Wprowadzenie do Sztucznej Inteligencji (WSI). Notatki wygenerowane automatycznie ze slajdów.

Wprowadzenie

Teoria gier - dział matematyki i ekonomii. Nagrody Nobla:

- 1994 (Nash, Selten, Harsanyi)
- 1996 (Vickrey, Mirrleesem)
- 2005 (Scheling, Aumann)
- 2007 (Hurwicz, Maskin, Myerson)

Będziemy się zajmować grami:

- w postaci ekstensywnej,
- o sumie stałej (o sumie zerowej),
- sprawiedliwymi,
- dwuosobowymi,
- o skończonym czasie rozgrywki.

Poszukiwanie strategii w takich grach jest problemem przeszukiwania przestrzeni.

Plan wykładu

- Definicje
- Algorytmy dla dwuosobowych gier deterministycznych z pełną informacją
  - Przegląd wyczerpujący
  - Algorytm MiniMax
  - Algorytm $\alpha - \beta$
- Algorytmy dla dwuosobowych gier niedeterministycznych

Gry dwuosobowe

Podział ze względu na czas (kolejność) podejmowania decyzji:

- gry w postaci strategicznej (normalnej)

  gracze podejmują decyzje jednocześnie, bez wiedzy o decyzjach innych uczestników
- gry w postaci ekstensywnej (rozwiniętej)

  gracze podejmują decyzje sekwencyjnie, mając określone informacje o decyzjach

Gry o sumie zerowej

Gry o sumie zerowej - zysk gracza oznacza stratę oponenta, interesy graczy są dokładnie przeciwstawne.

	Deterministyczne	Hazardowe
Informacja pełna	• szachy • warcaby • go	• tryktrak (backgammon) • monopoly
Informacja niepełna	• statki	• brydż • poker • scrabble

Deterministyczne gry dwuosobowe

- Dwaj przeciwnicy
- wykonują posunięcia naprzemiennie
- na przebieg gry nie wpływa element losowy
- stan gry jest znany obu przeciwnikom
- koniec gry ma przypisany wynik - jest to liczba rzeczywista nazywana wypłatą

Cel:

- dla gracza rozpoczynającego (nazywanego Max) - maksymalizacja wypłaty
- dla oponenta (nazywanego Min) - minimalizacja wypłaty

Deterministyczna gra dwuosobowa: model

$$\langle S, P, s_0, T, w \rangle$$

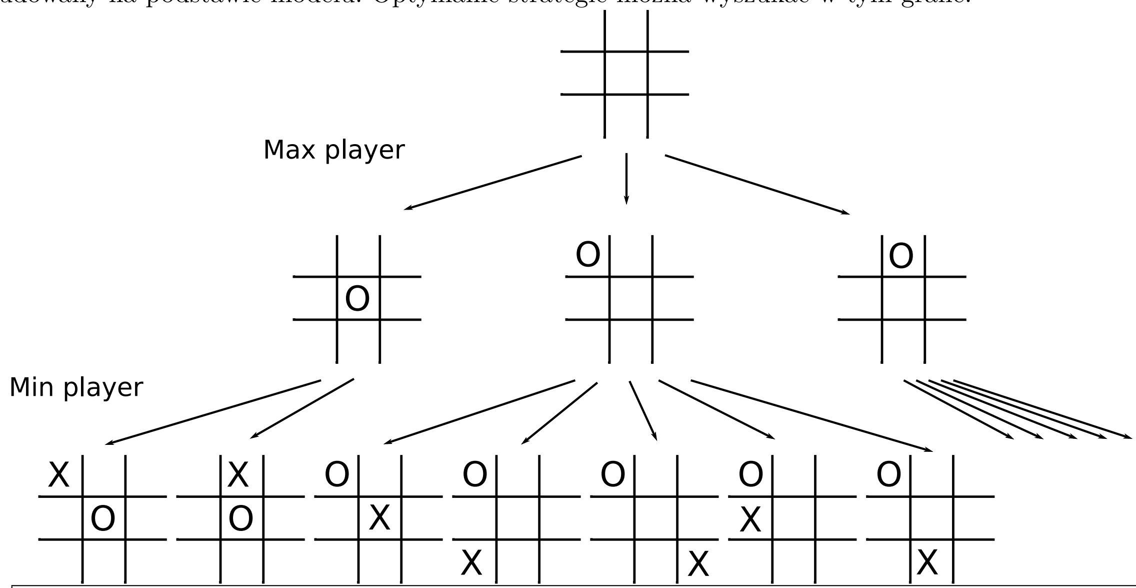
gdzie:

- $s \in S$ to stan i informacja kto wykonuje ruch, , np. ustawienie figur na szachownicy
- $p \in P$ funkcja następnika, reprezentuje ruchy (posunięcia w grze), $p : S \to S$ lista lista stanów spełniających reguły gry
- $s_0 \in S$ to stan początkowy, np. ustawienie początkowe figur na szachownicy
- $T \subseteq S$ -to zbiór stanów terminalnych, np. mat w szachach
- w to funkcja wypłaty zdefiniowana dla stanów terminalnych $s \in T$, np.

$$w(s) = \begin{cases} 1 & \text{zwycięstwo gracza} \\ 0 & \text{remis} \\ -1 & \text{przegrana} \end{cases}$$

Drzewo gry kółko i krzyżyk

Do wyznaczania następnego ruchu gracze mogą wykorzystywać graf acykliczny (nazywany drzewem gry) zbudowany na podstawie modelu. Optymalne strategie można wyszukać w tym grafie.



