WIKIPEDIA
The Free Encyclopedia

Game complexity

Combinatorial game theory measures **game complexity** in several ways:

1. State-space complexity (the number of legal game positions from the initial position),
2. Game tree size (total number of possible games),
3. Decision complexity (number of leaf nodes in the smallest decision tree for initial position),
4. Game-tree complexity (number of leaf nodes in the smallest full-width decision tree for initial position),
5. Computational complexity (asymptotic difficulty of a game as it grows arbitrarily large).

These measures involve understanding game positions, possible outcomes, and computation required for various game scenarios.

Measures of game complexity

State-space complexity

The **state-space complexity** of a game is the number of legal game positions reachable from the initial position of the game.[1]

When this is too hard to calculate, an upper bound can often be computed by also counting (some) illegal positions, meaning positions that can never arise in the course of a game.

Game tree size

The **game tree size** is the total number of possible games that can be played: the number of leaf nodes in the game tree rooted at the game's initial position.

The game tree is typically vastly larger than the state space because the same positions can occur in many games by making moves in a different order (for example, in a tic-tac-toe game with two X and one O on the board, this position could have been reached in two different ways depending on where the first X was placed). An upper bound for the size of the game tree can sometimes be computed by simplifying the game in a way that only increases the size of the game tree (for example, by allowing illegal moves) until it becomes tractable.

For games where the number of moves is not limited (for example by the size of the board, or by a rule about repetition of position) the game tree is generally infinite.

Decision trees

The next two measures use the idea of a *decision tree*, which is a subtree of the game tree, with each position labelled with "player A wins", "player B wins" or "drawn", if that position can be proved to have that value (assuming best play by both sides) by examining only other positions in the graph. (Terminal positions can be labelled directly; a position with player A to move can be labelled "player A wins" if any successor position is a win for A, or labelled "player B wins" if all successor positions are wins for B, or labelled "draw" if all successor positions are either drawn or wins for B. And correspondingly for positions with B to move.)

Decision complexity

Decision complexity of a game is the number of leaf nodes in the smallest decision tree that establishes the value of the initial position.

Game-tree complexity

The **game-tree complexity** of a game is the number of leaf nodes in the smallest *full-width* decision tree that establishes the value of the initial position.[1] A full-width tree includes all nodes at each depth.

This is an estimate of the number of positions one would have to evaluate in a minimax search to determine the value of the initial position.

It is hard even to estimate the game-tree complexity, but for some games an approximation can be given by raising the game's average branching factor b to the power of the number of plies d in an average game, or:

$GTC \geq b^d$.

Computational complexity

The **computational complexity** of a game describes the asymptotic difficulty of a game as it grows arbitrarily large, expressed in big O notation or as membership in a complexity class. This concept doesn't apply to particular games, but rather to games that have been generalized so they can be made arbitrarily large, typically by playing them on an n-by-n board. (From the point of view of computational complexity a game on a fixed size of board is a finite problem that can be solved in O(1), for example by a look-up table from positions to the best move in each position.)

The asymptotic complexity is defined by the most efficient (in terms of whatever computational resource one is considering) algorithm for solving the game; the most common complexity measure (computation time) is always lower-bounded by the logarithm of the asymptotic state-space complexity, since a solution algorithm must work for every possible state of the game. It will be upper-bounded by the complexity of any particular algorithm that works for the family of games. Similar remarks apply to the second-most commonly used complexity measure, the amount of space or computer memory used by the computation. It is not obvious that there is any lower bound on the space complexity for a typical game, because the algorithm need not store game states; however many games of interest are known to be PSPACE-hard, and it follows that their space complexity will be lower-bounded by the logarithm of the asymptotic state-space complexity as well (technically the bound is only a polynomial in this quantity; but it is usually known to be linear).

- The depth-first minimax strategy will use computation time proportional to game's tree-complexity, since it must explore the whole tree, and an amount of memory polynomial in the logarithm of the tree-complexity, since the algorithm must always store one node of the tree at each possible move-depth, and the number of nodes at the highest move-depth is precisely the tree-complexity.
- Backward induction will use both memory and time proportional to the state-space complexity as it must compute and record the correct move for each possible position.

Example: tic-tac-toe (noughts and crosses)

For tic-tac-toe, a simple upper bound for the size of the state space is $3^9 = 19{,}683$. (There are three states for each cell and nine cells.) This count includes many illegal positions, such as a position with five crosses and no noughts, or a position in which both players have a row of three. A more careful count, removing these illegal positions, gives 5,478.[2][3] And when rotations and reflections of positions are considered identical, there are only 765 essentially different positions.

