irenchuchu
www.wuolah.com/student/irenchuchu

TEMA 4.pdf

APUNTES PARCIAL II

2º Inteligencia Artificial

Grado en Ingeniería Informática

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
UGR - Universidad de Granada

INTELIGENCIA ARTIFICIAL TEMA 4:

BÚSQUEDA CON ADVERSARIO: JUEGOS

Definición búsqueda con adversario: La búsqueda con adversario se caracteriza porque hay un agente deliberativo resolviendo el problema, pero también hay otras personas/objetos que tienen el mismo objetivo que tú y no puedes optimizar el problema porque no puedes controlar a los adversarios.

Interés:

- Los juegos se describen siempre de una forma simple, pero son tremendamente complejos de resolver, son perfectos para resolverlos por medio de la IA.
- Es fácil medir el éxito o el fracaso.
- Fascinación de mucha gente junto con un aspecto comercial.
- Aplicaciones en ámbitos empresariales (se estudia en económicas, empresariales).

Tipo de juego:

- **Bipersonales:** Un jugador vs otro jugador.
- **Información perfecta:** conoces toda la información del juego.
- **Información imperfecta:** Se desconoce una parte de la información del juego. Como por ejemplo el póker.

~ Juegos bipersonales con información perfecta ~

Estas situaciones se estudian y resuelven utilizando la Teoría de Juegos. La teoría matemática de juegos fue inventada como tal por John von Neumann y por Oskar Morgenstern en 1944.

Juego: cualquier situación de decisión entre dos agentes (jugadores), gobernada por un conjunto de reglas y con un resultado bien definido, caracterizada porque ninguno de los jugadores con su sola actuación puede determinar el resultado.

En un juego, cada jugador intenta conseguir el mayor beneficio para sus intereses. La solución de un juego es la determinación de una sucesión de actuaciones que indican a cada jugador el resultado a esperar y cómo alcanzarlo. Por tanto, un juego puede plantearse como un problema de maximización, aunque finalmente y en muchos casos solo pueda llegar a una satisfacción.

Los juegos se dicen de información perfecta si conoces todos tus movimientos posibles y los del contrario.

Complejidad de un juego: número de llamadas que se hacen a la función heurística.

- Perfecto para investigar en técnicas de resolución de problemas.
- Fácil de medir el éxito o el fracaso.
- Fascinación para una amplia variedad de gente.

Elementos de un juego:

- **Estado inicial:** indica la disposición inicial del juego, e identifica al jugar que mueve.
- **Jugadores:** Qué jugador tiene el movimiento en un estado.
- **Acciones:** El conjunto de movimientos legales en un estado.
- **Función resultado:** Define el resultado de un movimiento.
- **Test terminal:** Determina cuando se termina el juego.
- **Función utilidad:** También llamada función objetivo/rentabilidad. Da un valor numérico a los estados terminales.
    - Ejemplo: Ajedrez: Triunfo = 1, Perdida = -1, Empate = 0
        - Suma nula: Las pérdidas o ganancias de un participante son equilibradas con las pérdidas o ganancias del otro participante. Por ejemplo, en el ajedrez puede pasar que uno gana y el otro pierde (1 – 0), lo contrario (0 – 1) o que haya tablas (1/2 – ½)
        - Suma no nula: La pérdida o ganancia de un participante no se equipara a la pérdida o ganancia del otro participante. Se intenta maximizar el beneficio sin que importe si el resto de jugadores gana o pierde.

- **Existencia o no de pagos colaterales.**

EJEMPLO: El dilema del prisionero.

- Dos individuos son detenidos por la policía debido a que cometieron cierto delito. Amos son encerrados en celdas diferentes y son interrogados de forma individual. Ambos tienen dos alternativas: no confesar o delatar al compañero. Saben que, si ninguno confiesa, ambos irán a la cárcel por 2 años, pero si uno delata a su compañero y el otro no, entonces al que confiesa la absuelven y al otro le encierran por 10 años. Si ambos confesasen, entonces la pena se repartiría y ambos irían a prisión por 5 años.