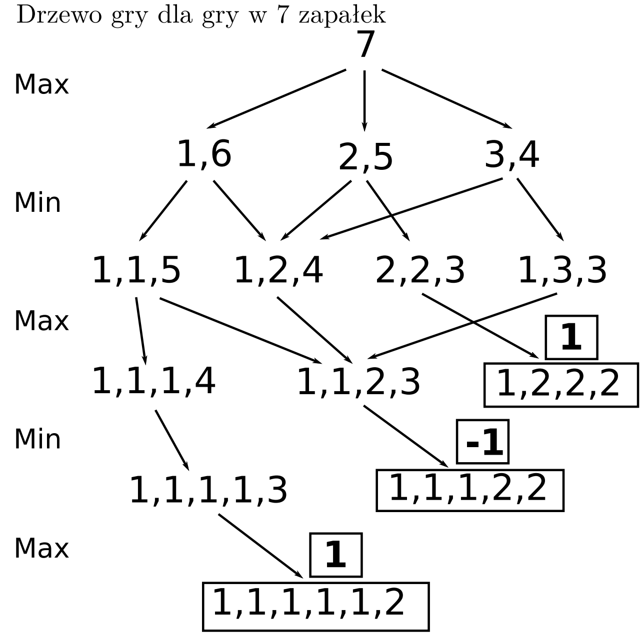
Ćwiczenie: dokończ drzewo gry w kółko i krzyżyk

Przykład: gra w zapałki

Zasady gry:

- początkowo istnieje jedna grupa N zapałek
- gracze na przemian dzielą dowolną grupę na nierówne części
- przegrywa gracz, który nie może wykonać ruchu

Drzewo gry dla gry w 7 zapałek



Przegląd wyczerpujący

Gracz wybiera ruch zapewniający największą wypłatę przy założeniu, że przeciwnik gra optymalnie.

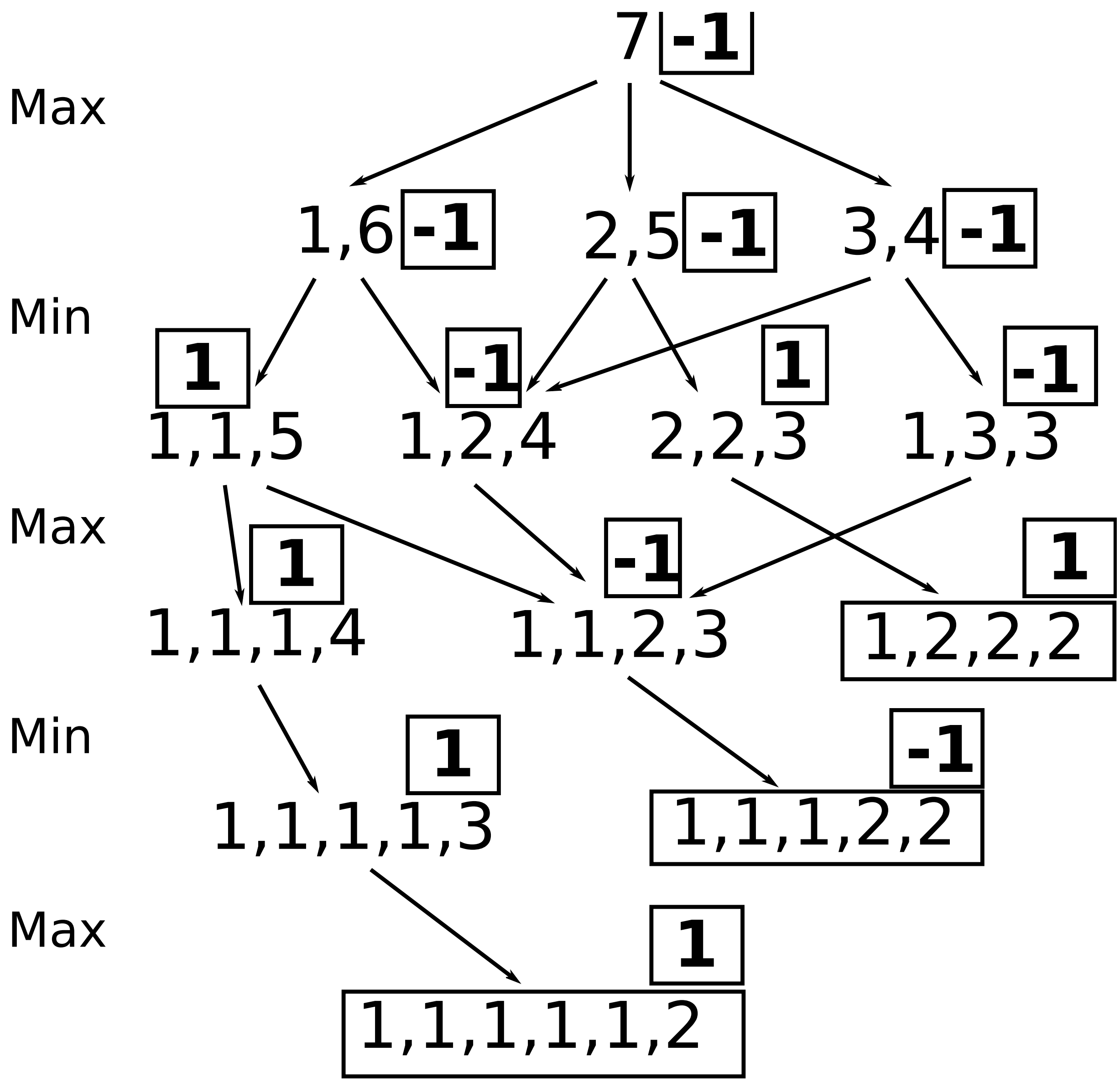
Dane wejściowe: drzewo gry, $w(s)$ dla $s \in T$

Algorithm 1: Przegląd wyczerpujący

```
def MinimaxFull(s)                                  // początkowo s = s_0
if s ∈ T then
|  return w(s)                                      // węzeł terminalny, wypłata
end
U := successors(s)
for u in U do
|  w(u) = MinimaxFull(u)
end
if max-move then                                    // ruch gracza Max
|  return max(w(u))
else
|  return min(w(u))
end
```

Przykład: przegląd wyczerpujący dla gry w zapałki

Wypłaty dla gry w 7 zapałek, $w(x) = \begin{cases} 1 & \text{wygrywa gracz} \\ -1 & \text{wygrywa oponent} \end{cases}$