To bound the game tree, there are 9 possible initial moves, 8 possible responses, and so on, so that there are at most 9! or 362,880 total games. However, games may take less than 9 moves to resolve, and an exact enumeration gives 255,168 possible games. When rotations and reflections of positions are considered the same, there are only 26,830 possible games.

The computational complexity of tic-tac-toe depends on how it is generalized. A natural generalization is to *m*,*n*,*k*-games: played on an *m* by *n* board with winner being the first player to get *k* in a row. It is immediately clear that this game can be solved in DSPACE(mn) by searching the entire game tree. This places it in the important complexity class PSPACE. With some more work it can be shown to be PSPACE-complete.[4]

Complexities of some well-known games

Due to the large size of game complexities, this table gives the ceiling of their logarithm to base 10. (In other words, the number of digits). All of the following numbers should be considered with caution: seemingly-minor changes to the rules of a game can change the numbers (which are often rough estimates anyway) by tremendous factors, which might easily be much greater than the numbers shown.

Note: ordered by game tree size

Game	Board size (positions)	State-space complexity (as log to base 10)	Game-tree complexity (as log to base 10)	Average game length (plies)	Branching factor	Ref	Complexity class of suitable generalized game
Tic-tac-toe	9	3	5	9	4		PSPACE-complete[5]
Sim	15	3	8	14	3.7		PSPACE-complete[6]
Pentominoes	64	12	18	10	75	[7][8]	?, but in PSP
Kalah[9]	14	13	18		50	[7]	Generalizat is unclear
Connect Four	42	13	21	36	4	[1][10]	?, but in PSP
Domineering (8 × 8)	64	15	27	30	8	[7]	?, but in PSP in P for certa dimensions
Congkak	14	15	33			[7]	
English draughts (8x8) (checkers)	32	20 or 18	40	70	2.8	[1][12][13]	EXPTIME-complete[14
Awari[15]	12	12	32	60	3.5	[1]	Generalizat is unclear
Qubic	64	30	34	20	54.2	[1]	PSPACE-complete[5]
Double dummy bridge[nb 1]	(52)	<17	<40	52	5.6		PSPACE-complete[16
Fanorona	45	21	46	44	11	[17]	?, but in EXPT
Nine men's morris	24	10	50	50	10	[1]	?, but in EXPT
Tablut	81	27				[18]	
International draughts (10x10)	50	30	54	90	4	[1]	EXPTIME-complete[14
Chinese checkers (2 sets)	121	23			180	[19]	EXPTIME-complete [2
Chinese checkers (6 sets)	121	78			600	[19]	EXPTIME-complete [2
Reversi (Othello)	64	28	58	58	10	[1]	PSPACE-complete[21
OnTop (2p base game)	72	88	62	31	23.77	[22]	
Lines of Action	64	23	64	44	29	[23]	?, but in EXPT
Gomoku (15x15, freestyle)	225	105	70	30	210	[1]	PSPACE-complete[5]
Hex (11x11)	121	57	98	50	96	[7]	PSPACE-complete[5]
Chess	64	44	123	70	35	[24]	EXPTIME-complete (without 50-move drawi rule)[25]
Bejeweled and Candy Crush (8x8)	64	<50			70	[26]	NP-hard
GIPF	37	25	132	90	29.3	[27]	
Connect6	361	172	140	30	46000	[28]	PSPACE-complete[29
Backgammon	28	20	144	55	250	[30]	Generalizat is unclear
Xiangqi	90	40	150	95	38	[1][31][32]	?, believed to EXPTIME complete
Abalone	61	25	154	87	60	[33][34]	PSPACE-ha and in EXPTIME

