MAGICIAN: A GGP Agent

Analiza zależności w opisie reguł gier GGP

Karol Walędzik

General Game Playing

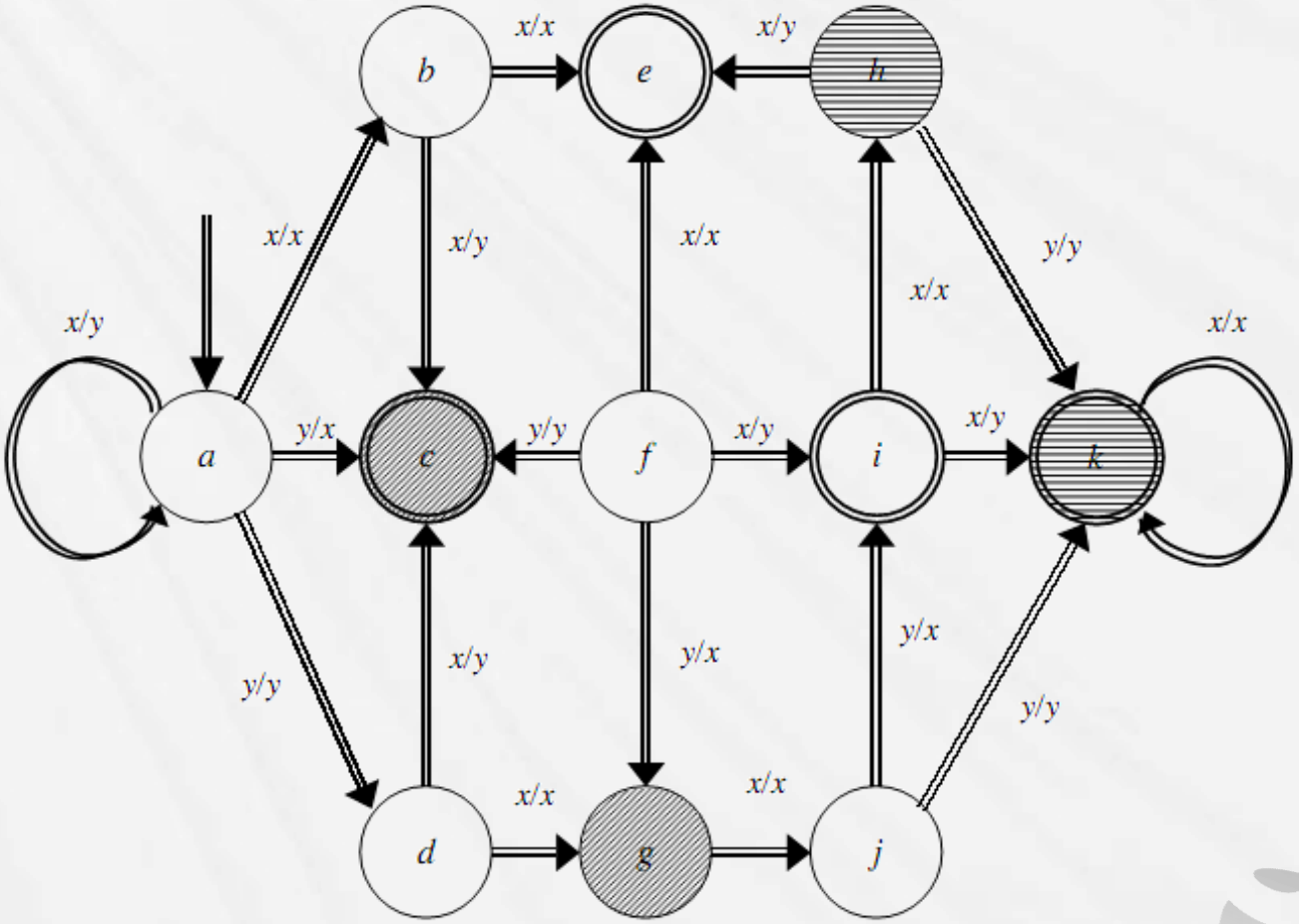
General Game Playing?