Przegląd wyczerpujący - własności

- Znajduje optymalną strategię, gdy przeciwnik gra optymalnie.
- Złożoność czasowa $O(b^N)$, gdzie - b maksymalne rozgałęzienie drzewa, N maksymalna wysokość drzewa
- Wysokość drzewa nie przekracza $2N$, gdzie N liczba poprawnych konfiguracji
- dla gry w szachy $b \approx 35$, $N \approx 100$ dokładne rozwiązanie nieosiągalne

algorytm MiniMax, założenia

Podobny do przeglądu wyczerpującego, ale:

- nie reprezentuje pełnego drzewa gry, stany potomne są analizowane niezależnie
  - oszczędza czas, bo nie bada, czy czy do stanu dochodzi inna, wcześniej rozważana, sekwencja ruchów

  - może wielokrotnie analizować te same stany,
  - graf acykliczny jest teraz drzewem

- pozwala analizować ścieżki o ograniczonej długości.

Wymaga dostarczenia funkcji oceny stanu $h(s)$

$$h(s) = \begin{cases} w(s) & \text{dla } s \in T \\ \text{heurystyka} & \text{dla pozostałych} \end{cases}$$

Heurystyczna funkcja oceny sytuacji w grze

Przykład heurystyki dla gry kółko i krzyżyk to suma liczba punktów za pola zgodnie z macierzą:

3	2	3
2	4	2
3	2	3

Macierz pokazuje w ilu konfiguracjach potencjalnie bierze udział dane pole.

$$h\left(\begin{array}{|c|c|c|}\hline O & X & \\ \hline & O & \\ \hline & & X \\ \hline\end{array}\right) = 2 \qquad h\left(\begin{array}{|c|c|c|}\hline O & & \\ \hline & O & \\ \hline X & & X \\ \hline\end{array}\right) = 1 \qquad h\left(\begin{array}{|c|c|c|}\hline O & & \\ \hline & O & X \\ \hline & & X \\ \hline\end{array}\right) = 2$$

gracz Max gra kółkiem, gracz Min gra krzyżykiem.

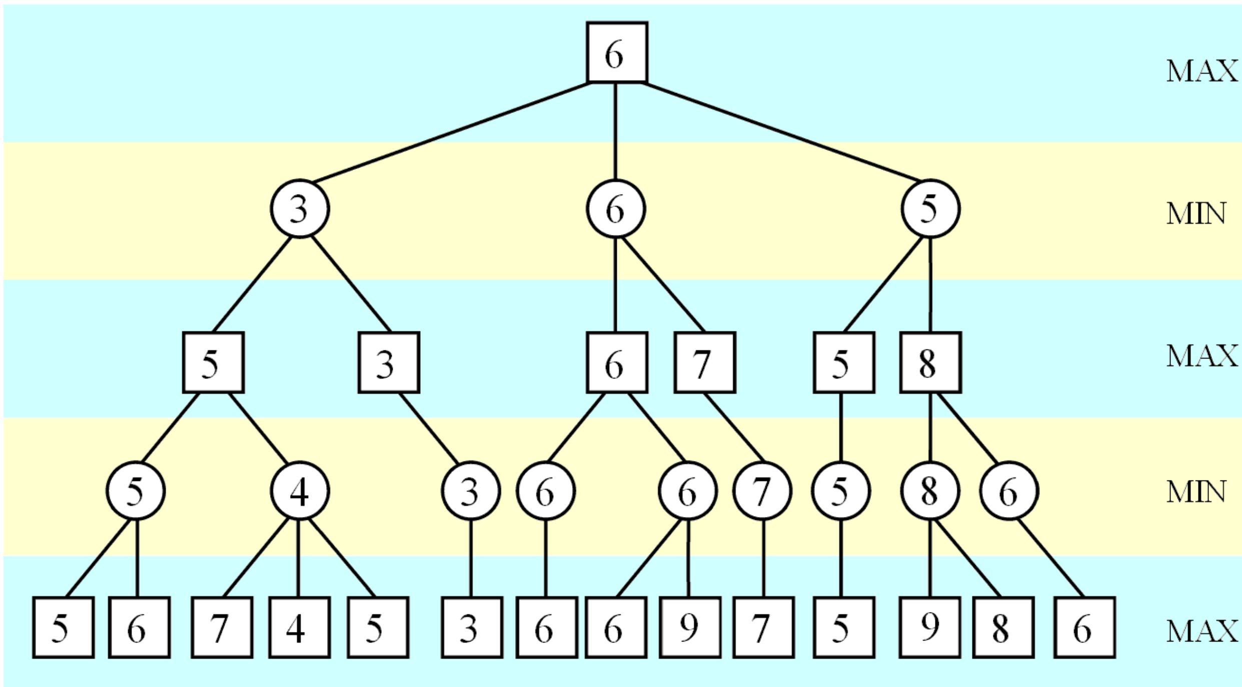
Algorytm Minimax

Gracz wybiera ruch zapewniający największą wypłatę, patrząc na d ruchów naprzód, zakładając, że przeciwnik gra optymalnie.

