Experimental Studies in General Game Playing: An Experience Report

Jakub Kowalski, Marek Szykuła

University of Wrocław, Poland

RAI@AAAI

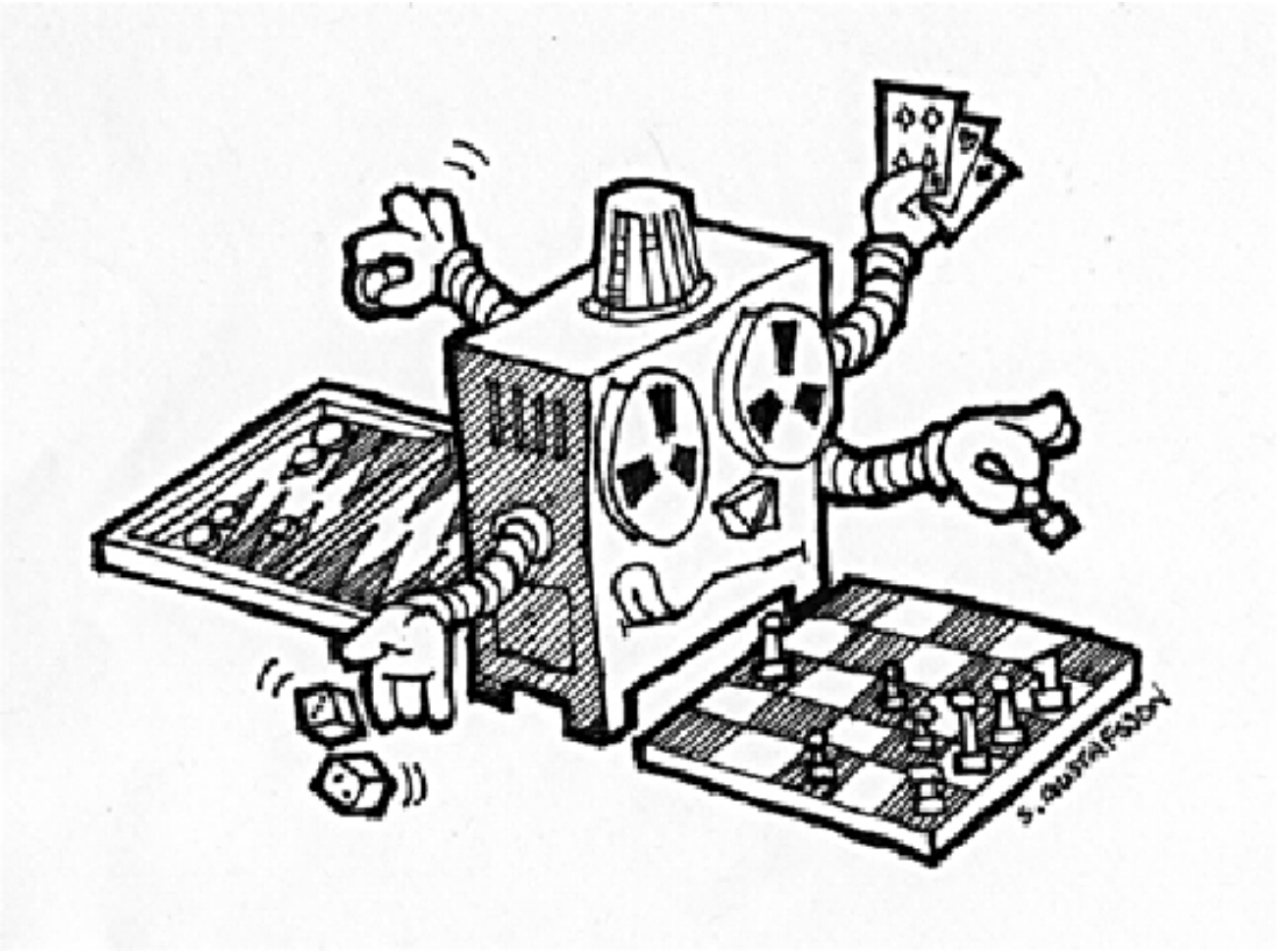
07.02.2020

Why this paper?

- It has been nearly fifteen years since announcing General Game Playing challenge.
- We think our survey may provide an interesting perspective of how chaotic methods were allowed when nothing better was possible.
- The goal of this note is to point out common difficulties and problems in the experimental research in the area.
- We hope that our recommendations will help in avoiding them in future works and allow more *fair* and *reproducible* comparisons.

