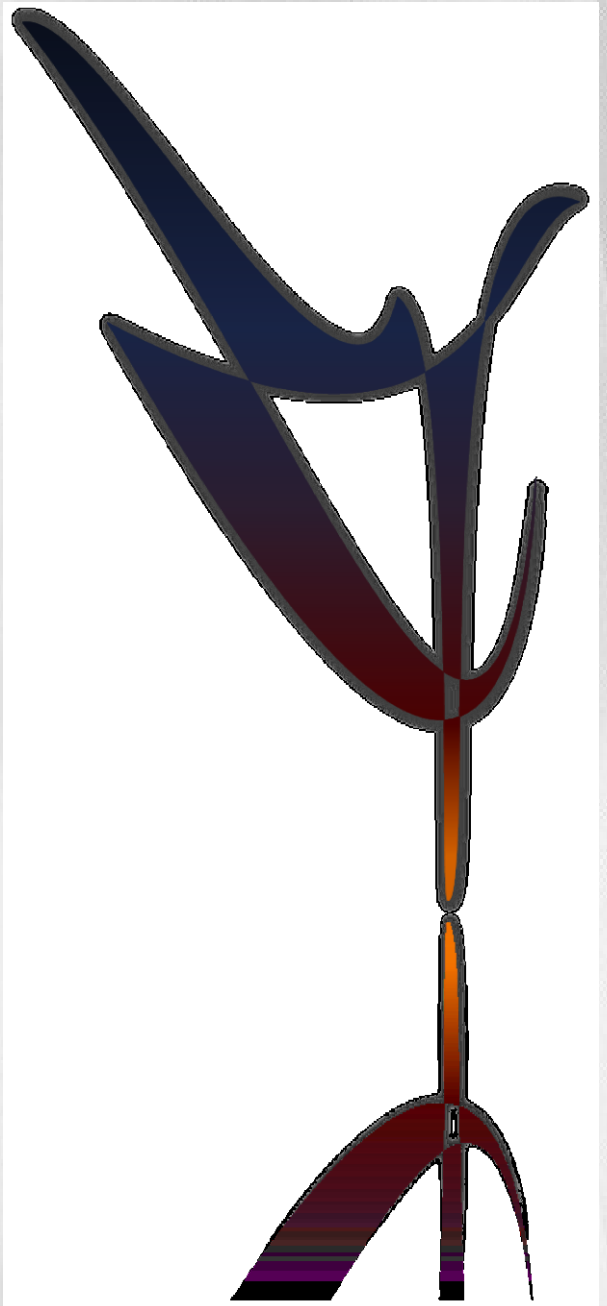


# General Game Playing

## Aktualnie stosowane algorytmy i metody konstrukcji agentów

Karol Walędzik

# General Game Playing

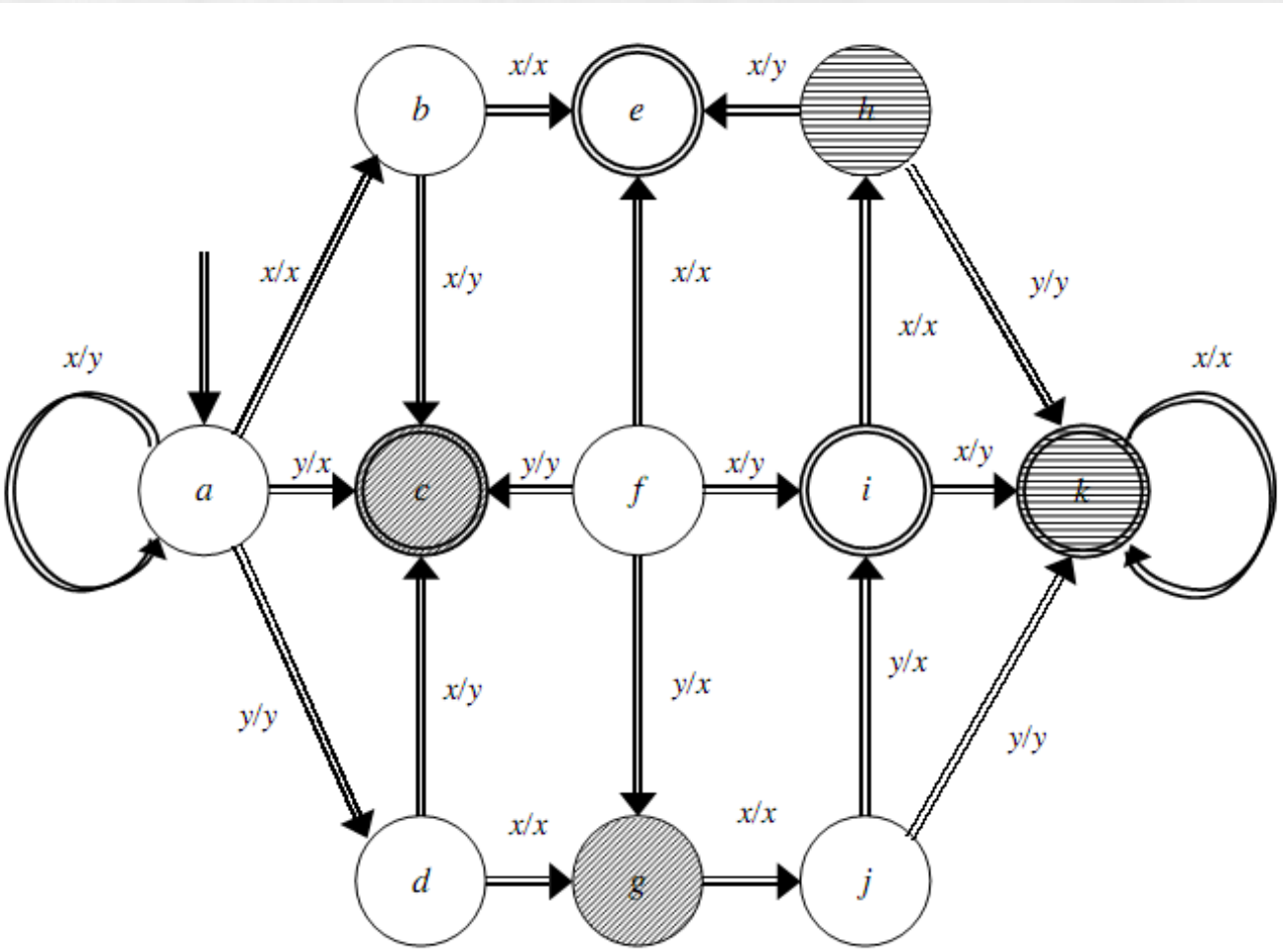
# General Game Playing?