Algorithm 2: Algorytm minimax

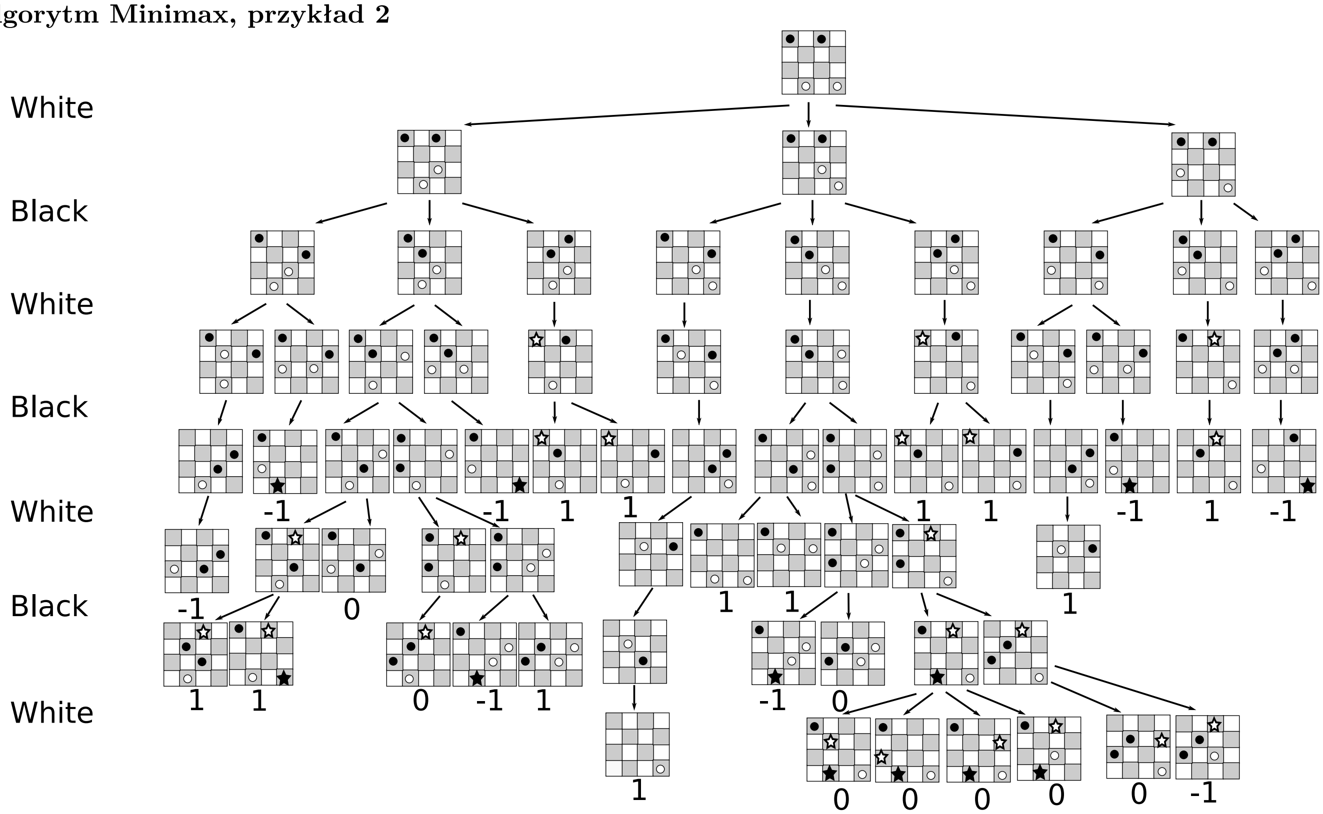
```
def Minimax(s,d)                                // d - głębokość przeszukiwania
if s ∈ T  or  d = 0 then
  | return h(s)                                 // heurystyka lub wypłata
end
U := successors(s)
for u in U do
  | w(u) = Minimax(u, d-1)
end
if Max-move then                                // ruch gracza Max
  | return max(w(u))
else
  | return min(w(u))
end
```

Algorytm Minimax, przykład



Źródło: wikipedia

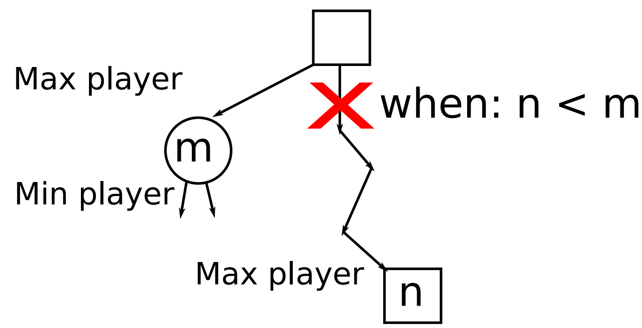
Algorytm Minimax, przykład 2



Algorytm $\alpha - \beta$

Algorytm Minimax z funkcją odcięcia $\alpha - \beta$

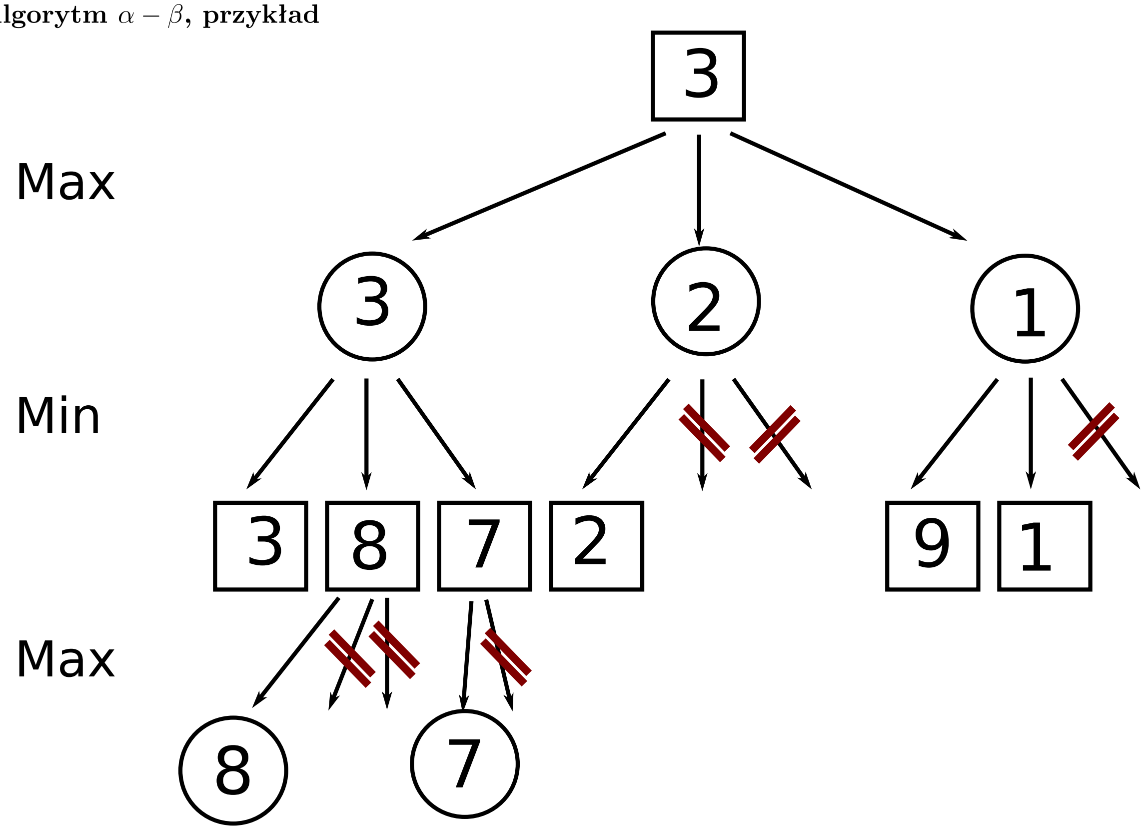
- Jeżeli analizowana ścieżka ma wybór gorszy niż obecnie najlepszy dla innej ścieżki, to nie ma sensu jej analizować.
- taka strategia pozwala wyeliminować średnio połowę ścieżek przy przeszukiwaniu