General Game Playing

General Game Playing (GGP)



General Game Description Languages

- 1968 Chess-like games *(Pitrat)*
- 1992 METAGAME *(Pell)*
- 2005 **Stanford's GDL** *(Genesereth, Love, Pell)*
- 2008 Ludi *(Browne)*
- 2010 GDL-II *(Thielscher)*
- 2010 Toss *(Kaiser, Stafiniak)*
- 2011 Strategy Game Description Language *(Mahlmann et al.)*
- 2012 Simplified Boardgames *(Björnsson)*
- 2013 Card Game Description Language *(Font et al.)*
- 2013 **General Video Game AI** *(Ebner et al.)*
- 2015 rtGDL *(Kowalski, Kisielewicz)*
- 2019 **Regular Boardgames (RBG)** *(Kowalski et al.)*
- 2019 **Ludii** *(Browne et al.)*

Stanford's GDL (Genesereth, Love, Pell; 2005)

- Describes any turn-based, finite, and deterministic *n*-player game with perfect information.
- Datalog-based, high-level, strictly declarative language.
- International General Game Playing Competition (2005-2016).
- State of the game is a set of true facts.
- No predefined concepts, only a few keywords.
- Keywords define different game elements and the game dynamics.

```
(role player)
(light p) (light q)
(<= (legal player (turnOn ?x)) (not (true (on ?x))) (light ?x))
(<= (next (on ?x)) (does player (turnOn ?x)))
(<= (next (on ?x)) (true (on ?x)))
(<= terminal (true (on p)) (true (on q)))
(<= (goal player 100) (true (on p)) (true (on q)))
```

Comparing Agents Efficiency in Stanford's GGP

Developing agents for GDL

Context and Limitations

- Http-based communication protocol.
- Technological freedom: any software, any hardware.
- Agents as servers.
- Domain competitiveness.
- Necessity for reproducible research not as clearly defined as today.
- Slightly more complicated management of open-source projects.

Overview of the Research

- We comment on 9 publications from years 2009-2017.
- Focused on improving efficiency of reasoning.

Findings

- Almost complete lack of direct comparisons.
  - systems not available as open-source, the ones available still not used
  - GGP agents are very complex systems,
  - usually not designed to be operated by anyone except the authors
- Common metric exists: number of Flat MC simulations.
  - hardware specified with various level of details
- Sets of testgames used in experiments are not standardized.
  - usually minimal, especially in early works
  - two common game repositories, but sometimes ambiguities occur
  - large variety in choices of test games
- With time, the quality of the presented experiments improves.

Comparison to General Video Game AI competition

- Agents stick to one interface, Java/Python only.
- Run on competition-side, unified hardware.
- Official, unambiguous game repository.

Comparing Different GGP Formalisms

Comparing GDL's

As we have multiple GGP languages, the questions arise

- Which language is better – more universal, more efficient, more readable, easier for learning or content generation?
- Do agents in one system have some natural advantage over the agents from the other system?
- Can we automatically translate rules between these languages?
- etc.

How these languages actually look like?

Comparing GDL's

As we have multiple GGP languages, the questions arise

- Which language is better – more universal, more efficient, more readable, easier for learning or content generation?
- Do agents in one system have some natural advantage over the agents from the other system?
- Can we automatically translate rules between these languages?
- etc.

How these languages actually look like?

