UCT

Wybrane modyfikacje i usprawnienia

Karol Walędzik

UCT

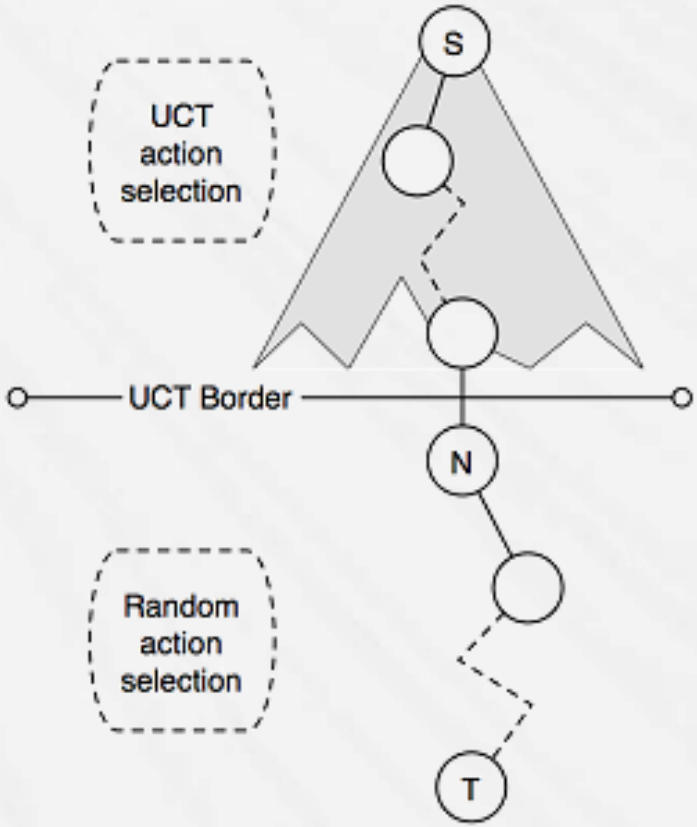
UCT

- Sposób wybierania kolejnego ruchu podczas symulacji (UCT):

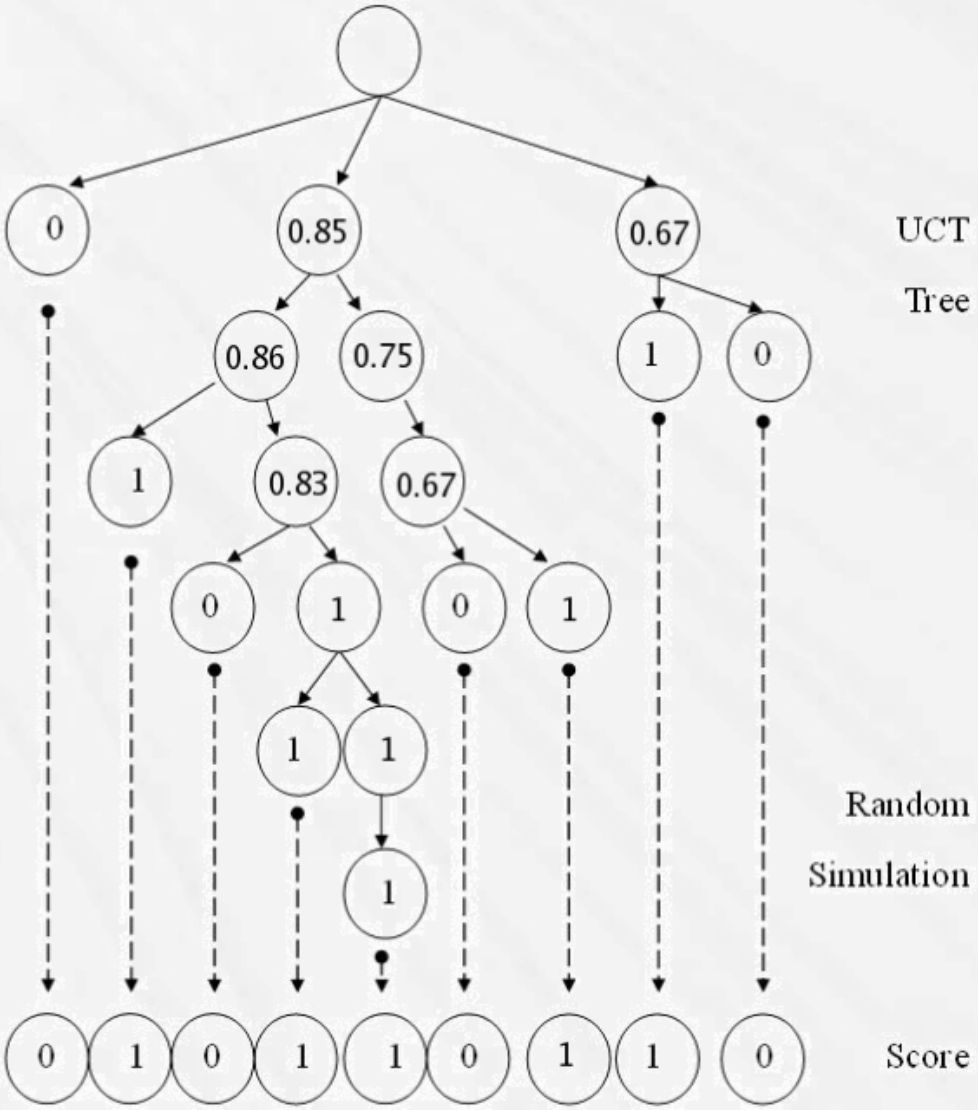
$$a^* = argmax_{a \in A(s)} \left\{ Q(s,a) + C\sqrt{\frac{\ln N(s)}{N(s,a)}} \right\}$$

