta un determinado nivel de profundidad. En ese nivel, valorar los estados que sean hojas del árbol con una función heurística.
- Propagar valores utilizando el principio *minimax*.

Decisiones óptimas en juegos multijugador

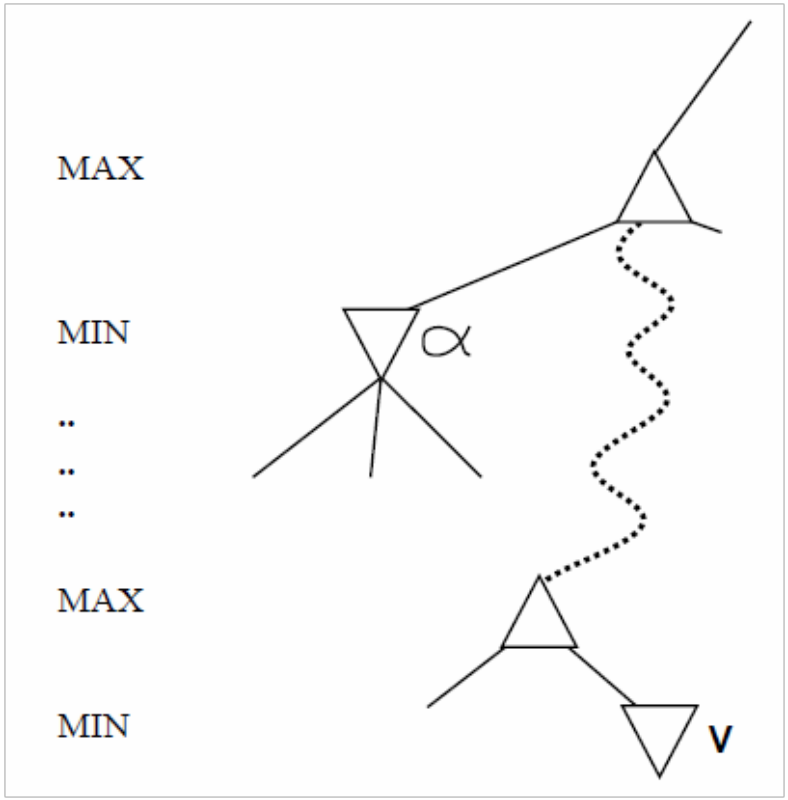
- Para la extensión de *minmax* a juegos hay que sustituir el valor para cada nodo por un vector de valores (uno por jugador)
- El objetivo es maximizarlo.



La poda alfa-beta

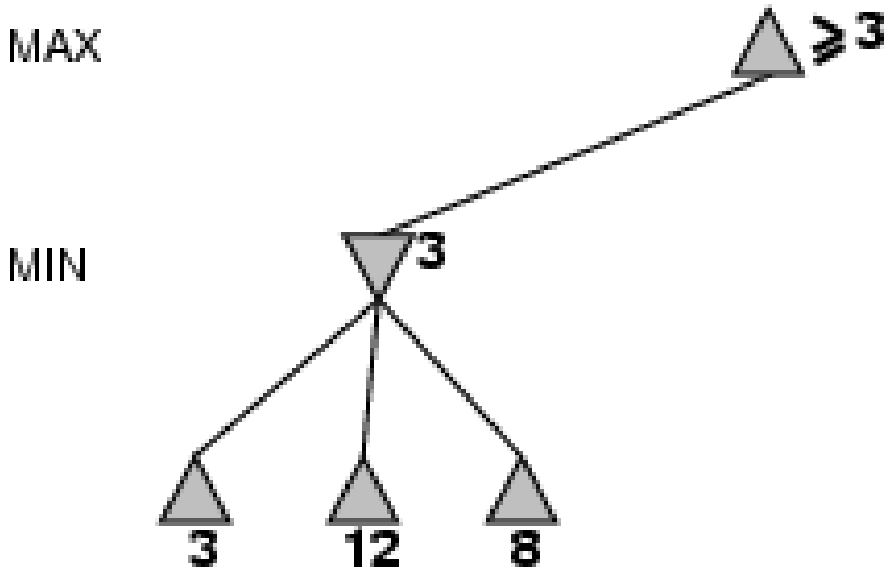
- Evita la exploración (poda) de las ramas del árbol minimax que no van a modificar los valores actuales.
- Cuando no hemos explorado todos los hijos, podemos *acotar* el valor de un nodo.
- Una vez que sabemos que una rama no va a mejorar el valor minimax del nodo actual, la descartamos.
- Debe su nombre a los parámetros que definen los límites:
  - alfa: El valor de la mejor opción que hemos encontrado hasta ahora en cualquier punto elegido a lo largo del camino para `MAX`
  - beta: Idem para `MIN`