Oznaczenia:

- α - najlepszy obecnie wybór dla Max
- β - najlepszy obecnie wybór dla Min

Algorytm $\alpha - \beta$, przykład

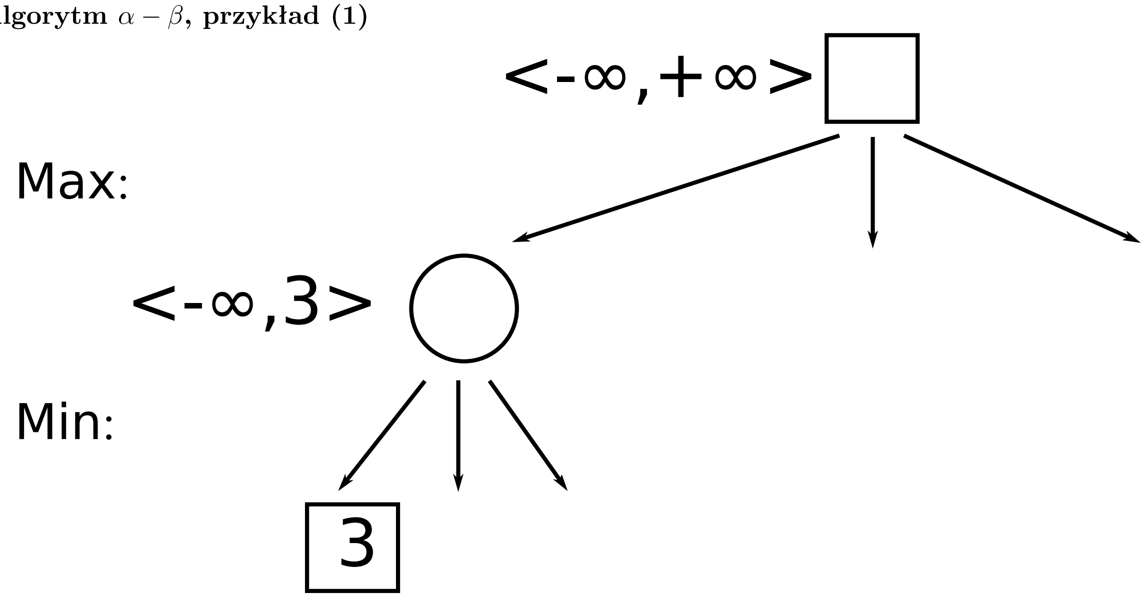


Algorytm $\alpha - \beta$

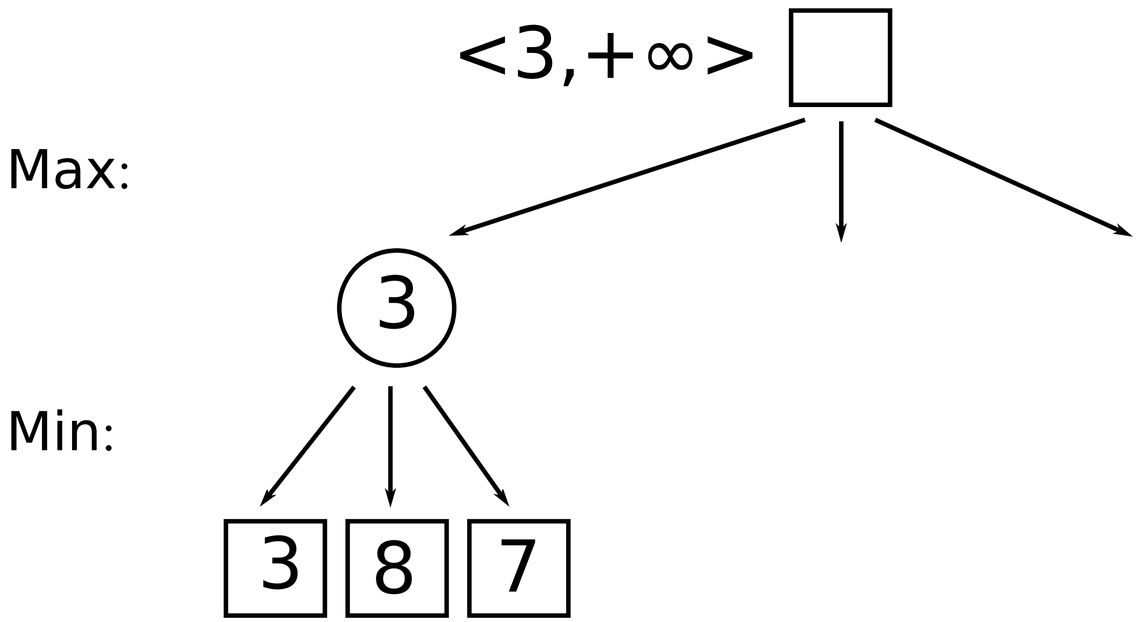
Algorithm 3: Algorytm Minimax $\alpha - \beta$

```
def AlfaBeta(s,d,α,β)                                  // α = −∞, β = +∞
if s ∈ T  or  d = 0 then
│ return h(s)                                          // heurystyka lub wypłata
end
U := successors(s)
if Max-move then                                       // ruch gracza Max
│ for u in U do
│ │ α := max(α, AlfaBeta (u, d-1, α, β) )
│ │ if α ⩾ β then
│ │ │ return β
│ │ end
│ end
│ return α
else
│                                                      // analogicznie dla oponenta, gracza Min
end
```