  - 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.


0
0.85
0.67
UCT
Tree
0.86
0.75
1
0
1
0.83
0.67
0
1
0
1
1
1
1
Random
Simulation
0
1
0
1
1
0
1
1
0
Score

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. inteligentne wykrywanie, kiedy można je usunąć z pamięci
         1. różne sposoby propagowanie wyniku w górę
  - dodawanie tylko jednego nowego węzła per symulacja
- Modelowanie przeciwnika
  - każdy przeciwnik ma przydzielony swój własny model
  - przeciwnicy podejmują losowe decyzje

UCT – wnioski

- Przy modelowaniu wszystkich graczy:
  - asymptotycznie zbieżny do wyników algorytmu minimax
    - czyli: optymalnej gry obu graczy
- Problemy przy niewielkiej liczbie symulacji:
    - jak częsty to przypadek?
  - bardzo duży wpływ losowości
  - niestacjonarność rezultatów dla pary stan-ruch
    - wynik rozbudowy drzewa UCT w pamięci

UCT – punkty modyfikacji

- Sposób rozbudowy drzewa UCT
- Strategia wyboru ruchów w fazie UCT
  - w szczególności: kolejność ruchów niezbadanych
- Sposób wyznaczania wartości Q(s,a)
  - w szczególności: sposób propagacji wyników symulacji w górę
- Strategia wyboru ruchów w fazie Monte-Carlo
  - w szczególności: opcja całkowitego zastąpienia fazy Monte-Carlo innym rozsądnie szybkim podejściem
- Kryterium ostatecznego wyboru ruchu

Go

MoGo

- Strategia wyboru ruchu w fazie MC:
  - ostatni ruch to Atari => wykonaj losowy ruch broniący, jeśli istnieje
  - poszukaj ruchu na 8 pozycjach wokół ostatniego
    - brane pod uwagę ruchy tworzące znany wzorzec 3x3
  - poszukaj ruchu zamykającego kamienie gdziekolwiek na planszy
    - pierwsza propozycja globalna
  - wykonaj losowy ruch

MoGo

- Modyfikacje funkcji preferencji ruchów

First Play Urgency

Starting with playing all arms is not optimal; Let c a default constant. Let X_i' such that

- $X_i' = \hat{X}_i + \sqrt{2\frac{\log T}{T_i}}$ if $T_i > 0$;
- $X_i' = c$ if $T_i = 0$;

Choose the highest X_i'.
Empirically $c \approx 1 \rightarrow +50$ ELO.