- 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.

Multi-
Approach
Game
Independent
Cunningly
Intelligent
Arch-
Nemesis

UCT
GUIDED UCT

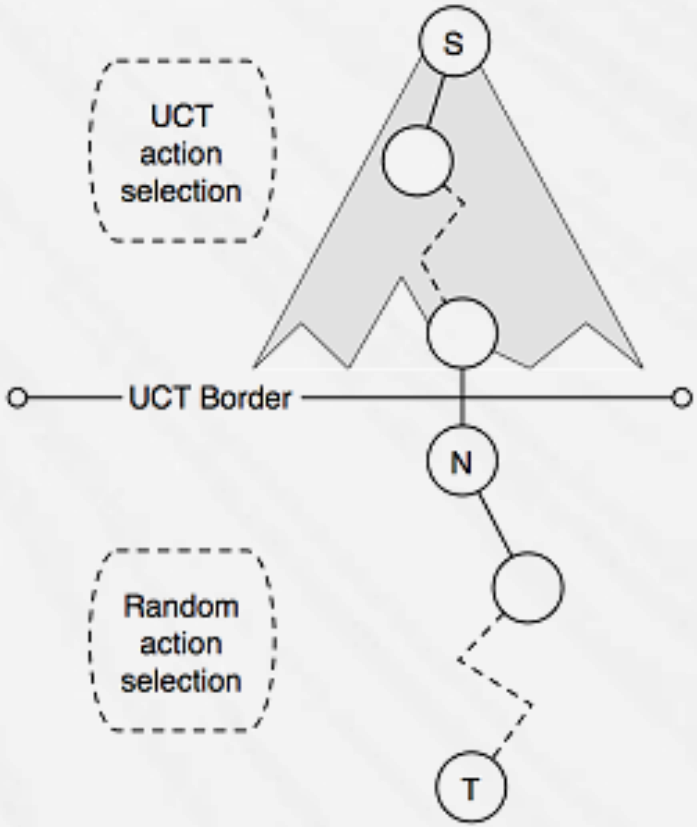
UCT

- 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

Guided UCT

- Guided UCT + funkcja ewaluacyjna
- Gdzie ta funkcja ewaluacyjna?
  1. W fazie MC:
     a) całkowite zastąpienia fazy
        - (z zadanym prawdopodobieństwem)
     b) **zakończenie symulacji przed osiągnięciem końca gry**
        - z zadanym prawdopodobieństwem w każdym węźle drzewa

Guided UCT

- Guided UCT + funkcja ewaluacyjna
- Gdzie ta funkcja ewaluacyjna?
  2. W fazie UCT:
     a) jako wstępne sortowanie ruchów nigdy nie wypróbowanych
     b) jako startowy szacunek wartości ruchów z wagą odpowiadającą *n* symulacjom
     - Problem:
       - porównywanie bardzo podobnych pozycji (różniących się tylko ostatnim ruchem)