Game	Board size (positions)	State-space complexity (as log to base 10)	Game-tree complexity (as log to base 10)	Average game length (plies)	Branching factor	Ref	Complexit class of suitable generalize game
Havannah	271	127	157	66	240	[7][35]	PSPACE-complete[36
Twixt	572	140	159	60	452	[37]	
Janggi	90	44	160	100	40	[32]	?, believed to EXPTIME complete
Quoridor	81	42	162	91	60	[38]	?, but in PSP
Carcassonne (2p base game)	72	>40	195	71	55	[39]	Generalizat is unclear
Amazons (10x10)	100	40	212	84	374 or 299[40]	[41][42]	PSPACE-complete[43
Shogi	81	71	226	115	92	[31][44]	EXPTIME-complete[45
Thurn and Taxis (2 player)	33	66	240	56	879	[46]	
Go (19x19)	361	170	360	150	250	[1][47][48]	EXPTIME-complete (without the superko rule)[49]
Arimaa	64	43	402	92	17281	[50][51][52]	?, but in EXPT
Stratego	92	115	535	381	21.739	[53]	
Infinite chess	infinite	infinite	infinite	infinite	infinite	[54]	Unknown, b mate-in-n i decidable[5
Magic: The Gathering						[56]	AH-hard[57]
Wordle	5	12,972		6		[58]	NP-hard, unknown if PSPACE-complete w parametizat

Notes

1. Double dummy bridge (i.e., double dummy problems in the context of contract bridge) is not a proper board game but has a similar game tree, and is studied in computer bridge. The bridge table can be regarded as having one slot for each player and trick to play a card in, which corresponds to board size 52. Game-tree complexity is a very weak upper bound: 13! to the power of 4 players regardless of legality. State-space complexity is for one given deal; likewise regardless of legality but with many transpositions eliminated. The last 4 plies are always forced moves with branching factor 1.

References

1. Victor Allis (1994). *Searching for Solutions in Games and Artificial Intelligence* (https://project.dke.maastrichtuniversity.nl/games/files/phd/SearchingForSolutions.pdf) (PDF) (Ph.D. thesis). University of Limburg, Maastricht, The Netherlands. ISBN 90-900748-8-0.
2. "combinatorics - TicTacToe State Space Choose Calculation" (https://math.stackexchange.com/questions/485752/tictactoe-state-space-choose-calculation). *Mathematics Stack Exchange*. Retrieved 2020-04-08.
3. T, Brian (October 20, 2018). "Btsan/generate_tictactoe" (https://github.com/Btsan/generate_tictactoe). *GitHub*. Retrieved 2020-04-08.
4. Stefan Reisch (1980). "Gobang ist PSPACE-vollständig (Gobang is PSPACE-complete)". *Acta Informatica*. **13** (1): 59–66. doi:10.1007/bf00288536 (https://doi.org/10.1007%2Fbf00288536). S2CID 21455572 (https://api.semanticscholar.org/CorpusID:21455572).
5. Stefan Reisch (1981). "Hex ist PSPACE-vollständig (Hex is PSPACE-complete)". *Acta Inform* (15): 167–191.
6. Slany, Wolfgang (2000). "The complexity of graph Ramsey games". In Marsland, T. Anthony; Frank, Ian (eds.). *Computers and Games, Second International Conference, CG 2000, Hamamatsu, Japan, October 26-28, 2000, Revised Papers*. Lecture Notes in Computer Science. Vol. 2063. Springer. pp. 186–203. doi:10.1007/3-540-45579-5_12 (https://doi.org/10.1007%2F3-540-45579-5_12).
7. H. J. van den Herik; J. W. H. M. Uiterwijk; J. van Rijswijck (2002). "Games solved: Now and in the future" (https://doi.org/10.1016%2FS0004-3702%2801%2900152-7). *Artificial Intelligence*. **134** (1–2): 277–311. doi:10.1016/S0004-3702(01)00152-7 (https://doi.org/10.1016%2FS0004-3702%2801%2900152-7).
8. Orman, Hilarie K. (1996). "Pentominoes: a first player win" (https://www.msri.org/publications/books/Book29/files/orman.pdf) (PDF). In Nowakowski, Richard J. (ed.). *Games of No Chance: Papers from the Combinatorial Games Workshop held in Berkeley, CA, July 11–21, 1994*. Mathematical Sciences Research Institute Publications. Vol. 29. Cambridge University Press. pp. 339–344. ISBN 0-521-57411-0. MR 1427975 (https://mathscinet.ams.org/mathscinet-getitem?mr=1427975).
9. See van den Herik et al for rules.
10. John Tromp (2010). "John's Connect Four Playground" (https://tromp.github.io/c4/c4.html).
11. Lachmann, Michael; Moore, Cristopher; Rapaport, Ivan (2002). "Who wins Domineering on rectangular boards?". In Nowakowski, Richard (ed.). *More Games of No Chance: Proceedings of the 2nd Combinatorial Games Theory Workshop held in Berkeley, CA, July 24–28, 2000*. Mathematical Sciences Research Institute Publications. Vol. 42. Cambridge University Press. pp. 307–315. ISBN 0-521-80832-4. MR 1973019 (https://mathscinet.ams.org/mathscinet-getitem?mr=1973019).