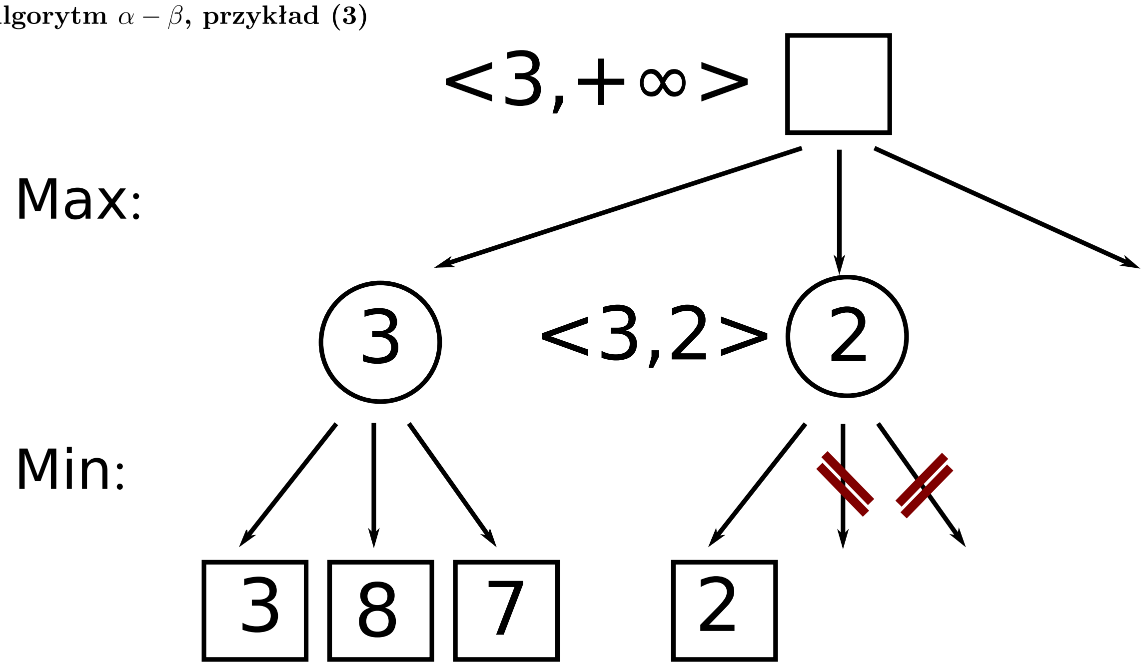
Algorytm $\alpha - \beta$, przykład (1)



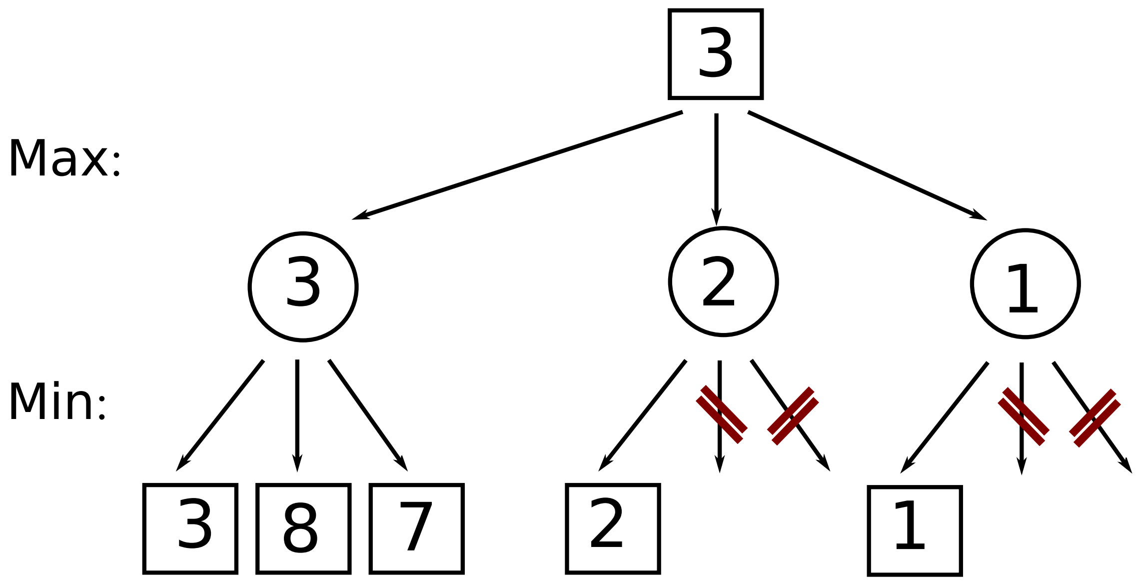
Algorytm $\alpha - \beta$, przykład (2)



Algorytm $\alpha - \beta$, przykład (3)



Algorytm $\alpha - \beta$, przykład (4)