Breakthrough in GDL

```
(role white)
(role black)

(init (cellHolds 1 1 white))
(init (cellHolds 2 1 white))
(init (cellHolds 3 1 white))
(init (cellHolds 4 1 white))
(init (cellHolds 5 1 white))
(init (cellHolds 6 1 white))
(init (cellHolds 7 1 white))
(init (cellHolds 8 1 white))
(init (cellHolds 1 2 white))
(init (cellHolds 2 2 white))
(init (cellHolds 3 2 white))
(init (cellHolds 4 2 white))
(init (cellHolds 5 2 white))
(init (cellHolds 6 2 white))
(init (cellHolds 7 2 white))
(init (cellHolds 8 2 white))

(init (cellHolds 1 7 black))
(init (cellHolds 2 7 black))
(init (cellHolds 3 7 black))
(init (cellHolds 4 7 black))
(init (cellHolds 5 7 black))
(init (cellHolds 6 7 black))
(init (cellHolds 7 7 black))
(init (cellHolds 8 7 black))
(init (cellHolds 1 8 black))
(init (cellHolds 2 8 black))
(init (cellHolds 3 8 black))
(init (cellHolds 4 8 black))
(init (cellHolds 5 8 black))
(init (cellHolds 6 8 black))
(init (cellHolds 7 8 black))
(init (cellHolds 8 8 black))

(init (control white))

(<= (legal white (move ?x ?y1 ?x ?y2))
    (true (control white))
    (true (cellHolds ?x ?y1 white))
    (succ ?y1 ?y2)
    (cellEmpty ?x ?y2))
(<= (legal white (move ?x1 ?y1 ?x2 ?y2))
    (true (control white))
    (true (cellHolds ?x1 ?y1 white))
    (succ ?y1 ?y2)
    (succ ?x1 ?x2)
    (not (true (cellHolds ?x2 ?y2 white))))
(<= (legal white (move ?x1 ?y1 ?x2 ?y2))
    (true (control white))
    (true (cellHolds ?x1 ?y1 white))
    (succ ?y1 ?y2)
    (succ ?x2 ?x1)
    (not (true (cellHolds ?x2 ?y2 white))))
(<= (legal black (move ?x ?y1 ?x ?y2))
    (true (control black))
    (true (cellHolds ?x ?y1 black))
    (succ ?y2 ?y1)
    (cellEmpty ?x ?y2))
(<= (legal black (move ?x1 ?y1 ?x2 ?y2))
    (true (control black))
    (true (cellHolds ?x1 ?y1 black))
    (succ ?y2 ?y1)
    (succ ?x1 ?x2)
    (not (true (cellHolds ?x2 ?y2 black))))
(<= (legal black (move ?x1 ?y1 ?x2 ?y2))
    (true (control black))
    (true (cellHolds ?x1 ?y1 black))
    (succ ?y2 ?y1)
    (succ ?x2 ?x1)
    (not (true (cellHolds ?x2 ?y2 black))))
(<= (legal white noop)
    (true (control black)))
(<= (legal black noop)
    (true (control white)))
(<= (next (cellHolds ?x2 ?y2 ?player))
    (role ?player)
    (does ?player (move ?x1 ?y1 ?x2 ?y2)))
(<= (next (cellHolds ?x3 ?y3 ?state))
    (true (cellHolds ?x3 ?y3 ?state))
    (role ?player)
    (does ?player (move ?x1 ?y1 ?x2 ?y2))
    (distinctCell ?x1 ?y1 ?x3 ?y3)
    (distinctCell ?x2 ?y2 ?x3 ?y3))
(<= (next (control white))
    (true (control black)))
(<= (next (control black))
    (true (control white)))

(<= terminal
    whiteWin)
(<= terminal
    blackWin)

(<= (goal white 100)
    whiteWin)
(<= (goal white 0)
    (not whiteWin))
(<= (goal black 100)
    blackWin)
(<= (goal black 0)
    (not blackWin))
(<= (cell ?x ?y)
    (index ?x)
    (index ?y))
(<= (cellEmpty ?x ?y)
    (cell ?x ?y)
    (not (true (cellHolds ?x ?y white)))
    (not (true (cellHolds ?x ?y black))))
(<= (distinctCell ?x1 ?y1 ?x2 ?y2)
    (cell ?x1 ?y1)
    (cell ?x2 ?y2)
    (distinct ?x1 ?x2))
(<= (distinctCell ?x1 ?y1 ?x2 ?y2)
    (cell ?x1 ?y1)
    (cell ?x2 ?y2)
    (distinct ?y1 ?y2))
(<= whiteWin
    (index ?x)
    (true (cellHolds ?x 8 white)))
(<= blackWin
    (index ?x)
    (true (cellHolds ?x 1 black)))
(<= whiteWin
    (not blackCell))
(<= blackWin
    (not whiteCell))
(<= whiteCell
    (cell ?x ?y)
    (true (cellHolds ?x ?y white)))
(<= blackCell
    (cell ?x ?y)
    (true (cellHolds ?x ?y black)))
(index 1)
(index 2)
(index 3)
(index 4)
(index 5)
(index 6)
(index 7)
(index 8)
(succ 1 2)
(succ 2 3)
(succ 3 4)
(succ 4 5)
(succ 5 6)
(succ 6 7)
(succ 7 8)
```