MoGo

- Modyfikacje funkcji preferencji ruchów

Tested formulas

With previous notation and $\hat{\sigma}_i$ the empirical standard deviation of rewards for arm i, we experimented:

- $\hat{X}_i + p \sqrt{\frac{\log T}{T_i}}$
- $\hat{X}_i + max(p\hat{\sigma}_i \sqrt{\frac{\log T}{T_i}}, \epsilon)$
- $\hat{X}_i + p \sqrt{\frac{\log T}{T_i} \min\{1/4, V_i\}}$, with $V_i = \hat{\sigma}_i{}^2 + q \sqrt{\frac{\log T}{T_i}}$
- $\hat{X}_i + p\hat{\sigma}_i \sqrt{\frac{\log T}{T_i}} + q\hat{\sigma}_i \frac{\log T}{T_i}$

MoGo

- Modyfikacje funkcji preferencji ruchów

Share information between arms

There is no independence between arms vertically and horizontally. The goal is to improve the performance when T is small.

- Average results from neihbor moves (add a term $\frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \hat{X}_j$);
- use results from ancestors: set a c_i for each move according to $\hat{X}_i$ of its grandfather.

MoGo: RAVE

- 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:

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

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

MoGo: RLGO

- MoGo + liniowa funkcja ewaluacyjna
  - cechy: wzorce w Go
  - nauka:
    - TD(0)
    - nauka przez grę z kopią z polityką ε-zachłanną

MoGo: RLGO c.d.

- MoGo + liniowa funkcja ewaluacyjna
  - strategia ε-zachłanna

$$\pi_\epsilon(s,a) = \begin{cases} 1-\epsilon+\frac{\epsilon}{|A(s)|} & \text{if } a = \text{argmax}_{a'} Q_{RLGO}(s,a') \\ \frac{\epsilon}{|A(s)|} & \text{otherwise} \end{cases}$$

  - strategia zachłanna z gaussowsko zaszumioną funkcją ewaluacyjną

$$\pi_\sigma(s,a) = \begin{cases} 1 & \text{if } a = \text{argmax}_{a'} Q_{RLGO}(s,a') + \eta(s,a') \\ 0 & \text{otherwise} \end{cases}$$

  - strategia softmax

$$\pi_\tau(s,a) = \frac{e^{Q_{RLGO}(s,a)/\tau}}{\sum_{a'} e^{Q_{RLGO}(s,a')/\tau}}$$

Strategie budowania drzewa UCT

- *All ends*
  - 1 nowy węzeł per symulacja
- *Visit count*
  - wymóg: co najmniej tyle symulacji ile rodzeństwa
- *Sibling2*
  - j.w., ale z dwukrotnością liczby rodzeństwa

Strategie budowania drzewa UCT c.d.

- *Transition probability:*
  - warunek *visit count* lub:

$$P(m_i) = \frac{eval(m_i)}{\sum_{m \in M} eval(m)}.$$

$$P(m_i) \geq Th.$$

Strategie budowania drzewa UCT c.d.

- *Salient winning rate:*
  - warunek *visit count* lub*:*

$$X_{Ri} = X_i + C(\frac{\sigma_i}{\sqrt{V_i}})$$

$$X_{Li} = X_i - C(\frac{\sigma_i}{\sqrt{V_i}}).$$

$$X_{Li} - X_{Rsecond} \geq Th.$$

Strategie budowania drzewa UCT c.d.