12. Jonathan Schaeffer; et al. (July 6, 2007). "Checkers is Solved" (https://doi.org/10.1126%2Fscience.1144079). *Science*. **317** (5844): 1518–1522. Bibcode:2007Sci...317.1518S (https://ui.adsabs.harvard.edu/abs/2007Sci...317.1518S). doi:10.1126/science.1144079 (https://doi.org/10.1126%2Fscience.1144079). PMID 17641166 (https://pubmed.ncbi.nlm.nih.gov/17641166). S2CID 10274228 (https://api.semanticscholar.org/CorpusID:10274228).
13. Schaeffer, Jonathan (2007). "Game over: Black to play and draw in checkers" (https://web.archive.org/web/20160403093928/https://ticc.uvt.nl/icga/journal/contents/Schaeffer07-01-08.pdf) (PDF). *ICGA Journal*. **30** (4): 187–197. doi:10.3233/ICG-2007-30402 (https://doi.org/10.3233%2FICG-2007-30402). Archived from the original (https://ticc.uvt.nl/icga/journal/contents/Schaeffer07-01-08.pdf) (PDF) on 2016-04-03.
14. J. M. Robson (1984). "N by N checkers is Exptime complete". *SIAM Journal on Computing*. **13** (2): 252–267. doi:10.1137/0213018 (https://doi.org/10.1137%2F0213018).
15. See Allis 1994 for rules
16. Bonnet, Edouard; Jamain, Florian; Saffidine, Abdallah (2013). "On the complexity of trick-taking card games" (https://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6920). In Rossi, Francesca (ed.). *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*. IJCAI/AAAI. pp. 482–488.
17. M.P.D. Schadd; M.H.M. Winands; J.W.H.M. Uiterwijk; H.J. van den Herik; M.H.J. Bergsma (2008). "Best Play in Fanorona leads to Draw" (https://dke.maastrichtuniversity.nl/m.winands/documents/Fanorona.pdf) (PDF). *New Mathematics and Natural Computation*. **4** (3): 369–387. doi:10.1142/S1793005708001124 (https://doi.org/10.1142%2FS1793005708001124).
18. Andrea Galassi (2018). "An Upper Bound on the Complexity of Tablut" (http://ai.unibo.it/biblio/galassiTablutComplexity).
19. G.I. Bell (2009). "The Shortest Game of Chinese Checkers and Related Problems". *Integers*. **9**. arXiv:0803.1245 (https://arxiv.org/abs/0803.1245). Bibcode:2008arXiv0803.1245B (https://ui.adsabs.harvard.edu/abs/2008arXiv0803.1245B). doi:10.1515/INTEG.2009.003 (https://doi.org/10.1515%2FINTEG.2009.003). S2CID 17141575 (https://api.semanticscholar.org/CorpusID:17141575).
20. Kasai, Takumi; Adachi, Akeo; Iwata, Shigeki (1979). "Classes of pebble games and complete problems". *SIAM Journal on Computing*. **8** (4): 574–586. doi:10.1137/0208046 (https://doi.org/10.1137%2F0208046). MR 0573848 (https://mathscinet.ams.org/mathscinet-getitem?mr=0573848). Proves completeness of the generalization to arbitrary graphs.
21. Iwata, Shigeki; Kasai, Takumi (1994). "The Othello game on an $n \times n$ board is PSPACE-complete" (https://doi.org/10.1016%2F0304-3975%2894%2990131-7). *Theoretical Computer Science*. **123** (2): 329–340. doi:10.1016/0304-3975(94)90131-7 (https://doi.org/10.1016%2F0304-3975%2894%2990131-7). MR 1256205 (https://mathscinet.ams.org/mathscinet-getitem?mr=1256205).
22. Robert Briesemeister (2009). *Analysis and Implementation of the Game OnTop* (https://project.dke.maastrichtuniversity.nl/games/files/msc/Briesemeister_Thesis.pdf) (PDF) (Thesis). Maastricht University, Dept of Knowledge Engineering.
23. Mark H.M. Winands (2004). *Informed Search in Complex Games* (https://dke.maastrichtuniversity.nl/m.winands/documents/informed_search.pdf) (PDF) (Ph.D. thesis). Maastricht University, Maastricht, The Netherlands. ISBN 90-5278-429-9.