Statistical GUCT:
Heurystyczna funkcja ewaluacyjna

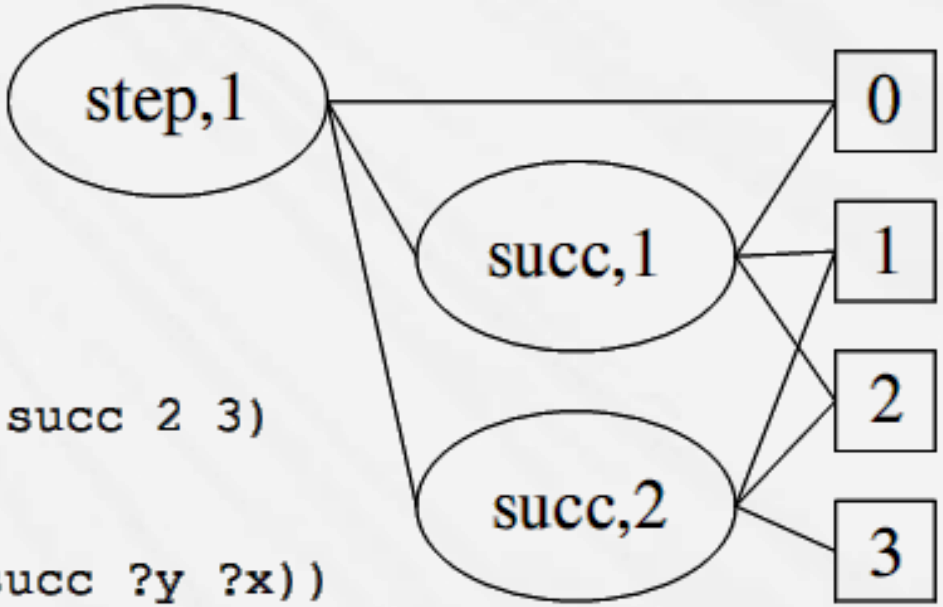
Generowanie komponentów

- Lista podstawowa:
  - wyrażenia występujące w opisie gry
    - cechy nie muszą być binarne
      - zliczanie sposobów rozwiązania zmiennych w wyrażeniu
- Uzupełnianie listy:
  - uogólniane przez podstawianie zmiennych za stałe i możliwych stałych za zmienne

```
cell(1, 1, x)        cell(1, 1, ?)
cell(?, ?, x)        cell(1, ?, x)
```

Generowanie komponentów c.d.

- „Podstawianie stałych za zmienne”
  - Wszystkich??
    - Wyznaczanie dziedzin argumentów predykatów/funkcji:




```
(succ 0 1) (succ 1 2) (succ 2 3)
(init (step 0))
(<= (next (step ?x))
    (true (step ?y)) (succ ?y ?x))
```

Złożone komponenty

- Relacje między wartościami komponentów różniących się na jednej pozycji:
  - cell(?, ?, w) – cell(?, ?, b)
  - cell(?, ?, wk) / cell(?, ?, bk)
  - a jako efekt uboczny także:
    - cell(a, 1, ?) / cell(a, 3, ?)
- Problem: co z dzieleniem przez 0?
  - Odp. 1: 0
  - Odp. 2: podzielmy przez ½ zamiast tego
  - **Odp. 3: zrezygnujmy na razie z proporcji**

Analiza statystyczna komponentów

- Pula przykładowych sekwencji
  - ok. 100
  - początek sekwencji: wynik losowej gry
  - długość: do 5 pozycji
  - odległość między pozycjami: 1-2 ruchy
  - generowanie sekwencji: losowe ruchy
  - powiązanie sekwencji ze średnim wynikiem: symulacje Monte Carlo (~10)

Analiza statystyczna komponentów c.d.

- Średnie wartości i wariancje wartości komponentów
  - dla całej gry
    - uwzględnia tylko pierwszą pozycję z każdej sekwencji
  - dla poszczególnych sekwencji
- Korelacja wartości komponentów z wynikiem gry
  - brana pod uwagę średnia wartość komponentu w sekwencji
- Stabilność wartości komponentów
  - iloraz wariancji między początkami sekwencji oraz średniej wariancji w ramach sekwencji
    - S = TV / (TV + 10*SV)

Funkcja ewaluacyjna - heurystyczna

- Wybór komponentów:
  - sortowanie:
    - min(stabilność, |korelacja z wynikiem|)
  - obcięcie do 30 pierwszych komponentów
- Wagi:
  - stabilność * korelacja z wynikiem
- Składnik stały:
  - średni wynik

Odległość faktów

Cel

- Zdefiniowanie (przybliżonych) miar „odległości” i „odległości probabilistycznej” faktów
  - d(a,b) – minimalna liczba ruchów potrzebnych na przejście ze z pewnego stanu z faktem a do stanu z faktem b
    - (minimum wyznaczane po wszystkich stanach z faktami a i b i wszystkich sekwencjach legalnych ruchów)
  - d’(a,b) – odległość ważona miarą prawdopodobieństwa, że znajdujemy się w odpowiednim stanie i kolejne przejścia będą możliwe