La poda alfa-beta

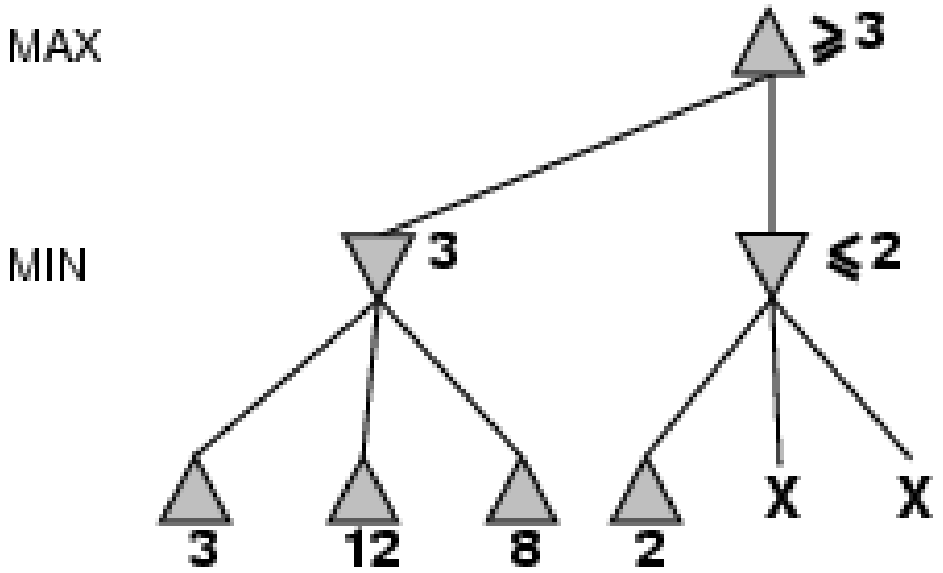


- α es el mejor valor (para `MAX`) encontrado hasta el momento
- Si V es peor que α, `MAX` lo evitará $\longrightarrow$ podar la rama
- β se define igual para `MIN`