- Cel:
  - stworzenie systemu umiejącego grać/nauczyć się grać we „wszystkie” gry
- Turniej w ramach AAAI National Conference
  - corocznie od 2005 roku
- Przebieg rozgrywki w ramach turnieju:
  - prezentacja zasad i czas na ich analizę (np. 5m)
  - rozgrywka z ograniczeniem czasowym na wykonanie pojedynczego ruchu

# Model gry

- Skończona
- Skończona liczba graczy
  - w tym 1
    - czyli *de facto* łamigłówka
- synchroniczna
  - wszyscy gracze ruszają się równocześnie (ale dopuszczalne ruchy typu *noop*)
- skończona liczba legalnych ruchów w każdym ze skończonej liczby stanów
- zmiany stanu tylko w wyniku ruchów
- ➔ maszyna stanowa

# Model gry c.d.




M. Genesereth, N. Love, and B. Pell. General Game Playing: Overview of the AAAI Competition. AI Magazine, 26(2):62-72, 2005.

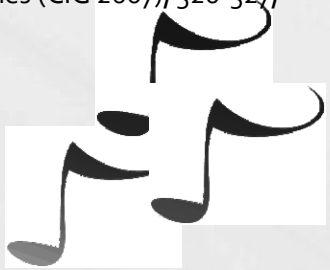
# Game Definition Language (GDL)

- Opis gry jako maszyny stanowej: zbyt rozwlekły
- Krótszy sposób opisu: Game Definition Language (GDL)
  - opis gry za pomocą fomuł logicznych
  - język bazujący na zmodyfikowanym Datalogu
    - Datalog to podzbiór Prologa
  - podstawowe relacje:
    - *role*, *true*, *init*, *next*, *legal*, *does*, *goal*, *terminal*

# GDL: Tic-Tac-Toe

```
 1. (role xplayer)
 2. (role oplayer)
 3. (init (cell 1 1 b))
 4. (init (cell 1 2 b))
 ...
 5. (init (cell 3 3 b))
 6. (init (control xplayer))
 7. (<= (next (cell ?m ?n x))
 8.     (does xplayer (mark ?m ?n))
 9.     (true (cell ?m ?n b)))
10. (<= (next (control xplayer))
11.     (true (control oplayer)))
 ...
13. (<= (row ?m ?x)
14.     (true (cell ?m 1 ?x))
15.     (true (cell ?m 2 ?x))
16.     (true (cell ?m 3 ?x)))
17. (<= (line ?x)
18.     (row ?m ?x))
 ...
19. (<= (legal ?w (mark ?x ?y))
20.     (true (cell ?x ?y b))
21.     (true (control ?w)))
22. (<= (legal xplayer noop)
23.     (true (control oplayer)))
 ...
24. (<= (goal xplayer 0)   (line o))
25. (<= (goal oplayer 100) (line o))
 ...
26. (<= terminal (line x))
```

J. Reisinger, E. Bahceci, I. Karpov, and R. Miikkulainen. Coevolving strategies for general game playing. In Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG 2007), 320-327, Honolulu, Hawaii, 2007. IEEE Press.



# GGP: Zwycięzcy turniejów

- 2005: Cluneplayer
  - minimax + funkcja ewaluacyjna
- 2006: Fluxplayer
  - minimax + funkcja ewaluacyjna
- 2007-2008: Cadiaplayer
  - UCT
- 2009-2010: Ary
  - UCT
- 2011: TurboTurtle
  - ???