Breakthrough in Regular Bowardgames

```
#players = white(100), black(100)
#pieces = e, w, b
#variables =
#board = rectangle(up,down,left,right,
          [b, b, b, b, b, b, b, b]
          [b, b, b, b, b, b, b, b]
          [e, e, e, e, e, e, e, e]
          [e, e, e, e, e, e, e, e]
          [e, e, e, e, e, e, e, e]
          [e, e, e, e, e, e, e, e]
          [w, w, w, w, w, w, w, w]
          [w, w, w, w, w, w, w, w])

#anySquare = ((up* + down*)(left* + right*))

#turn(me; myPawn; opp; oppPawn; forward) =
    // Move
    ->me
    anySquare {myPawn} [e]
    forward ({e} + (left+right) {e,oppPawn})
    // Check win
    ->>
    [$ me=100] [$ opp=0]
    (
       {? forward} [myPawn]
     + {! forward} ->> {}
    )

#rules = (
      turn(white; w; black; b; up)
      turn(black; b; white; w; down)
    )*
```

Breakthrough in Ludii

```
(game "BreakThrough" (mode 2)
  (equipment
    {
      (chessBoard (square 8))
      (pawn
        P1
        N
        (or
          {
            (step F (in (to) (empty)))
            (step ForwardDiagonal (not (isFriend (who (to)))) (remove (to)))
          }
        )
      )
      (pawn
        P2
        S
        (or
          {
            (step F (in (to) (empty)))
            (step ForwardDiagonal (not (isFriend (who (to)))) (remove (to)))
          }
        )
      )
    }
  )
  (rules
    (start {(place "Pawn1" (expand (bottom))) (place "Pawn2" (expand (top)))})
    (play (byPiece))
    (or
      {
        (end (in (lastToMove) (top)) (result P1 Win))
        (end (in (lastToMove) (bottom)) (result P2 Win))
      }
    )
  )
)
```

Overview of the research

Syntactical translations

We are interested in automatic translation of game rules from one GGP language to the other

- Complicated and laborious task.
- Resulting rules are inefficient and verbose.
- Usually no extensive tests, optimizations, nor theoretical justification.

Game playing comparison

The goal is to experimentally assess language properties via the agent's performance when both agents belong to different GGP systems.

- Even more challenging and definitely more error-prone.
- Direct comparison: agents strength, requires a game manager bridge.
- Indirect: only static efficiency comparison.
- Inputs in various systems should match!

Case study: comparing the efficiency of Ludii and Regular Boardgames

An Empirical Evaluation of Two General Game Systems: Ludii and RBG; Piette, Stephenson, Soemers, Browne; 2019

- A recent comparison of three different GGP languages: Ludii, Regular Boardgames, and GDL.
- Focusing on the efficiency of reasoning.
- We performed a detailed analysis of the experiments, trying to reproduce them.
- Which lead to some interesting conclusions regarding conducting and reproducing experimental research.

We present this analysis in the hope of avoiding similar problems in any further research of this kind, allowing comparisons as fair as possible, which do not cause the need for questioning the results.

Reproduction attempt problems

No access to the original system.

- The version of the Ludii used for the experiments is unavailable.
- We performed an analysis based on publicly released (and newer) version (still close-sourced).

The majority of the compared games do not have the same rules

- Game rules analysis showed that only 5 out of 14 Ludii games have fully equivalent rules to those existing in RBG 1.0.
- We made an attempt to reimplement mismatching variants in RBG to correspond to the Ludii ones.

Unreported, non-uniform setup

- The results for GDL were obtained on different hardware than those for RBG and Ludii.
- For chess, GGP-Prover instead of a propnet was used.

Results of experimental comparison

The original and reproduced results of the efficiency of reasoning in RBG, Ludii, and GDL for the *flat Monte Carlo* test. The values are the *numbers of playouts per second*.

Results from (Piette et al. 2019)			
Game	**RBG 1.0**	**Ludii**	**GDL**
Amazons	625	**4,349**	185
Arimaa	0.11	**714**	*n/a*
Breakthrough	**16,694**	4,741	1,123
Chess	714	**720**	0.06
Connect-4	84,124	**94,077**	13,664
English draughts	**14,262**	8,135	872
Gomoku	2,212	**42,985**	927
Hex	5,787	**11,077**	*n/a*
Reversi	2,012	**2,081**	203
The mill game	7,423	**72,734**	*n/a*
Tic-tac-toe	400,000	**535,294**	85,319

Results from (Kowalski et al. 2019)			
Game	**RBG 1.0/1.0.1**	**Ludii 0.3.0**	**GDL propnet**
Amazons-orthodox	569	*n/a*	4
Amazons-split	**8,798**	3,859	365
Arimaa-orthodox	0.14	*n/a*	*n/a*
Arimaa-split	**666***	446†	*n/a*
Breakthrough	**19,916**	3,546	2,735
Chess-fifty move	**523***	14†	45
Connect-4	**190,171**	63,427	45,894
English draughts-split	**23,361***	7,111†	3,466
Gomoku-free style	2,430*	**26,878**	*n/a*
Hex	6,794	**10,625**	*n/a*
Reversi	**8,682**	1,312	373
The mill game	**10,102***	2,467†	*n/a*
Tic-tac-toe	**526,930**	422,836	104,500

This demonstrates how technical details and different methodology can switch conclusions to the opposite.