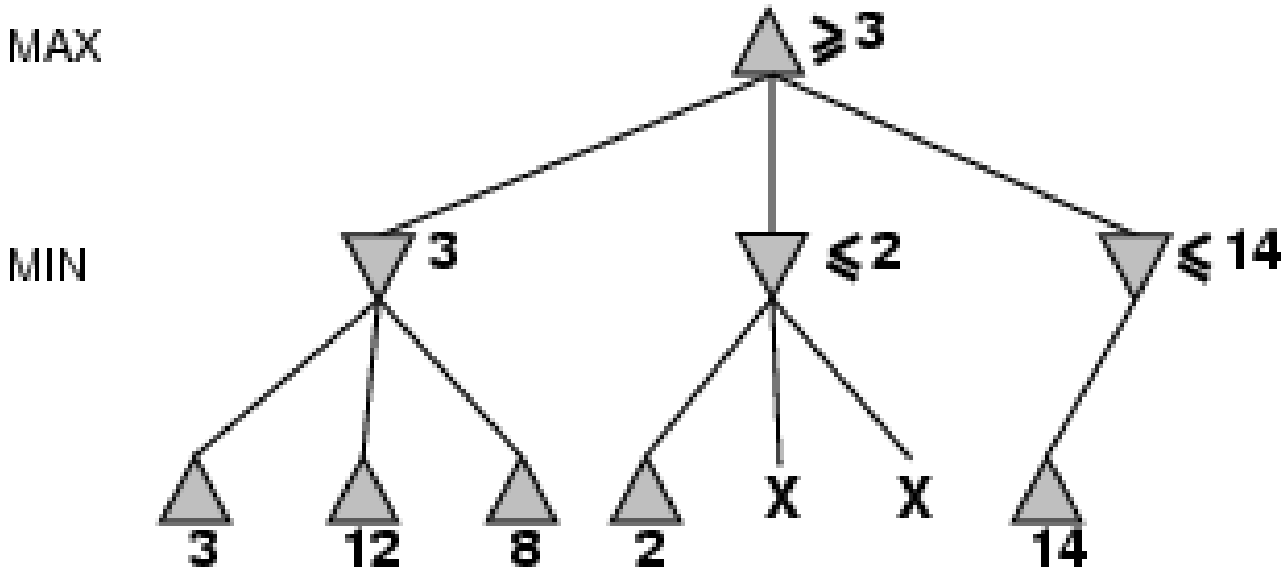
Ejemplo de poda alfa-beta



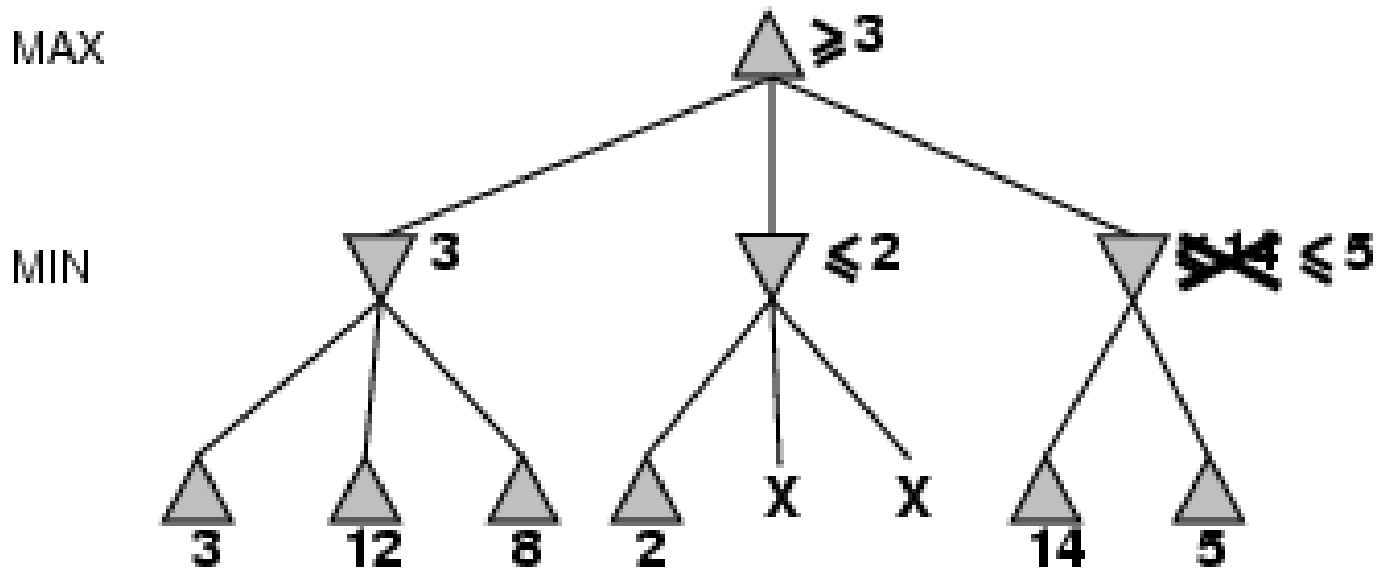
Ejemplo de poda alfa-beta



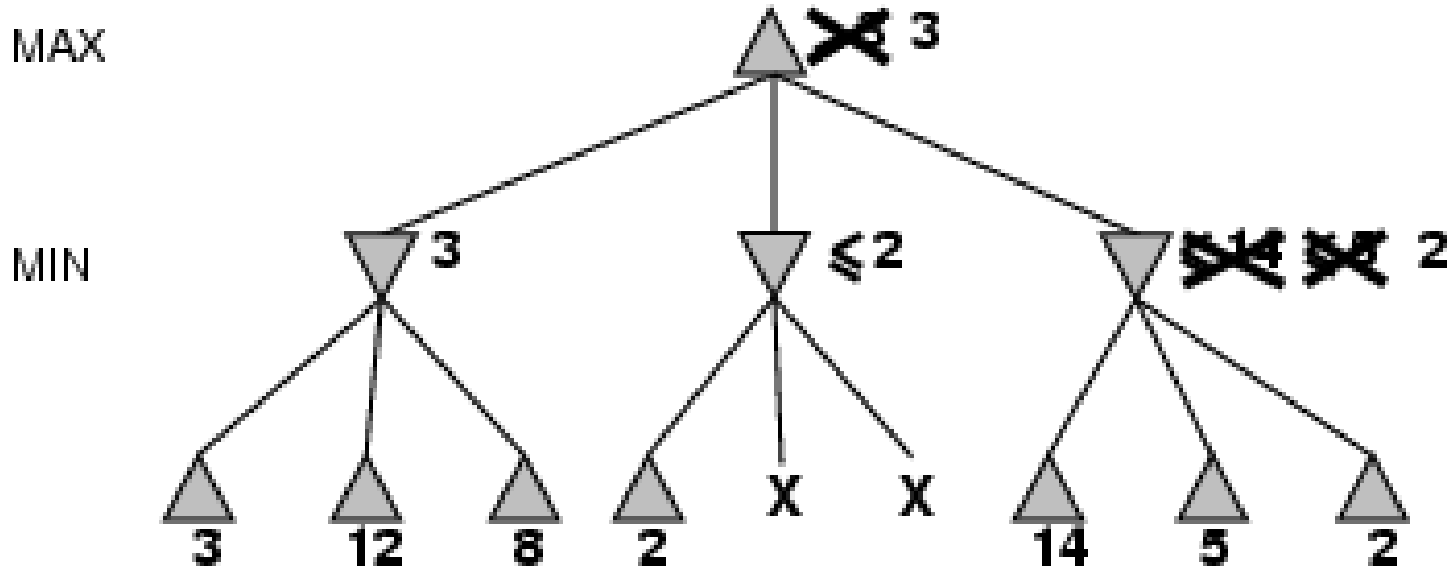
Ejemplo de poda alfa-beta



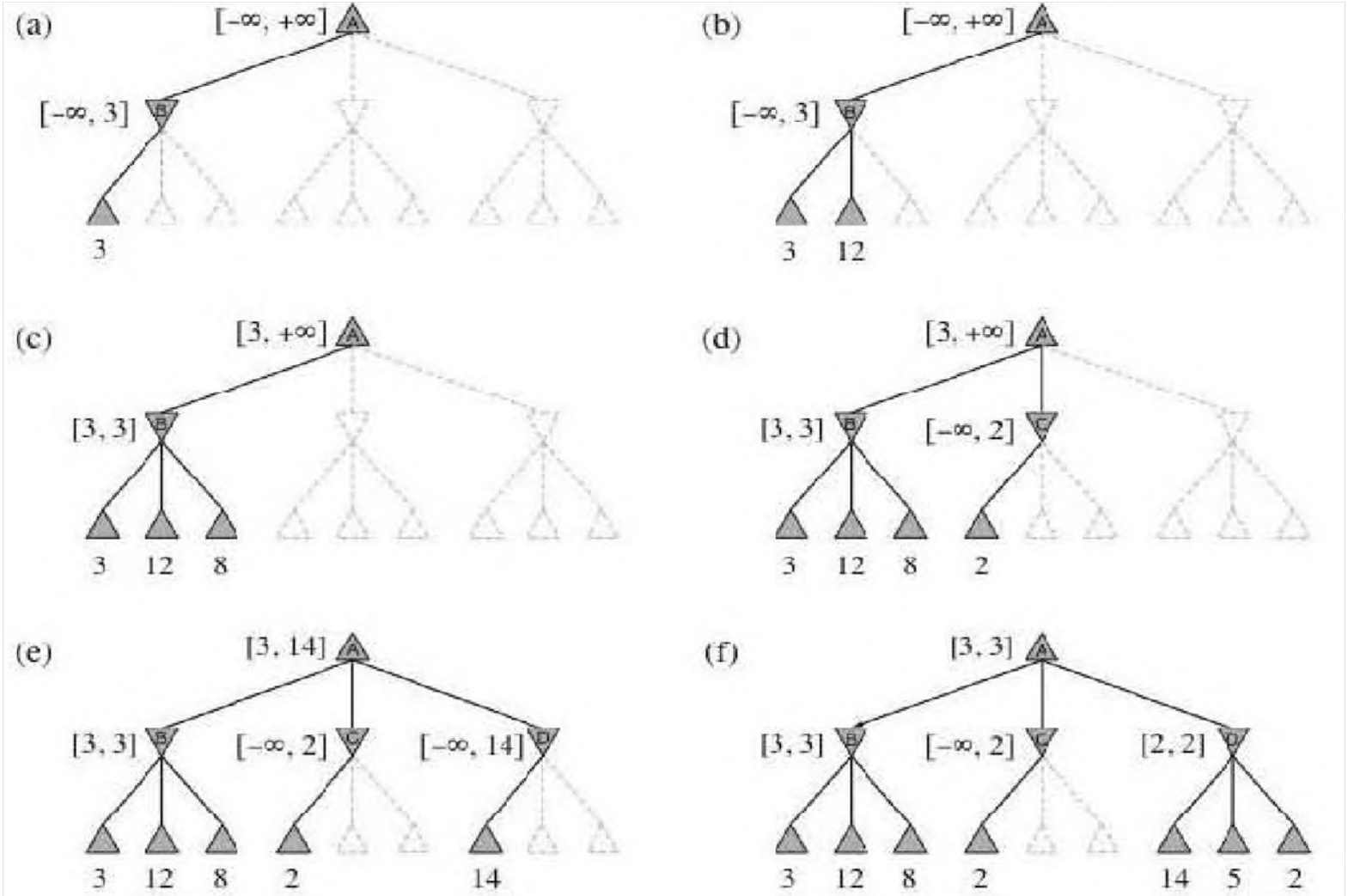
Ejemplo de poda alfa-beta



Ejemplo de poda alfa-beta



Ejemplo de poda alfa-beta



Algoritmo alfa-beta

función BÚSQUEDA-ALFA-BETA(*estado*) **devuelve** una acción
  variables de entrada: *estado*, estado actual del juego

  $v \leftarrow$ MAX-VALOR(*estado*, $-\infty$, $+\infty$)
  devolver la *acción* de SUCESORES(*estado*) con valor v

---

función MAX-VALOR(*estado*,α, β) **devuelve** *un valor utilidad*
  variables de entrada: *estado*, estado actual del juego
    α, valor de la mejor alternativa para MAX a lo largo del camino a *estado*
    β, valor de la mejor alternativa para MIN a lo largo del camino a *estado*

  si TEST-TERMINAL(*estado*) **entonces devolver** UTILIDAD(*estado*)
  $v \leftarrow -\infty$
  para *a*, *s* en SUCESORES(*estado*) **hacer**
  $v \leftarrow$ MAX(V, MIN-VALOR(*s*, α, β))
    si $v \geq \beta$ **entonces devolver** v
    $\alpha \leftarrow$ MAX(α, v)
  devolver v

---

función MIN-VALOR(*estado*, α, β) **devuelve** *un valor utilidad*
  variables de entrada: *estado*, estado actual del juego
    α, valor de la mejor alternativa para MAX a lo largo del camino a *estado*
    β, valor de la mejor alternativa para MIN a lo largo del camino a *estado*

  si TEST-TERMINAL(*estado*) **entonces devolver** UTILIDAD(*estado*)
  $v \leftarrow +\infty$
  para *a*, *s* en SUCESORES(*estado*) **hacer**
    $v \leftarrow$ MIN(v, MAX-VALOR(*s*, α, β))
    si $v \leq \alpha$ **entonces devolver** v
    $\beta \leftarrow$ MIN(β, V)
  devolver v