# ClunePlayer

# ClunePlayer

- Zwycięzca turnieju GGP w 2005
- Pomysł:
  - stworzenie heurystycznej funkcji ewaluacyjnej jako funkcji działającej na uproszczonym modelu gry
  - uproszczony model bazujący na 3 koncepcjach:
    - wypłata (*payoff*)
    - kontrola/mobilność (*control)*
    - terminalność/czas (*termination*)

# Generowanie komponentów funkcji ewaluacyjnej

- **Krok 1** – identyfikacja wyrażeń logicznych:
  - wydobycie wyrażeń pojawiających się w opisie reguł
  - generowanie nowych wyrażeń:
    - podstawienie wszystkich możliwych stałych w miejsce zmiennych
    - identyfikacja wyróżnionych stałych wśród zidentyfikowanych

# Generowanie komponentów funkcji ewaluacyjnej

- **Krok 2** – generowanie komponentów z wyrażeń:
  - 3 interpretacje wyrażeń:
    - *liczność*
      - poprzez znalezienie liczby różnych rozwiązań wyrażenia
    - *odległość*
      - wyznaczana z wykorzystaniem grafu zależności między symbolami w opisie gry
    - *częściowe rozwiązanie*
      - stosowane dla wyrażeń złożonych, które częściowo pokrywają się z wyrażeniami celu
  - stabilność komponentów:
    - stosunek zmienności całkowitej do zmienności między kolejnymi stanami (obliczane na bazie symulacji)

# Model abstrakcyjny

Table 1: Abstract Model Parameters

| Parameter | Meaning |
|---|---|
| $P : \Omega \rightarrow [0, 100]$ | Approximates payoff function. |
| $C : \Omega \rightarrow [-1, 1]$ | Degree of control player has. |
| $T : \Omega \rightarrow [0, 1]$ | Probability that state is terminal. |
| $S_P : (1, \infty)$ | Stability of P. |
| $S_C : (1, \infty)$ | Stability of C. |

- Komponenty nie wpływające na funkcję celu są usuwane (na podstawie analizy zależności)
- Dla pozostałych wyznaczany jest kierunek korelacji z cechami modelu abstrakcyjnego (na bazie symulacji)
  - wartości cech dla stanów wyznaczane w sposób przybliżony, np. wypłata jako potencjalna wartość funkcji celu, gdyby dany stan był terminalny
- Komponenty w funkcji ewaluacyjnej ważone są stabilnością

# Przykładowe rezultaty

- Reversi
  - tylko wypłata zidentyfikowana:

*Payoff =*
*3.361 (lower right corner: # white - # black)*
*+2.015 (upper right corner: # white - # black)*
*+1.379 (upper edge: # white - # black)*
*+1.221 (lower left corner: # white - # black)*
*+1.114 (lower edge: # white - # black)*
*+0.789 (upper left corner: # white - # black)*
*+0.646 (right edge: # white - # black)*
*+0.592 (left edge: # white - # black)*
*+50.000*

# Przykładowe rezultaty

- Chess
  - tylko funkcja kontroli zidentyfikowana:

*Control =*
*0.097 (# white queens - # black queens)*
*+0.092 (# white rooks - # black rooks)*
*+0.042 (# white bishops - # black bishops)*
*+0.039 (# white knights - # black knights)*
*+0.022 (# empty squares: rank 1 - rank 8)*
*+0.004 (# white pawns - # black pawns)*

# FluxPlayer

# FluxPlayer

- Zwycięzca turnieju GGP w 2005
- Pomysł:
  - stworzenie heurystycznej funkcji ewaluacyjnej na podstawie oceny stopnia spełnienia celów gry
    - wykorzystanie elementów logiki rozmytej

# Algorytm przeszukiwania

- Algorytm przeszukiwania:
  - depth-first z niejednorodną głębokością przeszukiwania
  - korzystający z iteracyjnego pogłębiania
  - usprawnienia:
    - tablice transpozycji
    - history heuristic
    - w przypadku gier 2-sobowych: alfa-beta obcięcia
  - dla gier symultanicznych:
    - założenie paranoiczne: modelowany gracz decyduje pierwszy

# Funkcja ewaluacyjna

- Idea:
  - wyznaczenie stopnia spełnienia celu
  - unikanie stanów terminalnych przy niskich (ujemnych) wartościach funkcji celu
  - wykorzystanie logiki rozmytej
- Ale:
  - każda t-norma zwróci 0 dla koniunkcji, w której jedno z wyrażeń jest fałszywe
- Rozwiązanie:
  - stosowanie wartości p i 1-p zamiast 0 i 1
    - ale wtedy złożona koniunkcja uzyskuje zbyt małe wartości
    - rozwiązanie:
      - zastąpienie t-normy wyrażeniem z dolnym ograniczeniem