Uwaga: definicje koncepcyjne, nieformalne

Cel

- Intuicyjne oczekiwania:
  - d[cell(1,1,x),goal(x,100)] = d[cell(2,2,x),goal(x,100)]
  - d'[cell(1,1,x),goal(x,100)] >= d'[cell(2,2,x),goal(x,100)]
  - d[cell(1,1,x),cell(1,1,0)] = d'[cell(1,1,x),cell(1,1,0)] = ∞

Zastosowanie

- Budowa funkcji celu lub jej komponentu
  - na podstawie odległości faktów aktualnego stanu od wyrażeń *goal*
- Miara stopnia spełnienia celów końcowych / cząstkowych
- …

Algorytm budowy grafu przejść

- Wyznacz / pobierz domeny argumentów funkcji
- Wyznacz zbiór wszystkich możliwych faktów
- Zastąp reguły zawierające zmienne na wszystkie możliwe ich realizacje
  - na podstawie domen argumentów
  - ignorując predykaty zaprzeczone – nie będą wykorzystywane przez algorytm

Algorytm budowy grafu przejść

- Dla każdej reguły:
  - typu *next / goal / terminal*:
    next(cell(1,1,x)) :- true(cell(1,1,b)), does(x,put(x,1,1))
    - dodaj krawędzie pomiędzy faktami *true* i faktem *next /true/terminal* z wagami wyznaczonymi za pomocą funkcji A
  - pozostałych:
    line(x) :- true(cell(1,1,x)), true(cell(1,2,x)), true(cell(1,3,x))
    - dodaj do grafu tymczasowy węzeł z predykatem wynikowym oraz krawędzie pomiędzy faktami *true* oraz nowym węzłem (z wagą wyznaczoną za pomocą funkcji A)
    - alternatywa: rozwinięcie wszystkich wyrażeń zawierających niestandardowe predykaty

Algorytm budowy grafu przejść

- Uprość graf:
  - scal krawędzie wielokrotne
    - waga nowej krawędzi wyznaczana za pomocą funkcji B
  - usuń węzły tymczasowe
    - poszczególne ścieżki pomiędzy faktami zamieniane w pojedyncze krawędzie
      - funkcja scalania ścieżki: C
      - ewentualne nowopowstałe krawędzie wielokrotne scalane jak wyżej

Algorytm budowy grafu przejść

- Funkcje heurystyczne:
  A. funkcja wagi dla inferencji *next*
     - 1/n
     - 1/sqrt(n)
     - 1/log(n)
     - …
       - n – liczba predykatów w inferencji (wliczając wyrażenia *not*)

Algorytm budowy grafu przejść

- Funkcje heurystyczne:

  B. funkcja wagi scalonej krawędzi wielokrotnej

  - $\Sigma(x_i)$
    - ryzyko wartości większych niż 1
  - $\min(1, \Sigma(x_i/i))$
  - $\Sigma(x_i/2^i)$
    - przy krawędziach posortowanych według malejących wag
  - ...

Algorytm budowy grafu przejść

- Funkcje heurystyczne:
  C. funkcja scalania ścieżki w krawędź
  - $\Pi(x_i)$
    - ryzyko bardzo niskich wartości
  - $\Pi(\mathrm{sqrt}(x_i))$
  - $1/\Sigma(1/x_i)$
  - $\mathrm{sqrt}(1/\Sigma(1/x_i)^2)$
    - efekt: przy braku krawędzi wielokrotnych, wartość analogiczna jak przy rozwinięciu wyrażeń
  - ...