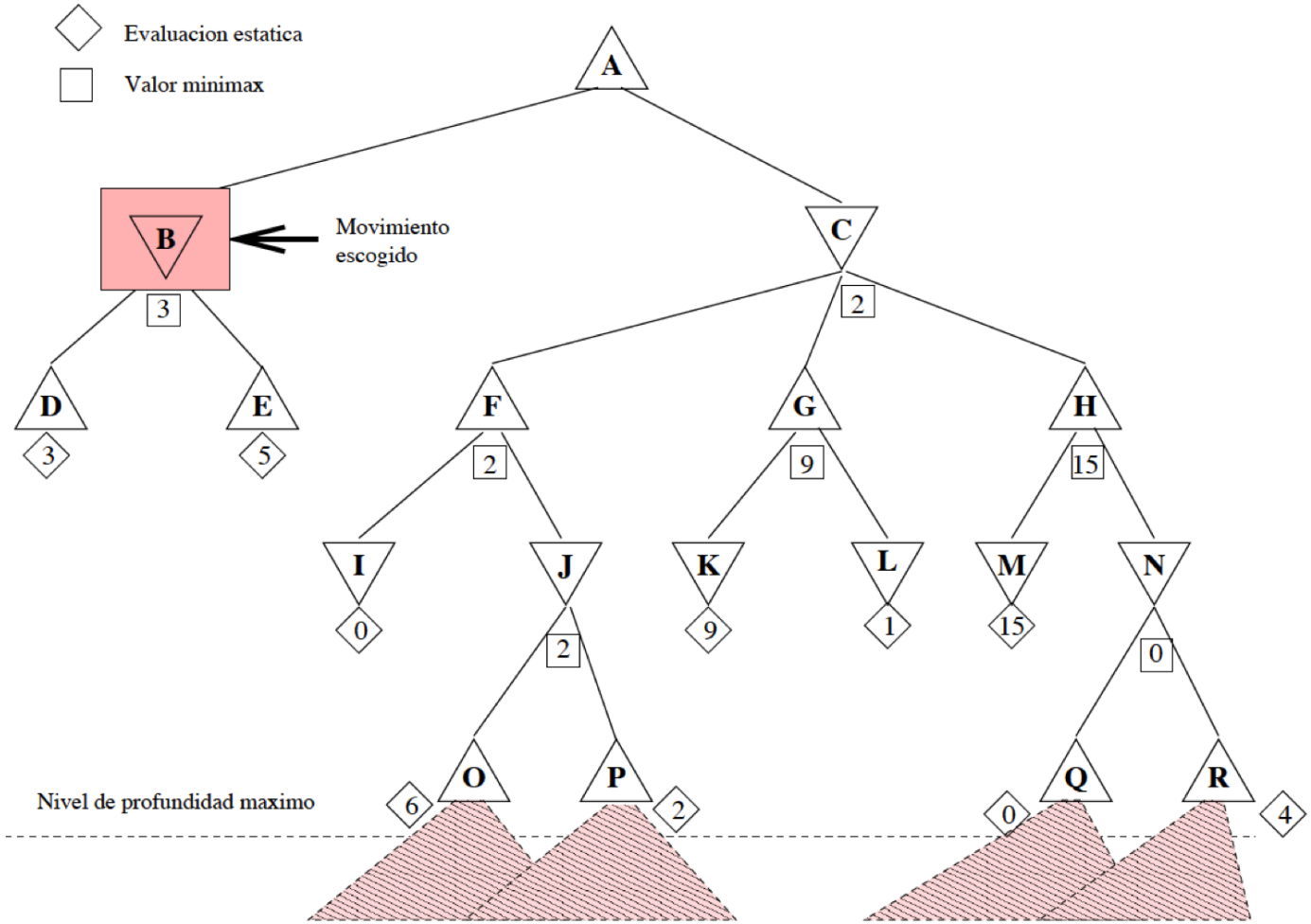
Propiedades de la poda alfa-beta

- La poda no afecta al resultado final.
- Una buena ordenación de los nodos mejora la eficiencia.
- Con una ordenación *perfecta*, la complejidad temporal se reduce a $O(b^{m/2})$, permitiendo explorar árboles del doble de profundidad.
- La poda *alfa-beta* proporciona un ejemplo sencillo de razonamiento sobre qué cálculos son relevantes (una forma de metarazonamiento).

Exploración con recursos limitados

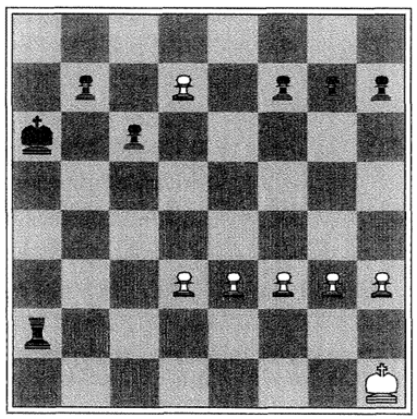
- Cuando el árbol de búsqueda es muy grande, no podemos calcularlo entero.
- Puede ponerse una profundidad límite de exploración.
  - En ese caso, no tendremos la recompensa final (partida ganada o perdida) en cada hoja.
  - Necesitamos una función de evaluación estática (heurística) para buscar las *mejores* posiciones.
  - Efecto horizonte: El algoritmo de búsqueda solo *ve* hasta el límite de profundidad, pudiendo obtener estimaciones *minimax* erróneas.

Árbol *minimax* con función de evaluación estática



Funciones de evaluación

- Deberían ordenar estados terminales igual que la función de utilidad.
- El cálculo no debe utilizar demasiado tiempo.
- En estados no terminales la función de evaluación debe estar fuertemente correlacionada con las posibilidades de ganar.
- La mayoría trabajan calculando varias características del estado (por ejemplo, número de peones capturados).
- Pueden reflejar una proporción de estados con cada resultado.
  - Ej.: el 72 % de los estados encontrados en la categoría conduce al triunfo, el 20 % a pérdida y el 8 % a empate. El valor esperado será una media ponderada.
- El efecto horizonte es difícil de eliminar:



Agente inteligente para ajedrez

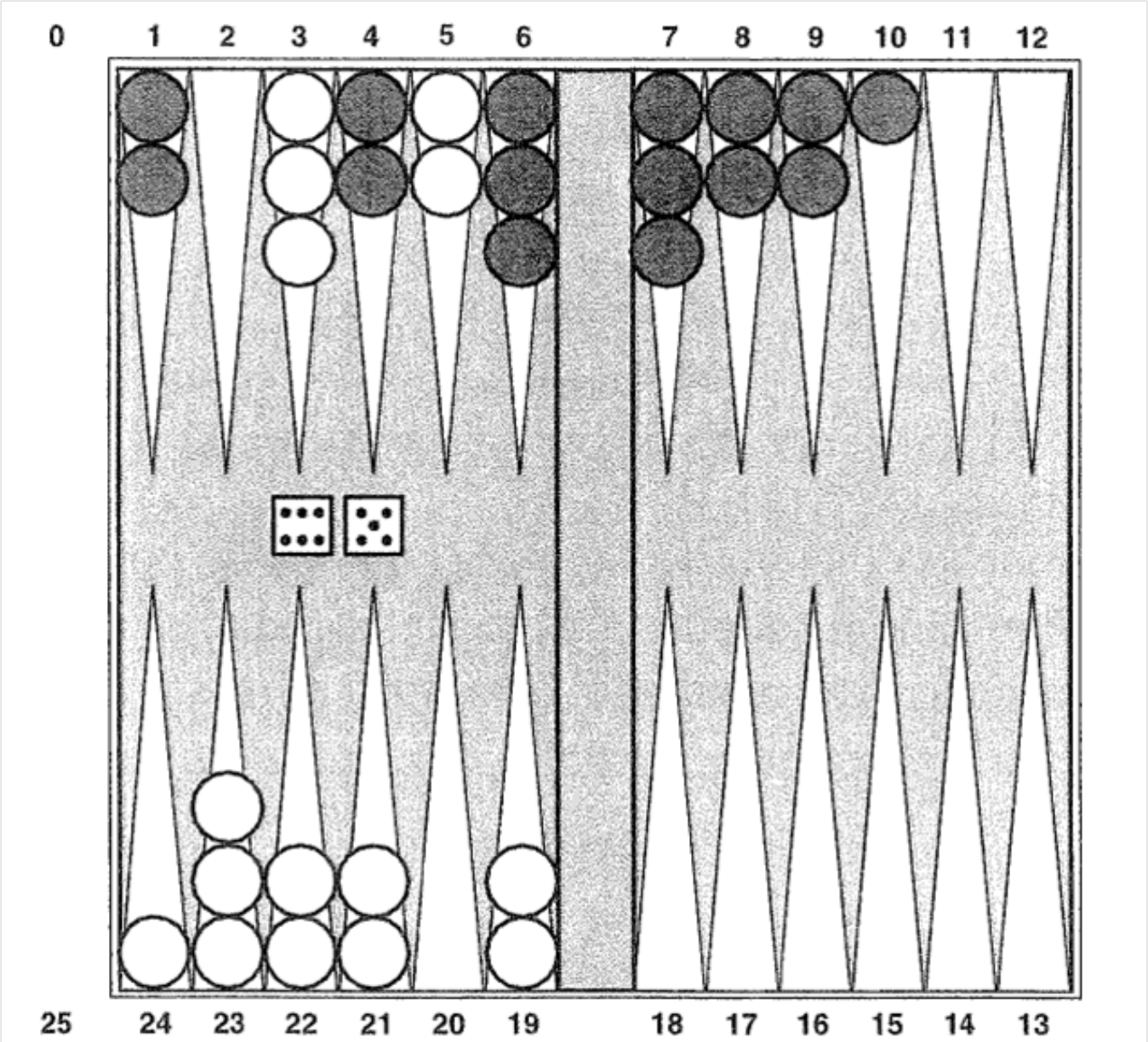
- Función de evaluación.
- Test del límite razonable con búsqueda estable.
- Con una CPU que procesa 200 millones de nodos por segundo podríamos mirar 5 capas (un jugador medio humano puede planear 8 capas).
- Con poda alfa-beta, se aumenta a 10 capas.
- Para el nivel de maestro necesitaríamos una función de evaluación ajustada, una base de datos grande movimientos de apertura y finales óptimos y un supercomputador.

Índice

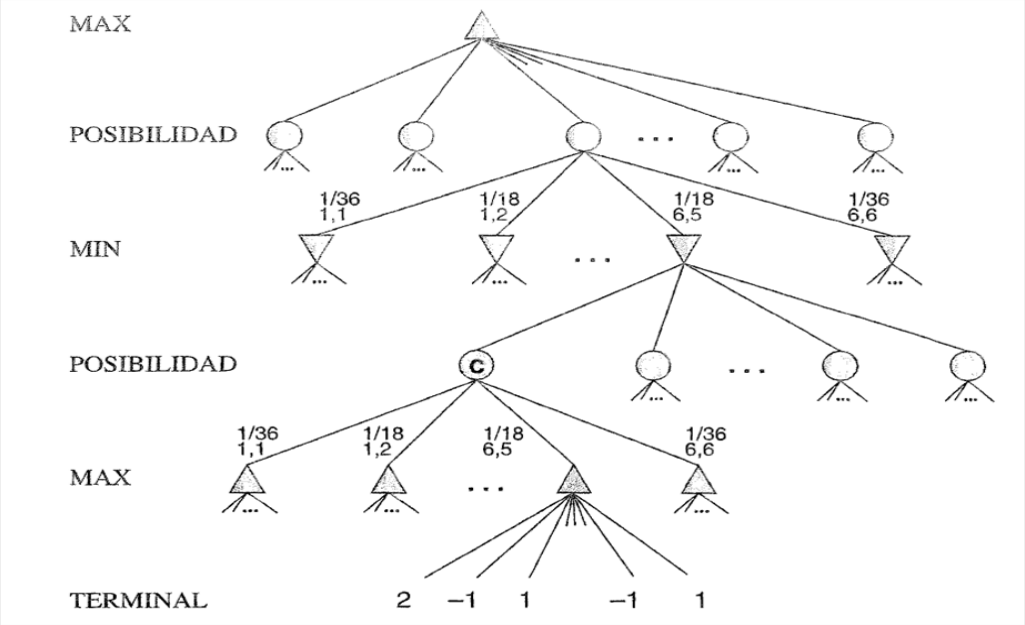
Juegos con elemento de posibilidad

- Muchos acontecimientos imprevisibles nos ponen en situaciones inesperadas.
- Esto puede venir representado en un juego con un lanzamiento de dados.
- Ejemplo: Backgammon
  - 2 dados.
  - Movimientos legales.
  - En el árbol se incluyen los nodos de posibilidad.
  - Se puede calcular el valor esperado, donde la expectativa se toma sobre todos los posibles resultados que podrían ocurrir.
  - Valor *minimaxesperado.*