UCT

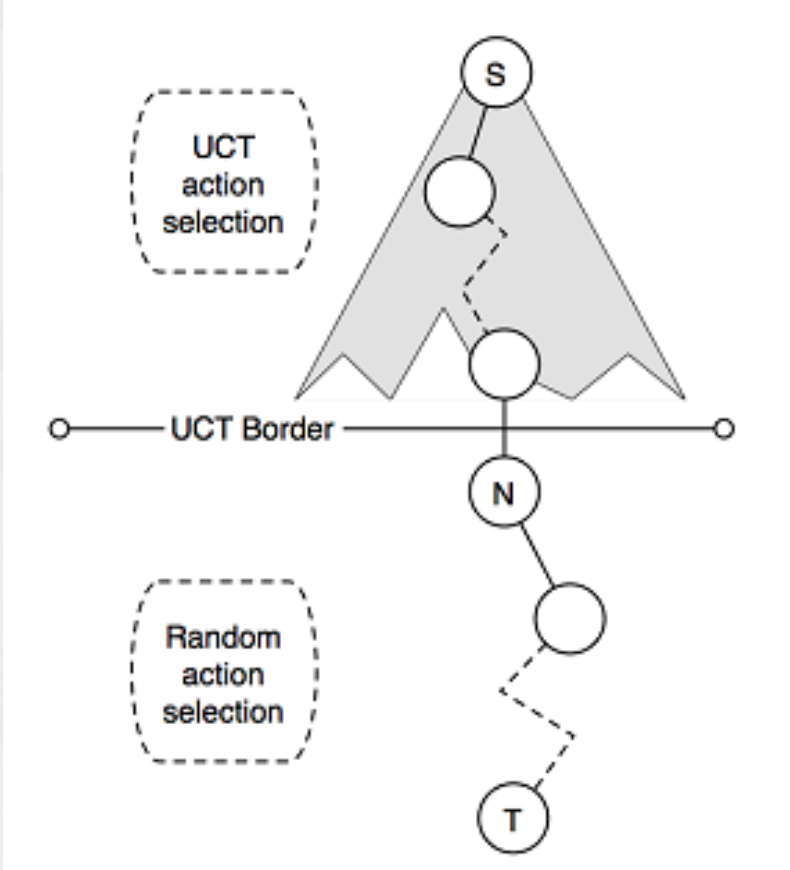
# UCT

- Sposób wybierania kolejnego ruchu podczas symulacji (UCT):

$$a^* = argmax_{a \in A(s)} \left\{ Q(s,a) + C\sqrt{\frac{\ln N(s)}{N(s,a)}} \right\}$$

  - Q(s,a) – średni dotychczasowy wynik pary stan-ruch
  - N – liczba wizyt w danym stanie/wykonań danej akcji
  - akcje nigdy nie wykonane wybierane są w pierwszej kolejności

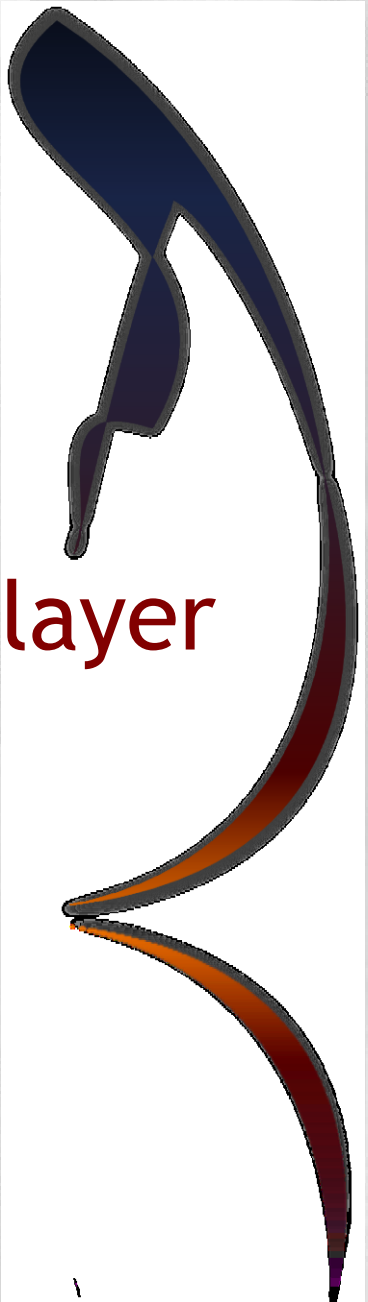
# UCT c.d.


S
UCT
action
selection
UCT Border
N
Random
action
selection
T

# UCT c.d.

- Ograniczanie zużycia pamięci:
  - usuwanie danych o węzłach powyżej aktualnego po każdym (realnie) wykonanym ruchu
    - co z pozycjami powtórzonymi w drzewie gry
      1. duplikowanie
      2. **intelegentne wykrywanie, kiedy można je usunąć z pamięci**
  - dodawanie tylko jednego nowego węzła per symulacja
- Modelowanie przeciwnika
  - każdy przeciwnik ma przydzielony swój własny model (*Cadia)*
  - przeciwnicy podejmują losowe decyzje

# CADIAPlayer



# CADIAPlayer

- Zwycięzca turnieju GGP w 2007 i 2008
- Koncepcja:
  - rozwiązanie oparte na symulacjach Monte-Carlo
    - konkretniej: UCT
      - (Upper Confidence-bounds applied to Trees)
  - logika gry reprezentowana przez automatycznie generowany podprogram w Prologu

# Algorytm przeszukiwania

- Dal gier jednoosobowych:
  - Memory Enhanced IDA*
    - (Iterative Deepening A*)
  - aplikowane już na etapie analizy reguł
  - jeśli nie znajdzie na tym etapie przynajmniej częściowego rozwiązania, na etapie gry będzie wykorzystywany znów UCT
- Dla pozostałych gier:
  - UCT

# Sterowanie symulacjami MC

- MAST – Move-Average Sampling Technique
  - analogiczna do *history heuristics*
  - dla każdego ruchu (niezależnie od kontekstu wykonania) wyznaczana średnia wyników wszystkich gier, w których został wykonany
  - w fazie MC prawdopodobieństwo wybrania ruchu proporcjonalne do tej średniej (zgodnie z rozkładem Gibbsa)

$$P(a) = \frac{e^{Q_h(a)/\tau}}{\sum_{b=1}^{n} e^{Q_h(a)/\tau}}$$

    - może być zastąpione podejściem ε-zachłannym
  - stosowany w turnieju w 2008

# Sterowanie symulacjami MC

- TO-MAST – Tree-Only MAST
  - średnie wyliczane tylko dla ruchów wykonanych w fazie UCT