Deterministyczne gry dwuosobowe, algorytmy wspomagające

- iteracyjne pogłębianie
- heurystyki określające kolejność analizy ruchu
- heurystyki określające dokładność oszacowania wartości stanu
- książka otwarć
- książka zamknięć

Deterministyczne gry dwuosobowe, stan obecny

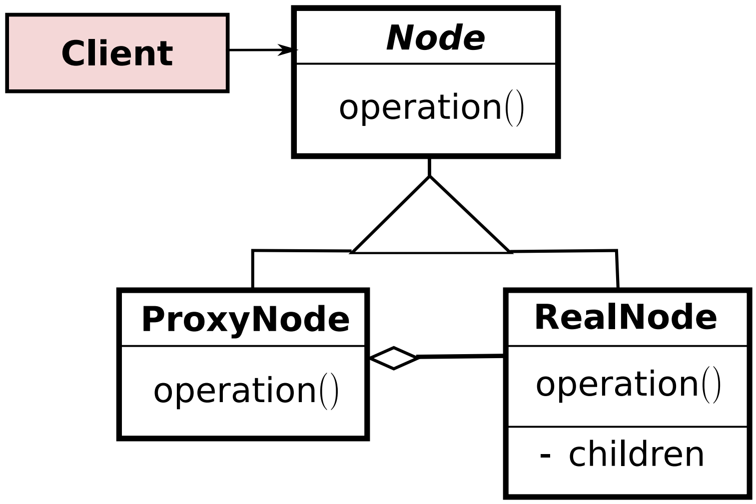
- Warcaby: 1994 program Chinook wygrał z mistrzem świata. Program używał wstępnie obliczone końcówki gry dla 444’000’000’000 pozycji.
- Szachy: 1997 komputer DeepBlue pokonał arcymistrza świata w sześciu partiach, wynik meczu $3\frac{1}{2} : 2\frac{1}{2}$. DeepBlue wykorzystywał algorytm $\alpha - \beta$, obliczenia na klastrze zawierającym 30 CPU i specjalizowane układy do gry w szachy.
- Go: 2016, program AlphaGo wygrał 4:1 z jednym z najlepszych zawodowych graczy. Wykorzystuje sztuczne sieci neuronowe i klaster z 1920 CPU i 280 GPU.
- AlphaGo Zero w 2017 r program wygrał 100:0 z AlphaGo. AlphaGo Zero uczył się grając sam ze sobą. Używa TPU (Tensor Processing Unit), procesory wspierające sprzętowo operacje charakterystyczne dla sieci neuronowych.

Gry a poszukiwania

Przeszukiwanie	Gry
• nie ma przeciwnika • heurystyki pomagają znaleźć rozwiązanie optymalne	• są przeciwnicy • rozwiązanie to strategia gry, odpowiedź na każde działanie przeciwnika

Zagadnienia związane z implementacją

Struktura danych do reprezentacji gry często stosuje wzorzec Virtual Proxy, (leniwe tworzenie, tworzenie przy pierwszym użyciu).