		Prisionero 1	
		No delatar	**Delatar**
Prisionero 2	**No delatar**	(-2, -2)	(0, -10)
	Delatar	(-10, 0)	(-5, -5)

- ¿Qué harán los prisioneros? Con toda lógica: cooperar. Sin embargo, la tentación de hacer la promesa de no delatar, para después traicionar al compañero es muy grande.
- El juego tiene una estructura no cooperativa.

~ Árboles de exploración de juegos ~

Un árbol del juego es una representación explícita de todas las formas de jugar a un juego.

Fase 1: ¿Cómo podría resolver el juego?

Árbol de juego: representación explícita de todas las formas de jugar a un juego. El árbol de juegos contiene todas las partidas (caminos) que se pueden realizar en el juego.

Los dos jugadores utilizan el mismo árbol de juegos. Pero según el que lo mire cambian los arcos Y por O y viceversa. Los movimientos que controlas (los tuyos, por ejemplo, si eres el primero) son arcos O, los que hace el contrario para ti son arcos Y, para cada uno de los movimientos que él pueda hacer tú tienes que GANAR.

Los juegos se abordan con árboles, puesto que algunos algoritmos son más eficientes en árboles que en grafos y los juegos tienen una gran complejidad por lo que necesitan la eficiencia.

Estrategia: secuencia de movimientos que yo voy a realizar y que respuesta le voy a dar a los movimientos del contrario.

Notación min-max:

- MAX: Primer jugador.
- MIN: Segundo jugador.
- Nodos MAX son los que controla el primer jugador.
- Nodos MIN son los que controla el segundo jugador.
- Los nodos terminales se etiquetan con V(victoria), D(derrota) o E(empate) desde el punto de vista de MAX. Normalmente se contempla como victoria para max, derrota para max.

Algoritmo STATUS:

Es un modelo de propagación de información que utiliza un grafo Y/O. Pongo los estados siguientes V o D para ver por donde seguir el camino.

Algoritmo exacto para resolver juegos bipersonales de información perfecta con un requisito, se necesita el árbol del juego completo. No se puede aplicar el algoritmo status para problemas que tienen un algoritmo exacto, pero demasiado complejo como el ajedrez o las damas. Se aplica para juegos simples o para finales de juegos complejos.

- Si J es un nodo MAX no terminal, entonces STATUS(J):
  - V si algunos de los sucesores de J tienen STATUS V.
  - D si todos los sucesores de J tienen STATUS D.
  - E en otro caso.
- Si J es un nodo MIN no terminal, entonces STATUS(J)=
  - V si todos los sucesores de J tienen STATUS V.
  - D si alguno de los sucesores de J tiene STATUS D.
  - E en otro caso.

Fase 2: Nuevo modelo de solución:

Los juegos son complejos no se pueden resolver con un algoritmo status ya que es imposible la exploración total hasta la terminación. Ahora nos centramos en buscar una buena jugada inmediata. Se basa en una heurística.

~ El modelo básico ~

Arquitectura percepción/planificación/actuación.

Búsqueda de horizonte:

- Se establece un horizonte límite en el problema. Se elige un método de búsqueda que explorará hasta llegar al horizonte (búsqueda parcial), cuando se llega al horizonte se aplica una heurística que analiza los nodos del horizonte. Esa heurística se devuelve de alguna manera por una regla de propagación al nodo actual para que éste determine la siguiente acción a realizar.
- Cuanto más lejos esté el horizonte mejor será la respuesta / siguiente movimiento.

Uso de heurísticas:

A.Turing -> B/N

A.Samuel -> damas.

Samuel dijo que, ante un problema complejo, con una gran cantidad de posibilidades de movimiento, no resulta práctico "solucionar" el problema matemáticamente mediante el cálculo de la secuencia perfecta de movimientos sino aprender a jugar dando valoraciones a las jugadas anteriores. La máquina debe de ser capaz de aprender de sus errores y valorar positivamente aquellas jugadas que le han llevado a victoria en otras ocasiones.