# Sterowanie symulacjami MC

- NST – N-Gram Selection Technique
  - tylko dla gier 2-osobowych
  - średnie wartości wyznaczane nie tylko dla pojedynczych akcji, ale też dla sekwencji 1-3 półruchów (gracza i przeciwnika na przemian)
  - wartość ruchu to średnia z wartości dla wszystkich dostępnych sekwencji
    - sekwencje o długości większej niż 1 brane pod uwagę dopiero po przekroczeniu zadanej liczby (7) symulacji

# Sterowanie symulacjami MC

- PAST – Predicate-Average Sampling Technique
  - wartości wyznaczane dla wszystkich par
    (predykat, akcja)
    dla wszystkich predykatów zachodzących w pozycji, w której wykonywana jest akcja
  - prawdopodobieństwo wyboru ruchu proporcjonalne do maksymalnej wartości par dla wszystkich predykatów spełnionych w danym stanie

# Sterowanie symulacjami MC

- FAST – Features-To-Action Sampling Technique
  - funkcja ewaluacyjna w postaci kombinacji liniowej cech generowana za pomocą TD(λ)
    - 2 rodzaje cech (zależnie od gry):
      - liczby kamieni różnych typów
      - położenie kamieni na planszy

# Sterowanie symulacjami MC

- LGR – Last-Good-Reply Policy
  - tylko dla gier 2-osobowych
  - słownik najlepszych odpowiedzi dla sekwencji 1-2 półruchów
  - po wygranej partii wszystkie jej odpowiedzi trafiają do słownika, po przegranej – są (o ile istnieją) usuwane

# Inne modyfikacje

- RAVE – Rapid Action Value Estimate
  - podczas fazy propagacji wyniku dodatkowo wyliczana wynik dla ruchów niewybranych (rodzeństwa wybranego), które zostały wykonane niżej w drzewie gry
  - ostateczna wartość ruchu:

$$\beta(s) \times Q_{RAVE}(s,a) + (1 - \beta(s)) \times Q(s,a)$$

$$\beta(s) = \sqrt{\frac{k}{3n(s) + k}}$$

  - możliwe tworzenie podejść mieszanych: np. RAVE/MAST, RAVE/FAST

# Wnioski

- Podejście ε-zachłanne wydaje się skuteczniejsze od zastosowania rozkładu Gibbsa
- NST (N-Grams) osiąga bardzo pozytywne rezultaty w prawie wszystkich grach
- W ogólności skuteczność wszystkich metod bardzo silnie zależy od gry


Ary

# Ary

- Zwycięzca turniejów w latach 2009-2010
- Wykorzystuje UCT z tablicami transpozycji oraz Nested Monte Carlo (NMC)
  - NMC najprawdopodobniej tylko do gier jednoosobowych
  - ewentualna przewaga NMC nad UCT mocno zależna od gry

# Nested Monte Carlo

```
nested (position, level)
best playout ← {}
while not end of game do
  if level = 1 then
    move ← argmax_m(sample (play (position, m)))
  else
    move ← argmax_m(nested (play (position, m), level − 1))
  end if
  if score of playout after move > score of the best playout then
    best playout ← playout after move
  end if
  position ← play (position, move of the best playout)
end while
return score (position)
```

# Zrównoleglanie UCT bez współdzielonej pamięci

- *Root Parallelization*
  - Każdy węzeł przeprowadza oddzielnie pełną analizę i buduje własne drzewo UCT
  - Sposób podejmowania ostatecznej decyzji:
    - *Best* – ruch o najwyższej ocenie
      - z natury najlepsza dla gier jednoosobowych
    - *Sum* – na podstawie średniej ważonej liczbą symulacji
    - *Sum10* – jak *Sum*, ale uwzgladniając tylko 10 najlepszych ruchów z każdego węzła
    - *Raw* – na podstawie średniej arytmetycznej wartości ruchów
  - Wnioski:
    - Najlepsze: *Sum* i *Sum10*
    - Dla niektórych gier zyski ze zrównoleglania stosunkowo niewielkie

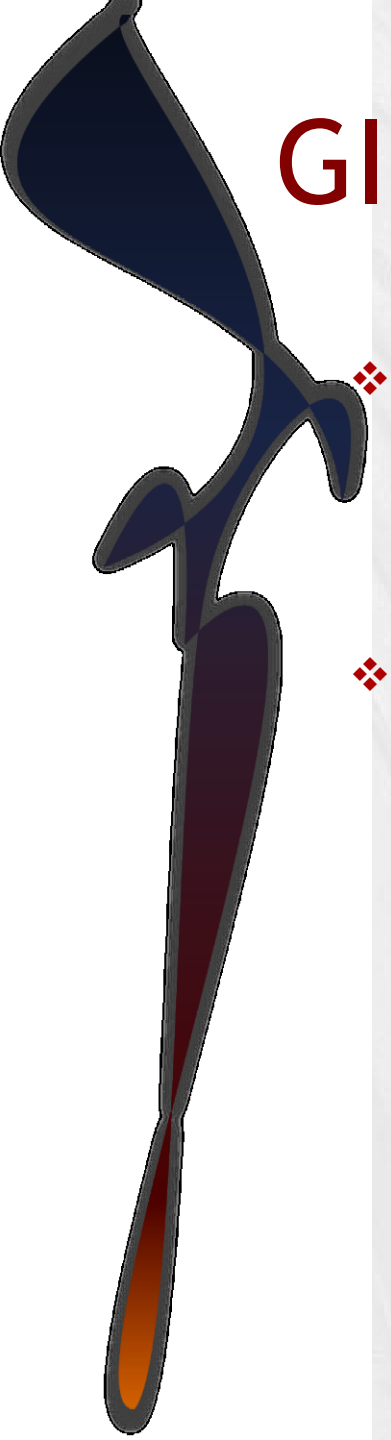
# Zrównoleglanie UCT bez współdzielonej pamięci