- *Visit count estimate:*
  - warunek *visit count* lub:

$$E_i = \begin{cases} \frac{8 \ln E_p}{(X^* - X_i)^2} + 1 + \frac{\pi^2}{3} \; if \, X_i \neq X^*, \\ \frac{E_p - (\sum_{X_j \neq X^*} E_j)}{|\{X_k : X_k = X^*\}|} \; if \, X_i = X^* \end{cases}$$

$$E_i \geq |M|.$$

  - estymacja dla korzenia: oszacowanie liczby symulacji w zadanym czasie

General Game Playing

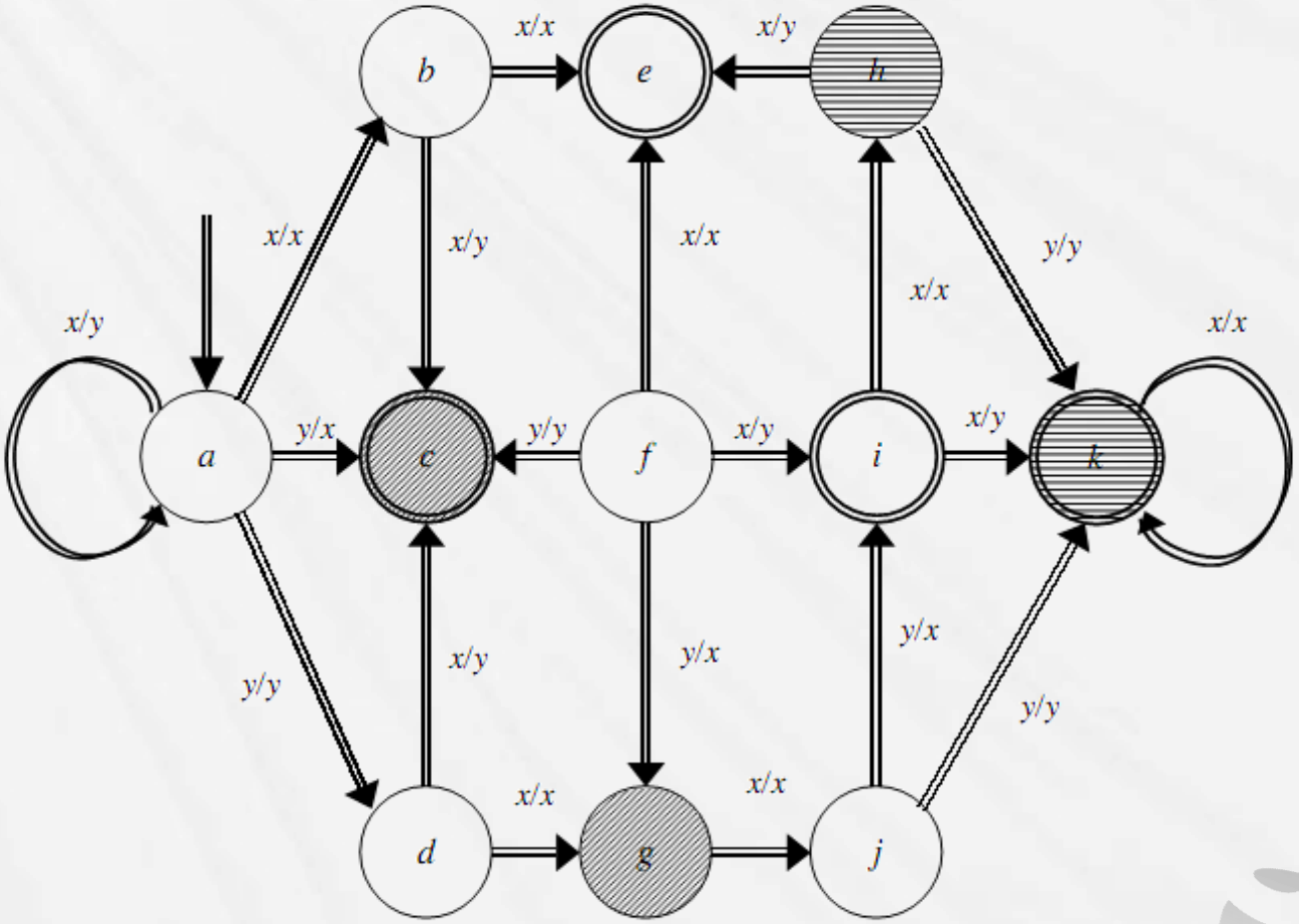
General Game Playing?