Turing fue el primero en desarrollar un algoritmo para que un programa jugase al ajedrez a finales de los 40 y lo publicó en su artículo Digital Computers Applied to Games.

Turing sienta las bazas de los posteriores programas de ajedrez: la simulación de secuencias de movimientos, la evaluación de los estados finales de dichas secuencias y la propagación de dicha evaluación de los estados sucesores del juego.

$$f \quad x1,x2, \ldots\ldots xn \longrightarrow x1, y1 + x2, y2 \ldots + xn, yn$$

La complejidad de un juego se mide por el factor de ramificación (B) del mismo y por la profundidad (P). La complejidad es el número de llamadas que hago a la función heurística B^p.

Búsqueda parcial: Utilizaríamos una búsqueda sin heurística, no se necesita ningún algoritmo complejo, solo se quiere llegar a la frontera. Búsqueda sin información: anchura (con la desventaja de que se tienen que guardar todos los nodos) o búsqueda retroactiva (se llega a la frontera con el menor coste posible).

La regla minimax:

En este algoritmo todo está contemplado desde el punto de vista de max.

- El valor V(J) de un nodo J de la frontera de búsqueda es igual al de su evaluación estática; en otro caso:
- Si J es un nodo MAX, entonces su valor V(J) es igual al máximo de los valores de sus nodos sucesores.
- Si J es un nodo MIN, entonces su valor V(J) es igual al mínimo de los valores de sus nodos sucesores.

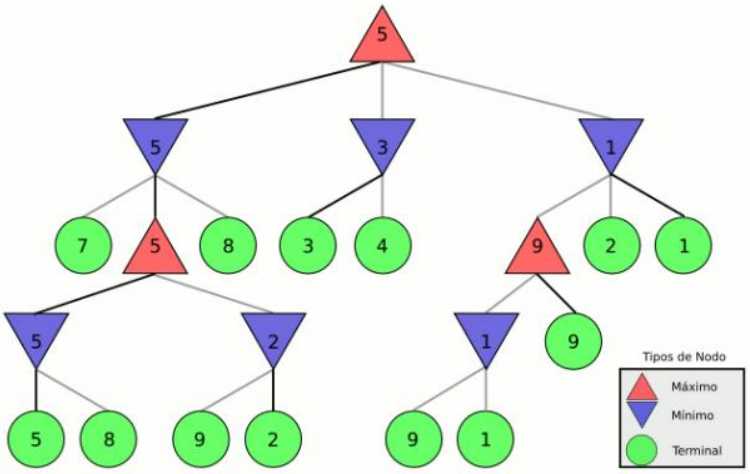
Algoritmo MINIMAX:

El algoritmo de minimax consiste en la elección del mejor movimiento para un jugador (MAX). Suponiendo que el contrincante (MIN) es inteligente y escogerá siempre un movimiento que perjudique al jugador para así beneficiarse él, este algoritmo realiza una búsqueda en profundidad de un árbol para conseguir la victoria o el mejor resultado posible teniendo en cuenta que el adversario siempre elegirá el mínimo valor (de los nodos) de entre sus posibles movimientos y el jugador elegirá el máximo valor posible (de los nodos).

Componentes:

- **Estado inicial:** Es una configuración inicial del juego, es decir, un estado en el que se encuentre el juego. Ejemplo: el ajedrez incluye la posición del tablero e identifica al jugador que mueve.
- **Jugadores:** A los que se denominan Max y Min, siendo Max el jugador que comienza la partida.
- El minimax se aplica a juegos bipersonales únicamente.
- **Operadores:** Corresponden a las jugadas legales que se pueden hacer en el juego, en el caso del tres en raya no puedes marcar una casilla ya antes marcada.
- **Condición terminal:** determina el fin del juego.
- **Función de utilidad:** Da un valor numérico a una configuración final/estado terminal de un juego.