- *Tree Parallelization*
  - Stosowany w turniejowej wersji Ary
  - Węzeł główny utrzymuje jedno drzewo UCT
  - Symulacje MC podzlecane są innym węzłom
  - Skuteczność nadal zależna od gier, ale dla gier mało podatnych na zrównoleglanie nieco lepsza niż *Root Parallelization*

Maligne

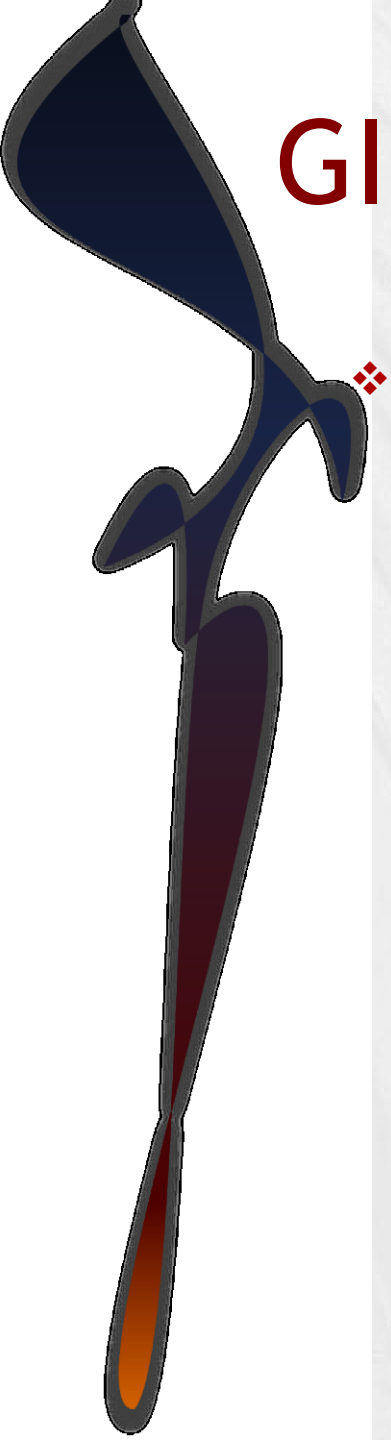
# Maligne

- Drugie miejsce w turnieju w 2010r.
- UCT + Game Independent Feature Learning (GIFL)



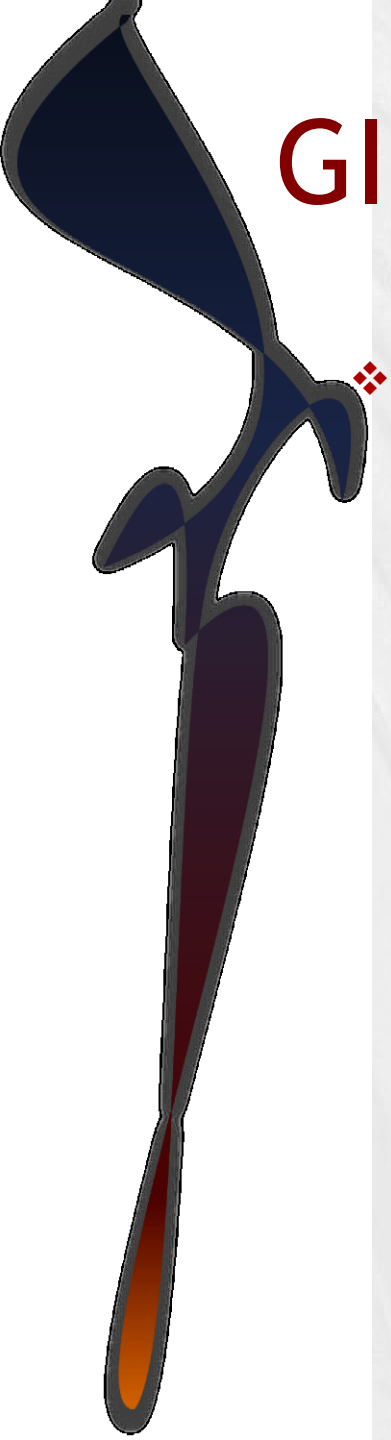
# GIFL

- Tworzy funkcję ewaluacyjną do wykorzystania w fazie MC
  - analogicznie jak CadiaPlayer
- Funkcja tworzona poprzez identyfikację cech ofensywnych i defensywnych