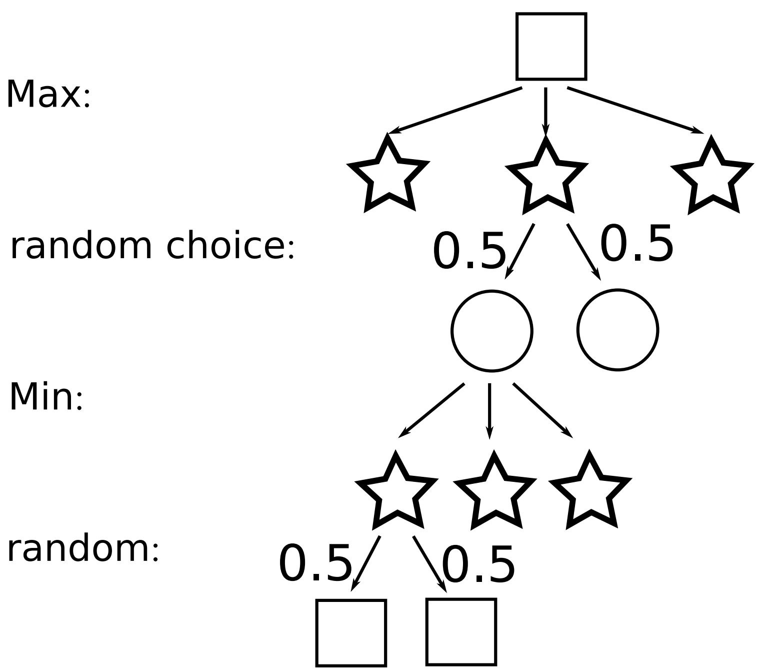


Gry niedeterministyczne z pełną informacją

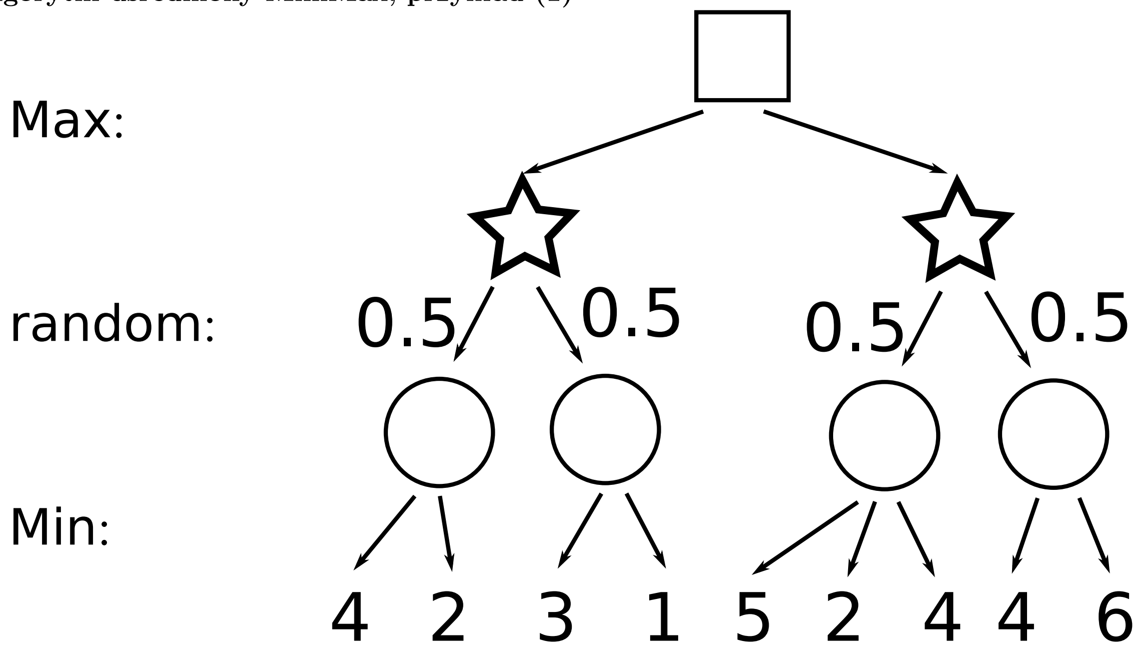
Algorytm uśredniony MiniMax(Expected MiniMax)

- drzewo ma węzły losowe
- wypłata dla węzłów losowych uwzględnia prawdopodobieństwo

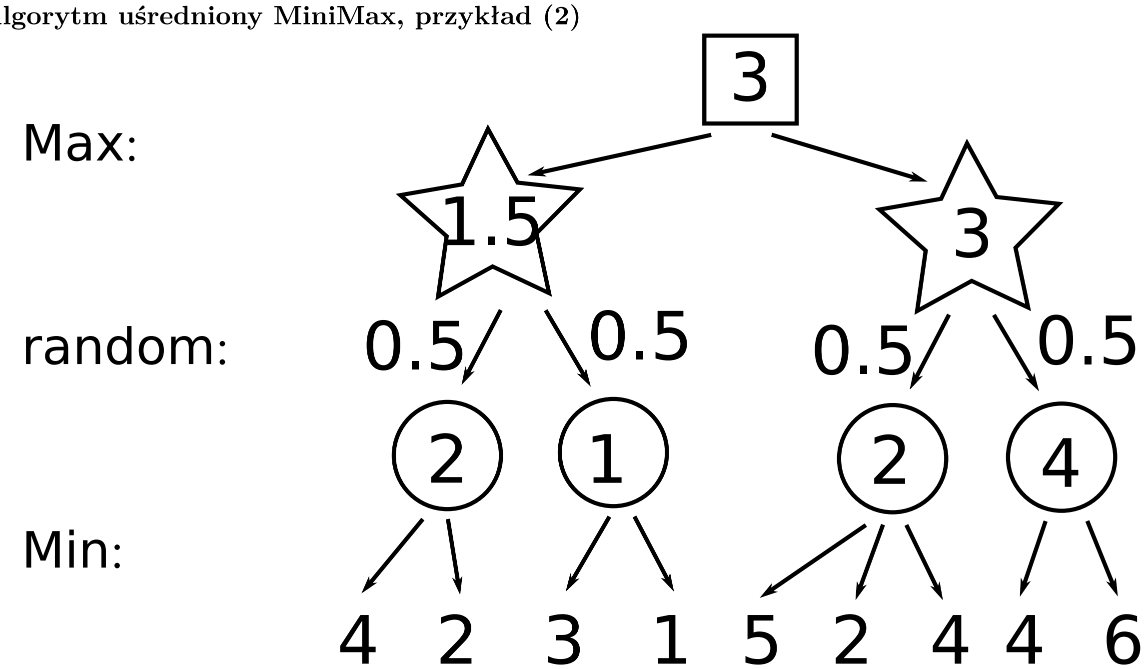
$$w(n) = \Sigma P(s) * w(s)$$



Algorytm uśredniony MiniMax, przykład (1)



Algorytm uśredniony MiniMax, przykład (2)



Gry dwuosobowe o sumie zerowej - uwagi

Gry niedeterministyczne z pełną informacją - uwagi

- węzły losowe zwiększają wysokość drzewa
- dla wysokich drzew wartość sprawdzania wprzód maleje, bo prawdopodobieństwo osiągnięcia danego węzła maleje
- odcinanie $\alpha - \beta$ mniej efektywne

- Program TDGammon (gra w backgammon) ma przeszukiwanie na głębokość 2 i ma poziom mistrza świata

Gry z niepełną informacją:

- gry z niepełną informacją – obliczanie prawdopodobieństwa każdego rozdania)

Gry w postaci strategicznej (normalnej)

Gracze podejmują decyzje jednocześnie, bez wiedzy o decyzjach innych uczestników. Opis to k macierzy wypłat, k - liczba graczy.

		Gracz 1	
		strategia A	strategia B
Gracz 2	strategia A	wypł. gracz 1, wypł. gracz 2	...
	strategia B	...	...

Przykład: polowanie na jelenia (Stag Hunt)

		Myśliwy 1	
		Jeleń	Zając
Myśliwy 2	Jeleń	2,2	0,1
	Zając	1,0	1,1

Gry w postaci strategicznej (2)

Przykład: Papier-Kamień-Nożyczki

		Gracz 1		
		Papier	Kamień	Nożyczki
Gracz 2	Papier	0, 0	1,-1	-1, 1
	Kamień	-1, 1	0, 0	1,-1
	Nożyczki	1,-1	-1, 1	0, 0

Przykład: dylemat więźnia

		Więzień 1	
		Zeznaje	Milczy
Więzień 2	Zeznaje	-5, -5	-10, 0
	Milczy	0, -10	-1, -1

Lektura: 'Sztuczna Inteligencja dla Inżynierów', wydawnictwo Politechniki Warszawskiej, 2022.

Dziękuję