Árbol de juego con todos los estados:



1. Generación del árbol de juego. Se generarán todos los nodos hasta llegar a un estado terminal.
2. Los nodos terminales muestran los valores de utilidad para max.
   Los otros nodos son etiquetados por sus valores minimax.
3. Cálculo de los valores de la función de utilidad para cada nodo terminal.
4. Calcular el valor de los nodos superiores a partir del valor de los inferiores.
   Alternativamente se elegirán los valores mínimos y máximos representando los movimientos del jugador y del oponente.
5. Elegir la jugada valorando los valores que han llegado al nivel superior.

- El algoritmo RECURSIVO explorará los nodos del árbol asignándoles un valor numérico mediante una función de utilidad, empezando por los nodos terminales y subiendo hacia la raíz.

Valoración min-max: el valor del nodo de inicio.

Jugada: siguiente nodo a explorar.

En este algoritmo todo está contemplado desde el punto de vista de max.

Para determinar el valor minimax, V(J) de un nodo J, hacer lo siguiente:

- Si J es un nodo terminal, devolver V(J)=f(J); en otro caso
- Para k=1,2,...,b, hacer:
  - Generar J_k, el k-ésimo sucesor de J
  - Calcular $V(J_k)$
  - Si k=1, hacer $AV(J) \leftarrow V(J_1)$; en otro caso, para k>=2,
  - hacer $AV(J) \leftarrow \max\{AV(J),V(J_k)\}$ si J es un nodo MAX o
  - hacer $AV(J) \leftarrow \min\{AV(J),V(J_k)\}$ si J es un nodo MIN
- Devolver V(J)=AV(J)

Ejemplo del juego 3 en raya:

- Heurística: (nº de filas, clumnas, diagonales aun libres para MAX(x)) – (n1 de filas, columnas, diagonales aun libres para MIN(o))

$$f(T)\begin{cases} +\infty & \text{si T es Victoria} \\ -\infty & \text{si T es Derrota} \\ \text{heurística} & \text{en otro caso} \end{cases}$$

Algoritmo ALFA-BETA:

Requiere que la búsqueda parcial sea un modelo retroactivo.

Se descarta explorar nodos (se poda) que no aporten nada/no modifican el nodo inicial. Simplifica el procedimiento min-max.

	Nodos	Cálculo	Actua
Cota α	MIN	Máximo nodos MAX (raiz)	Cota inferior
Cota β	MAX	Mínimo nodos MIN (raiz)	Cota superior

- Para calcular el valor V(J, alfa, beta), hacer lo siguiente:

1. Si J es un nodo terminal, devolver V(J)=f(J). En otro caso, sean $J_1,...,J_k,...,J_b$ los sucesores de J. Hacer k ←1 y, si J es un nodo MAX ir al paso 2; si J es un nodo MIN ir al paso 5.
2. Hacer alfa ← max(alfa, $V(J_k$,alfa,beta)).
3. Si alfa >= beta devolver beta; si no, continuar
4. Si k=b, devolver alfa; si no, hacer k ← k+1 y volver al paso 2.
5. Hacer beta ← min(beta, $V(J_k$,alfa,beta)).
6. Si beta <= alfa devolver alfa; si no, continuar
7. Si k=b, devolver beta; si no, hacer k ← k+1 y volver al paso 5.

Complejidad en poda alfa-beta.

- Peor caso: no se poda nada, B^p igual que en min-max.
- Mejor caso: B^p/2.
- Caso promedio: se podría profundizar un 33% más.
- No es fácil determinar la complejidad porque a priori no se puede saber el número de podas que se van a realizar, por eso se estudiar 3 posibles casos: el peor, el mejor y el caso promedio.
- Otros datos:
    - Shannon dividió los juegos en dos tipos:
        - **Programa tipo A:** Modelo establecido con min-max, es un modelo deficiente con problemas porque no alcanzan completamente el pensamiento humano. Capacidad de cálculo muy grande, gana al humano en computación mientras que el ser humano gana en conocimiento.
        - **Programa tipo B:** Humano hace selección en el análisis de elección de las jugadas. Profundidad fija generará problemas.