# GIFL

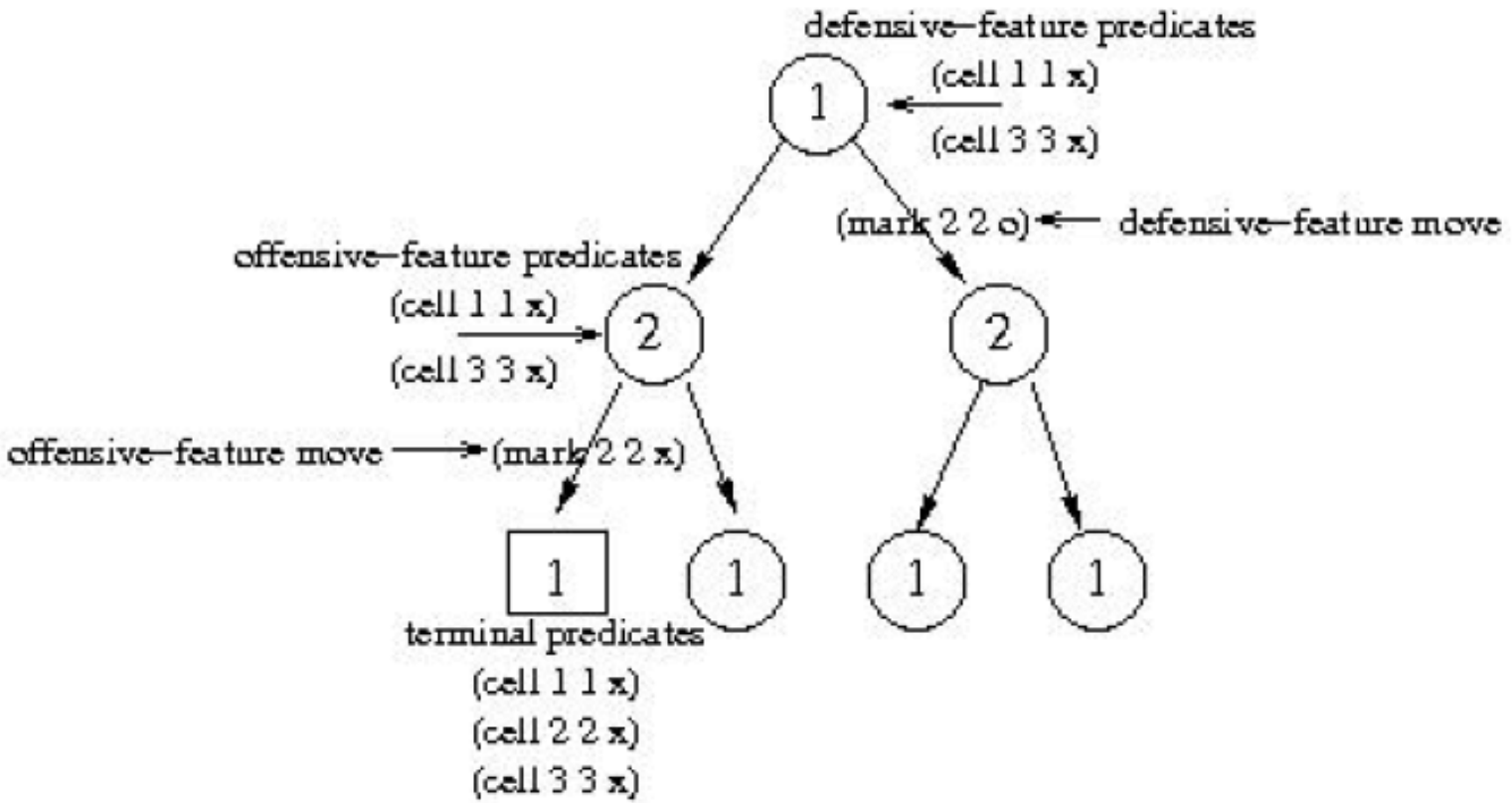
- Generowanie cech:
  - rozegraj do końca losową grę, cofnij się o 2 ruchy
  - zbuduj drzewo od wysokości 2 i z korzeniem w aktualnej pozycji
  - wygeneruj cechę ofensywną składającą się z:
    - ruchu wygrywającego grę
    - predykatów niezbędnych, by ruch był legalny oraz stan wynikowy był wygraną
      - testuj próbując usunąć każdy z predykatów, wykonując ruch i sprawdzając efekty
        - dodatkowo: jeśli wykonanie ruchu przywróciło usunięty predykat to też jest on częścią cechy



# GIFL

- Generowanie cech:
  - wygeneruj cechę defensywną, jako:
    - ruch + predykaty uniemożliwiające wykonanie ruchu ofensywnego na kolejnym poziomie
  - wygeneruj drzewo 2 poziomy wyżej
  - wygeneruj cechy ofensywne / defensywne jak wcześniej starając się zmierzać ścieżką do zidentyfikowanej wygranej
  - przypisz cechom wagę według wzoru $100*C^{level-1}$

# GIFL


defensive–feature predicates
(cell 1 1 x)
(cell 3 3 x)
1
(mark 2 2 o)
defensive–feature move
offensive–feature predicates
(cell 1 1 x)
(cell 3 3 x)
2
2
offensive–feature move
(mark 2 2 x)
1
1
1
1
terminal predicates
(cell 1 1 x)
(cell 2 2 x)
(cell 3 3 x)

# INNE ALGORYTMY I OBSZARY BADAŃ

# Kolejne modifykacje UCT

- Wykorzystanie drzew decyzyjnych jako funkcji ewaluacyjnej w fazie MC
  - drzewo zbudowane na podstawie wcześniejszych gier analizuje predykaty w aktualnym stanie i przypisuje stan gry do oczekiwanego przedziału wyników
- Uczenie za pomocą TD(0)
  - funkcja ewaluacyjna w postaci kombinacji liniowej predykatów pojawiających się w opisach stanów