- 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.

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 – wg ostatnich testów zwykle skuteczniejszym

TO-MAST

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

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

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

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

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
  - wydaje się najskuteczniejszą techniką

Sufficiency Threshold

- Modyfikacja dla gier deterministycznych z binarnym wynikiem (wygrana/przegrana)
  - potwierdzenie jakiegokolwiek wygrywającego ruchu ważniejsze niż weryfikowanie drobnych różnic między dwoma ruchami

$$\hat{C} = \begin{cases} C & \text{when all } Q(s,a) \leq \alpha, \\ 0 & \text{when any } Q(s,a) > \alpha. \end{cases}$$

Moving Average Return Function

- częściowe rozwiązanie problemu niestabilności wyników symulacji

$$Q(s,a) = Q_{old}(s,a) + \lambda(r - Q_{old}(s,a))$$

$$\lambda = \begin{cases} 1/N(s,a) & \text{when } N(s,a) \leq M, \\ \lambda_0 & \text{when } N(s,a) > M. \end{cases}$$

Early Cutoffs

Algorithm 1 Pseudo-code for deciding cuts for the Early Cutoff extension

```
if not useEarlyCutoff then
    return false
end if
if playoutSteps < minimumSteps then
    return false
end if
if IsGoalStable() then
    // Cutoff point has been calculated as:
    // cut ← firstGoalChange + numPlayers
    return playoutSteps ≥ cut
end if
if hasTerminalInterval() then
    // Cutoff point has been calculated as:
    // cut ← firstTerminal + 0.33 ∗ terminalInterval
    return playoutSteps ≥ cut
end if
```

Unexplored Action Urgency

Algorithm 2 Pseudo-code for action selection when using the Unexplored Action Urgency extension

```
if state.explored = ∅ then
    // We are in the Playout phase
    return playoutStrategy(state.unexplored)
end if
action ← selectionStrategy(state.explored)
if state.unexplored = ∅ then
    // Fringe not reached in the Selection phase
    return action
end if
// Fringe reached in the Selection phase
exploit ← action.uctValue
discount ← state.unexplored.size() / state.actions.size()
urgency ← 50 + Cp* √ln state.visits() * discount
if exploit ≥ urgency then
    return action
else
    return playoutStrategy(state.unexplored)
end if
```

Najważniejsza bibliografia

- Sylvain Gelly and David Silver. 2007. Combining online and offline knowledge in UCT. In *Proceedings of the 24th international conference on Machine learning* (ICML '07), Zoubin Ghahramani (Ed.). ACM, New York, NY, USA, 273-280.
- Sylvain Gelly, Yizao Wang, Remi Munos, Olivier Teytaud. 2006. MoGo: Improvements in Monte-Carlo Computer-Go using UCT and sequence-like simulations; https://www.lri.fr/~sebag/COURS/SG_Mogo.pdf
- Takayuki Yajima, Tsuyoshi Hashimoto, Toshiki Matsui, Junichi Hashimoto, and Kristian Spoerer. 2010. Node-expansion operators for the UCT algorithm. In *Proceedings of the 7th international conference on Computers and games* (CG'10), H. Jaap van den Herik, Aske Plaat, and Hiroyuki Iida (Eds.). Springer-Verlag, Berlin, Heidelberg, 116-123.
- Stefan F Gudmundsson, Yngvi Björnsson. 2011. MCTS: Improved Action Selection Techniques for Deterministic Games. In proceeding of: IJCAI'11 Workshop on General Intelligence in Game Playing Agents (GIGA'11)
- Hilmar Finnsson. 2012. Generalized Monte-Carlo Tree Search Extensions for General Game Playing. Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada.
- 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)