Findings

- With the task of comparing GDL's, there is even less common ground between various approaches.
- Early works have extremely small experimental sections, providing only a few results, usually not documented in detail.
- Information about the system specifications and the used algorithms are crucial.
- Proper game matching is a subtle cause that can have a tremendous impact on the final results.
- We think that the most natural definition of game "being the same" could be "have isomorphic game trees".

Conclusions and Recommendations

Recommendations

Provide the exact game definitions that were used.

When comparing different game descriptions of the same game, make sure that they are equivalent.

Choose the testset that maximally covers the ones in correlated research.

Run the test on a maximally idle system. If the test is designed to run on only one core, force this additionally to avoid misuse.

If possible, when comparing with an existing implementation, ask the authors to ensure the correct usage.

Question

How to include all those important technical details in limited-space, self-contained paper so they could be verified during the review process?

Recommendations

Provide the exact game definitions that were used.

When comparing different game descriptions of the same game, make sure that they are equivalent.

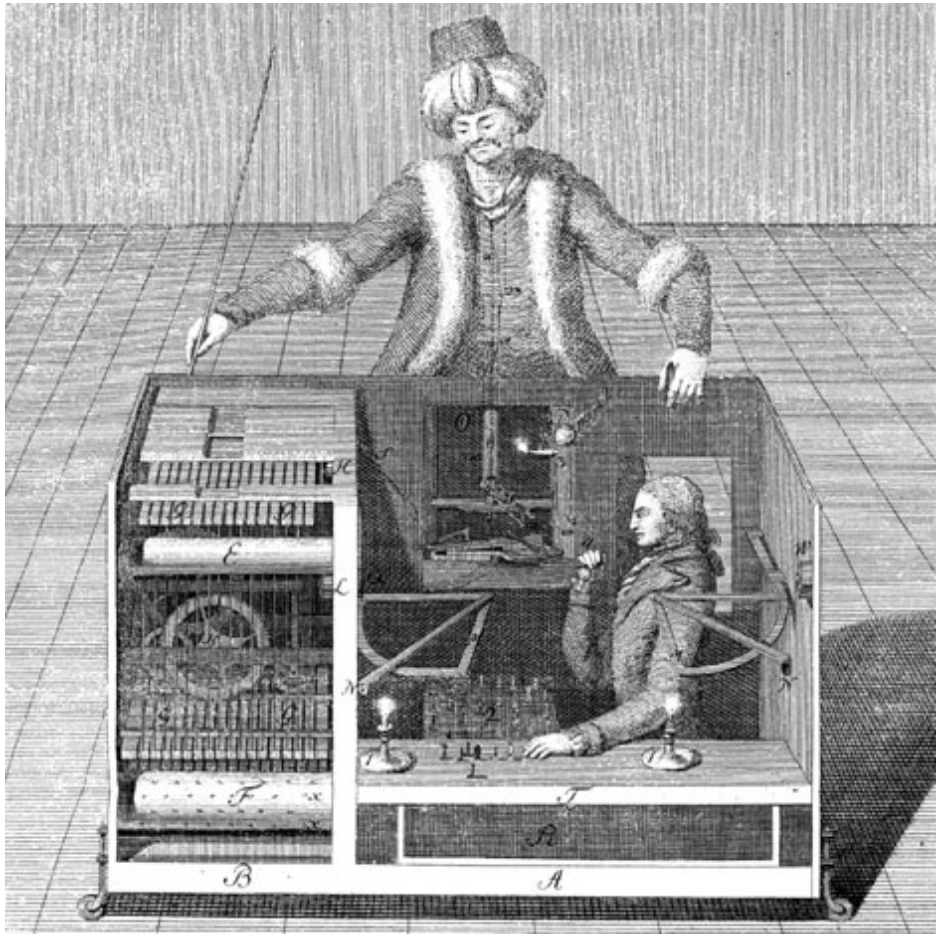
Choose the testset that maximally covers the ones in correlated research.

Run the test on a maximally idle system. If the test is designed to run on only one core, force this additionally to avoid misuse.

If possible, when comparing with an existing implementation, ask the authors to ensure the correct usage.

Question

How to include all those important technical details in limited-space, self-contained paper so they could be verified during the review process?



Thank you