# Kolejne modifykacje UCT

- UCT -> UCD (Upper Confidence bound for rooted Directed acyclic graphs)
  - uwzględnienie w algorytmie UCT powtórzeń pozycji (przejście z drzew na skierowane grafy acykliczne)
  - wynik może być propagowany nie tylko wzdłuż ścieżki, którą podążyła symulacja, ale na wszystkich rodziców stanów na tej ścieżce

# Inne podejścia do GGP

- Wykorzystanie algorytmów mrówkowych
  - każda mrówka jako gracz szukający najlepszej ścieżki w drzewie
- Wykorzystanie sieci neuronowych
- Analiza reguł i przekształcanie do innej / wydajniejszej postaci

# Najważniejsza bibliografia


- Banerjee, B. & Stone, P. General Game Learning using Knowledge Transfer, 2007, The 20th International Joint Conference on Artificial Intelligence, pp. 672-677
- Clune, J. Heuristic Evaluation Functions for General Game Playing, 2007, AAAI, pp. 1134-1139
- Finnsson, H. & Björnsson, Y. CadiaPlayer: Search-Control Techniques, 2011, KI, Vol. 25(1), pp. 9-16
- Finnsson, H. & Björnsson, Y. Simulation Control in General Game Playing Agents, 2009, Proceedings of the IJCAI-09 Workshop on General Game Playing (GIGA'09)
- Genesereth, M.R., Love, N. & Pell, B. General Game Playing: Overview of the AAAI Competition, 2005, AI Magazine, Vol. 26(2), pp. 62-72
- Kaiser, D.M. The Design and Implementation of a Successful General Game Playing Agent, 2007, FLAIRS Conference, pp. 110-115
- Kaiser, D.M. Automatic Feature Extraction for Autonomous General Game Playing Agents, 2007, Proceedings of the Sixth Intl. Joint Conf. on Autonomous Agents and Multiagent SystemsKaiser, D.M. Automatic Feature Extraction for Autonomous General Game Playing Agents [Abstract] [BibTeX]
- 2007 Proceedings of the Sixth Intl. Joint Conf. on Autonomous Agents and Multiagent Systems

# Najważniejsza bibliografia c.d.


- Kirci, M., Schaeffer, J. & Sturtevant, N. Feature Learning Using State Differences, 2009, Proceedings of the IJCAI-09 Workshop on General Game Playing (GIGA'09)
- Kuhlmann, G., Dresner, K. & Stone, P. Automatic Heuristic Construction in a Complete General Game Player, 2006, Proceedings of the Twenty-First National Conference on Artificial Intelligence, pp. 1457-62
- Levinson, R.A. General Game-Playing and Reinforcement Learning, 1996, Computational Intelligence, Vol. 12(1), pp. 155-176
- Michulke, D. Neural Networks for High-Resolution State Evaluation in General Game Playing, 2011, Proceedings of the IJCAI-11 Workshop on General Game Playing (GIGA'11)
- Michulke, D. & Schiffel, S. Distance Features for General Game Playing, 2011, Proceedings of the IJCAI-11 Workshop on General Game Playing (GIGA'11)
- Michulke, D. & Thielscher, M. Neural Networks for State Evaluation in General Game Playing, 2009, Proceedings of the European Conference on Machine Learning (EMCL), pp. 95-110

# Najważniejsza bibliografia c.d.


- Méhat, J. & Cazenave, T. A Parallel General Game Player, 2011 KI, Vol. 25(1), pp. 43-47
- Méhat, J. & Cazenave, T. Combining UCT and Nested Monte Carlo Search for Single-Player General Game Playing, 2010 IEEE Transactions on Computational Intelligence and AI in Games, Vol. 2(4), pp. 271-277
- Reisinger, J., Bahceci, E., Karpov, I. & Miikkulainen, R. Coevolving Strategies for General Game Playing, 2007, IEEE Symposium on Computational Intelligence and Games, pp. 320-327
- Saffidine, A. & Cazenave, T. A Forward Chaining Based Game Description Language Compiler, 2011, Proceedings of the IJCAI-11 Workshop on General Game Playing (GIGA'11)
- Schiffel, S. Symmetry Detection in General Game Playing, 2009, Proceedings of the IJCAI-09 Workshop on General Game Playing (GIGA'09), pp. 67-74

# Najważniejsza bibliografia c.d.


- Schiffel, S. & Thielscher, M. Automatic construction of a heuristic search function for General Game Playing, 2007, Proceedings of the Seventh IJCAI International Workshop on Nonmontonic Reasoning, Action and Change
- Schiffel, S. & Thielscher, M. Fluxplayer: A Successful General Game Player, 2007, Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI-07), pp. 1191-1196
- Sharma, S., Kobti, Z. & Goodwin, S.D. General Game Playing with Ants, 2008, Vol. 5361Proceedings of the Seventh International Conference on Simulated Evolution And Learning (SEAL'08), pp. 381-390
- Sharma, S., Kobti, Z. & Goodwin, S.D. General Game Playing: An Overview and Open Problems, 2009, Computing, Engineering and Information, International Conference on
Vol. 0, pp. 257-260
- Sheng, X. & Thuente, D. Decision Tree Learning in General Game Playing, 2011, International Conference on Artificial Intelligence and Applications
- Walędzik, K. & Mańdziuk, J. CI in General Game Playing - To Date Achievements and Perspectives, 2010, Vol. 6114Artifical Intelligence and Soft Computing, pp. 667-674