Juegos con elemento de posibilidad



Juegos con elemento de posibilidad



$$\text{MINIMAXESPERADO}(n) = \begin{cases} \text{UTILIDAD}(n) & \text{si } n \text{ es un nodo terminal} \\ \max_{s \in Sucesores(n)} \text{MINIMAXESPERADO}(s) & \text{si } n \text{ es un nodo MAX} \\ \min_{s \in Sucesores(n)} \text{MINIMAXESPERADO}(s) & \text{si } n \text{ es un nodo MIN} \\ \sum_{s \in Sucesores(n)} P(s) \cdot \text{MINIMAXESPERADO}(s) & \text{si } n \text{ es un nodo posibilidad} \end{cases}$$

Índice

Juegos con movimientos simultáneos de un único movimiento

- Representan el escenario más simple con incertidumbre.
- Elementos:
  - Jugadores: Son los agentes que toman las decisiones.
  - Acciones: Son las diferentes alternativas que cada agente puede elegir en un momento dado.
  - Matriz de balances finales: Recoge la *utilidad* de cada jugador para cada combinación de acciones de todos los jugadores.

	impar	*par*
impar	1	-1
par	-1	1

Matriz de balances finales para el juego *pares o nones* (1: ganan *pares*; -1: ganan *nones*).

Juegos con movimientos simultáneos de un único movimiento

	piedra	papel	tijera
piedra	0	1	-1
papel	-1	0	1
tijera	1	-1	0

Matriz de balances finales para el juego *piedra*, *papel*, *tijera*.

	piedra	papel	tijera	lagarto	Spock
piedra	0	1	-1	-1	1
papel	-1	0	1	1	-1
tijera	1	-1	0	-1	1
lagarto	1	-1	1	0	-1
Spock	-1	1	-1	1	0

Matriz de balances finales para el juego *pierda*, *papel*, *tijera*, *lagarto*, *Spock* (S. Kass y K Bryla).

Estrategias

Definición:

- Una estrategia es una política que permite escoger una acción en un momento dado.
- Tipos de estrategia:
  - Pura: indica de forma determinista qué acción realizar en cada situación.
  - Mixta: proporciona una distribución de probabilidad para cada situación. La acción a llevar a cabo se escogerá aleatoriamente usando dicha distribución.
- Un perfil de estrategia es una asignación concreta de estrategia para todos los jugadores.

El dilema del prisionero

- Dos ladrones, A y B, son detenidos.
- Si uno testifica contra el otro, será puesto en libertad, y el otro condenado a 10 años.
- Si ninguno testifica, ambos serán condenados a un año.
- Si ambos testifican, serán condenados a cinco años cada uno.

	A: *testificar*	A: *rechazar*
B: *testificar*	$A = -5, B = -5$	$A = -10, B = 0$
B: *rechazar*	$A = 0, B = -10$	$A = -1, B = -1$

Matriz de balances finales para el dilema del prisionero.

Suponiendo que el otro prisionero es *racional*, ¿cuál es la mejor opción?

Estrategias dominantes

Análisis del dilema del prisionero

- Razonamiento de A: *Si B testifica, me compensa más testificar, y si B rechaza, también*. Testificar es una estrategia dominante.
- Si todos los jugadores tienen estrategia dominante, la combinación de estas se denomina *equilibrio de estrategias dominantes*.
- Sin embargo, si ambos rechazaran testificar, obtendrían un mejor resultado.
  - Esta situación resulta inverosímil.

Estrategias dominantes

- Fuerte: Si el resultado de la misma es mejor para todas las posibles acciones del contrincante.
- Débil: Si su resultado es el mejor en al menos un perfil de estrategia, y no es peor en cualquier otro.

Situaciones de equilibrio

Equilibrio de estrategias dominantes

- Se da cuando todos los jugadores tienen una estrategia dominante.
- Un perfil de estrategias es un equilibrio si ningún jugador puede beneficiarse por el hecho de cambiar de estrategia.
- Las situaciones de equilibrio son *óptimos locales*.

Equilibrio de Nash

- Todos los juegos presentan estrategias (mixtas) de equilibrio, aunque no haya una estrategia dominante.

Juegos de coordinación y juegos de suma cero

Juegos de coordinación

- Hay juegos que presentan situaciones en las que existe un beneficio mutuo.
- Los jugadores pueden adoptar una estrategia para coordinarse y encontrarlas, o bien comunicarse entre ellos.

Juegos de suma cero

- Son aquellos en los que los valores de cada casilla de la matriz de balances suman cero.
- *Todo juego de suma cero con dos jugadores tiene un equilibrio cuando se permiten estrategias mixtas*.

Índice

Juegos reiterativos

- Los jugadores se enfrentan a la misma elección de forma reiterada, conociendo los movimientos previos de los jugadores.
- Un perfil de estrategia especifica una acción a partir de todas las historias posibles de movimientos anteriores.
- Ejemplos (dilema del prisionero reiterativo):
  - Castigo perpetuo: Se elige *rechazar*, hasta que el contrario nos traiciona, y a partir de entonces siempre se elige *testififcar*.
  - Ojo por ojo: Se elige al principio *rechazar*, y a partir de entonces se escoge la misma acción que eligió el contrario en la anterior ronda.
- Los juegos reiterativos en entornos parcialmente observables se conocen como juegos de información parcial.
  - Ejemplos: *póker*, mus, simulaciones militares, etc.
- En los juegos de información parcial hay que recopilar información sobre la estrategia del contrario, y ocultar la propia (faroles).

Índice

Diseño de mecanismos

- En algunos casos, se conoce el comportamiento racional del agente.
- ¿Es posible, a partir de ese comportamiento, diseñar el juego para que las soluciones parciales de cada agente maximicen una función de utilidad global?
  - Este problema se denomina diseño de mecanismos o teoría inversa de juegos.