    - Shannon describió dos tipos de estrategias para el juego de ajedrez por máquinas.
    - La más primitiva, llamada "tipo-A", consistía en la búsqueda en profundidad con un límite a lo largo de cada "rama" del árbol, para luego utilizar el algoritmo "minmax".
    - La estrategia "tipo-B" introdujo un método de búsqueda selectiva en el cual ciertas líneas eran más profundizadas que otras.
    - El mismo Shannon acotó que la estrategia "tipo-A" presenta al menos tres desventajas. En una posición de medio juego hay cerca de 30 posibles movimientos. Después de una movida de cada bando las posibilidades crecen a 1000 nodos terminales los cuales deben ser evaluados. Luego de tres movimientos la proporción crece a 3x3 nodos terminales, lo cual implica que a una tasa de evaluación de 1 por cada microsegundo el programa tardaría al menos 16 minutos para cada movimiento.
    - Segundo, un jugador de ajedrez examina algunas variantes con profundidad de pocos niveles y otras con cerca de 20 niveles de profundidad. Esto es lo que se llamó "búsqueda selectiva" en la estrategia "tipo-B".
    - La tercera desventaja es el problema derivado de las posiciones "inestables", aquellas en donde hay cambios de piezas o bien secuencias de jaques.
    - Referido a esto último, Shannon introdujo el concepto de "estabilidad", la cual aplicó a su segundo tipo de estrategia, proponiendo lo siguiente:
        1. Examinar variantes forzadas lo más profundamente posible, evaluando sólo posiciones estables.
        2. Seleccionar mediante un proceso externo las variantes importantes de ser analizadas, evitando perder tiempo analizando variantes inútiles.

    - Estos dos criterios son claramente utilizados por jugadores humanos, siendo el segundo nada más que el algoritmo alfa-beta, el cual fue utilizado en programas a partir de 1960.

- Efecto horizonte: Problema en algoritmos de búsqueda de profundidad fija.
    - El juego analiza hasta un horizonte fijo. El ordenador entiende que, en el horizonte, los nodos terminales son el fin del juego, (si lo sería en el status) pero en este caso el horizonte no significa el fin, aunque el ordenador lo entienda así. Esto lleva al ordenador a realizar movimientos que un humano no haría, pensando que finaliza el juego. Los beneficios solo se consiguen hasta el horizonte. Aumentar el horizonte mejora el problema.

- Ordenación:
    - Ordenar la poda alfa-beta, los movimientos, siguiendo un criterio de ordenación.
    - Una posible ordenación sería ordenar por los movimientos de refutación (reaccionar al último movimiento).
    - Otros modelos de ordenación son la ordenación fijada y la ordenación dinámica.
        - **Ordenación fijada:** se ordenan los nodos con la función heurística intentando aproximarse al mejor caso de la poda alfa-beta. Algoritmo min-max retroactiva pero que ordena con una heurística.
        - **Ordenación dinámica:** se utiliza una variación del algoritmo A* como búsqueda parcial (algoritmo SSS*) y el resto la poda alga-beta. Este algoritmo mejoraba las podas (más eficiente) con respecto al algoritmo alfa-beta pero tiene un desventaja, la complejidad de memoria del algoritmo A* (muy costoso para juegos complejos).

~ Juegos en los que interviene un elemento aleatorio ~

¿Qué pasa cuando hay información aleatoria o falta de información?

Juegos de cartas, juego backgammon.

Sobre el juego backgammon:

- Siguen una filosofía parecida a las vistas anteriormente, sigue existiendo max-min pero los movimientos no se conocen hasta que no se tira el dado.
- Se juega a lo que te permite la incertidumbre, al promedio. Cada nodo tiene su propia probabilidad. Cuando se propaga a un nodo, se propaga la media ponderada de todos sus hijos.
- La incertidumbre genera muchas más situaciones que explorar por lo que son más difíciles de analizar. Es más difícil a su vez crear una buena heurística ya que es preciso el valor exacto y no una ordenación.