Algorytm budowy grafu przejść

- Funkcje heurystyczne:

  D. funkcja wyznaczania probabilistycznej długości ścieżki

  - 1/C
  - $\Sigma(1/x_i)$
  - $\Sigma(1/\mathrm{sqrt}(x_i))$
  - …

Wyznaczanie odległości faktów

- Wariant podstawowy:
  - długość najkrótszej ścieżki między faktami
- Wariant rozszerzony:
  - uwzględnienie obecności wielu ścieżek, scalając je za pomocą funkcji C
    - istotnie problemy algorytmiczne

Odległość stanu od faktu

- Dla stanu A i faktu x:
  - $avg(d'(a_i,x))$
  - $max(d'(a_i,x))$
  - $\Sigma(d'(a_i,x)/2^{i-1})$
    - a_i – wszystkie fakty stanu A, w kolejności malejącej odległości od x

Zastosowania podstawowe

- Oszacowanie wartości faktu (np. na potrzeby budowy funkcji ewaluacyjnej):
  - średnia osiągalnych wyników ważona ich odległościami od faktu
- Funkcja oceny (lub jej komponent)
  - średnia oszacowań wszystkich faktów opisujących stan

Zastosowania zaawansowane

- Identyfikacja celów cząstkowych
  - Faktów / zestawów faktów bliskich wysokim wynikom
- Wyznaczanie odległości od celów cząstkowych
- Identyfikacja fazy gry (ocena bliskości końca rozgrywki)

Pozostałe pomysły

Pozostałe pomysły

- Dodatkowy komponent aplikacji - *wyrocznia*:
  - Zaimplementowana jako oddzielny proces, oddzielna maszyna lub mini cluster
  - Generuje sugestie:
    - zmiany parametrów nauki i gry
    - zmiany algorytmu gry
  - Podstawowe kryterium:
    - zmiany szacunkowej wartości pozycji
    - sprawdzalność przewidywań rozwoju gry (ruchów pozostałych graczy)

Bibliografia

- J. Clune: Heuristic evaluation functions for General Game Playing. In Proceed- ings of the Twenty-Second AAAI Conference on Articial Intelligence (AAAI- 07), pages 1134–1139, Vancouver, BC, Canada, 2007. AAAI Press.
- H. Finnsson, Y. Bjornsson: Simulation-based approach to General Game Playing. In Proceedings of the Twenty-Third AAAI Conference on Articial Intelligence (AAAI-08), pages 259–264, Chicago, IL, 2008. AAAI Press.
- General Game Playing website by Stanford University. http://games.stanford.edu/.
- General Game Playing website by Dresden University of Technology. http://www.general-game-playing.de/.
- M. Genesereth, N. Love: General Game Playing: Overview of the AAAI Competition. http://games.stanford.edu/competition/misc/aaai.pdf, 2005.
- D. Kaiser: The Design and Implementation of a Successful General Game Playing Agent. In Proceedings of FLAIRS Conference, pages 110–115, 2007
- G. Kuhlmann, K. Dresner, and P. Stone: Automatic heuristic construction in a complete General Game Player. In Proceedings of the Twenty-First AAAI Conference on Artificial Intelligence (AAAI-06), pages 1457–1462, Boston, MA, 2006. AAAI Press.
- N. Love, T. Hinrichs, D. Haley, E. Schkufza, M. Genesereth: General Game Playing: Game Description Language Specification. http://games.stanford.edu/language/spec/gdl spec 2008 03.pdf, 2008.
- J. Mańdziuk, M. Kusiak, K. Walędzik: Evolutionary-based heuristic generators for checkers and give-away checkers. Expert Systems, 24(4): 189–211, Blackwell-Publishing, 2007.
- J. Reisinger, E. Bah ̧ceci, I. Karpov and R. Miikkulainen: Coevolving strategies for general game playing. In Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG'07), pages 320–327, Honolulu, Hawaii, 2007. IEEE Press.
- S. Schiffel, M.Thielscher: Automatic Construction of a Heuristic Search Func tion for General Game Playing. In Seventh IJCAI International Workshop on Nonmonotonic Reasoning, Action and Change (NRAC07).