- Aplicaciones:
  - Economía, política, ciencias sociales, asignación de recursos, construcción de robots capaces de cooperar.
- Elementos de un mecanismo:
  - Un lenguaje para describir las posibles estrategias de cada agente.
  - La regla G que determina los balances para los agentes, dado un perfil de estrategia.

Diseño de mecanismos

- Puede suponerse que si cada agente maximiza su propia utilidad, en conjunto la utilidad global crece. Falso.
- Tragedia de los mancomunados: El uso de los recursos comunes es beneficioso para un agente, pero puede degradar la utilidad global.
- Este problema corregirse penalizando *(cobrando)* por los recursos comunes.
- Es necesario asignar los precios. Se suele hacer mediante diferentes tipos de subastas:
  - Subasta inglesa (requiere comunicación entre los agentes).
  - Subasta con oferta cerrada (requiere complicados cálculos para *adivinar* las ofertas de otros).
  - Subasta con oferta cerrada del segundo precio o *de Vickrey*.

Índice

Programas de juegos

Ajedrez

- Deep blue (2002)
  - 30 procesadores IBM RS/600 para búsqueda software.
  - 480 procesadores VLSI para búsqueda hardware y generación de movimientos.
  - Buscaba 126 millones de nodos/segundo. Generó hasta 30 billones de posiciones por movimiento y una profundidad de 14 capas.
  - Usaba una alfa-beta de profundidad iterativa y generaba extensiones en líneas interesantes.
- HYDRA: 64 procesadores de 1 Gigabyte basados en FPGA.
- RYBKA, es el mejor jugador artificial. Se conoce poco sobre él.

Programas de juegos

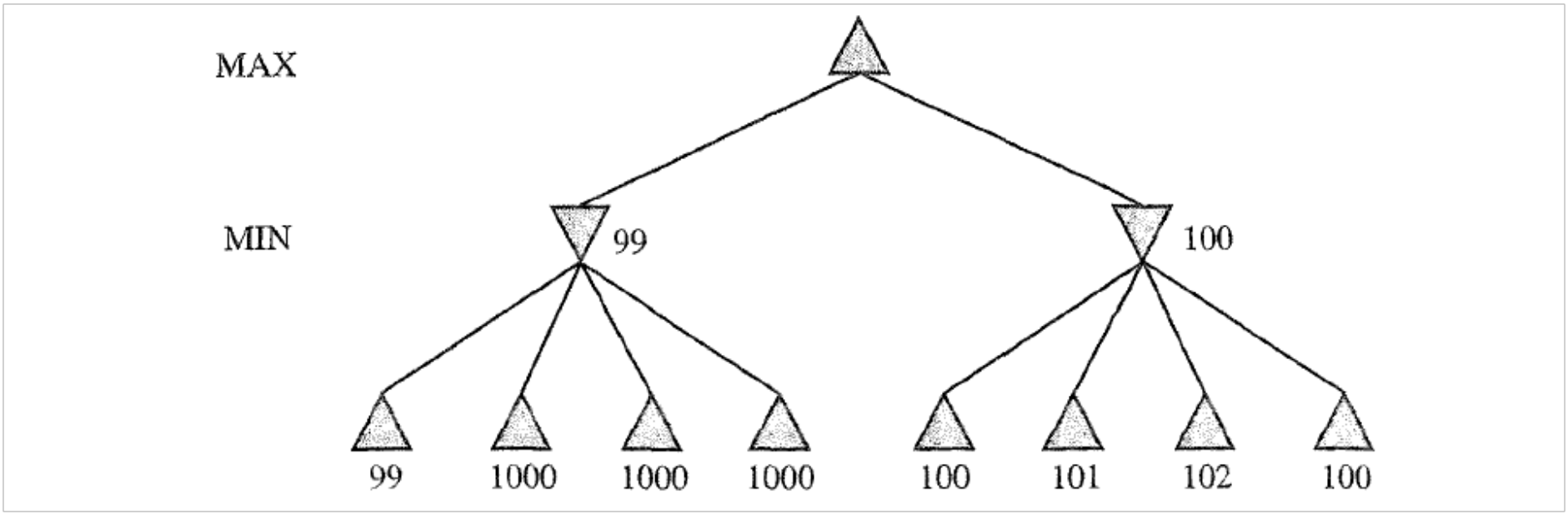
Damas (Checkers): Solucionado en 2007.
http://www.sciencemag.org/content/317/5844/1518.full.pdf

Backgammon

- Mejoras de la función de evaluación.
- TD-GAMMON (1992), es uno de los tres mejores del mundo.

Conecta-4: Solucionado matemáticamente en 1988. El primer jugador puede forzar la victoria empezando en la columna de en medio. Si empieza en una columna exterior, el segundo jugador puede forzar un empate. Si se empieza en otro lado, gana el segundo jugador.

Discusión Minimax



Índice

Bibliografía

- U. Cortés, J. Béjar, A. Moreno. Inteligencia Artificial. Ediciones UPC, 1994.
- J. Mira, A.E. Delgado, J.G. Boticario, F.J. Díez. Aspectos básicos de la Inteligencia Artificial. Sanz y Torres, 1995.
- E. Rich, D. Knight. Inteligencia Artificial (2da. edición). McGraw-Hill Interamericana, 1994.
- S. Russell, P. Norvig. Inteligencia Artificial. Un enfoque moderno. Prentice-Hall, 2004.
- P.R. Winston. Inteligencia Artificial. Add-Wesley, 1994.