24. The size of the state space and game tree for chess were first estimated in Claude Shannon (1950). "Programming a Computer for Playing Chess" (https://web.archive.org/web/20100706211229/http://archive.computerhistory.org/projects/chess/related_materials/text/2-0%20and%202-1.Programming_a_computer_for_playing_chess.shannon/2-0%20and%202-1.Programming_a_computer_for_playing_chess.shannon.062303002.pdf) (PDF). *Philosophical Magazine*. **41** (314). Archived from the original (http://archive.computerhistory.org/projects/chess/related_materials/text/2-0%20and%202-1.Programming_a_computer_for_playing_chess.shannon/2-0%20and%202-1.Programming_a_computer_for_playing_chess.shannon.062303002.pdf) (PDF) on 2010-07-06. Shannon gave estimates of 10^{43} and 10^{120} respectively, smaller than the upper bound in the table, which is detailed in Shannon number.
25. Fraenkel, Aviezri S.; Lichtenstein, David (1981). "Computing a perfect strategy for $n \times n$ chess requires time exponential in n" (https://doi.org/10.1016%2F0097-3165%2881%2990016-9). *Journal of Combinatorial Theory, Series A*. **31** (2): 199–214. doi:10.1016/0097-3165(81)90016-9 (https://doi.org/10.1016%2F0097-3165%2881%2990016-9). MR 0629595 (https://mathscinet.ams.org/mathscinet-getitem?mr=0629595).
26. Gualà, Luciano; Leucci, Stefano; Natale, Emanuele (2014). "Bejeweled, Candy Crush and other match-three games are (NP-)hard". *2014 IEEE Conference on Computational Intelligence and Games, CIG 2014, Dortmund, Germany, August 26-29, 2014*. IEEE. pp. 1–8. arXiv:1403.5830 (https://arxiv.org/abs/1403.5830). doi:10.1109/CIG.2014.6932866 (https://doi.org/10.1109%2FCIG.2014.6932866).
27. Diederik Wentink (2001). *Analysis and Implementation of the game Gipf* (https://project.dke.maastrichtuniversity.nl/games/files/msc/Wentink_thesis.pdf) (PDF) (Thesis). Maastricht University.
28. Chang-Ming Xu; Ma, Z.M.; Jun-Jie Tao; Xin-He Xu (2009). "Enhancements of proof number search in connect6". *2009 Chinese Control and Decision Conference*. p. 4525. doi:10.1109/CCDC.2009.5191963 (https://doi.org/10.1109%2FCCDC.2009.5191963). ISBN 978-1-4244-2722-2. S2CID 20960281 (https://api.semanticscholar.org/CorpusID:20960281).
29. Hsieh, Ming Yu; Tsai, Shi-Chun (October 1, 2007). "On the fairness and complexity of generalized k -in-a-row games" (http://dl.acm.org/citation.cfm?id=1290195.1290250). *Theoretical Computer Science*. **385** (1–3): 88–100. doi:10.1016/j.tcs.2007.05.031 (https://doi.org/10.1016%2Fj.tcs.2007.05.031). Retrieved 2018-04-12 – via dl.acm.org.
30. Tesauro, Gerald (May 1, 1992). "Practical issues in temporal difference learning" (https://doi.org/10.1007%2FBF00992697). *Machine Learning*. **8** (3–4): 257–277. doi:10.1007/BF00992697 (https://doi.org/10.1007%2FBF00992697).
31. Shi-Jim Yen, Jr-Chang Chen; Tai-Ning Yang; Shun-Chin Hsu (March 2004). "Computer Chinese Chess" (https://web.archive.org/web/20070614111609/http://www.csie.ndhu.edu.tw/~sjyen/Papers/2004CCC.pdf) (PDF). *International Computer Games Association Journal*. **27** (1): 3–18. doi:10.3233/ICG-2004-27102 (https://doi.org/10.3233%2FICG-2004-27102). S2CID 10336286 (https://api.semanticscholar.org/CorpusID:10336286). Archived from the original (http://www.csie.ndhu.edu.tw/~sjyen/Papers/2004CCC.pdf) (PDF) on 2007-06-14.
32. Donghwi Park (2015). "Space-state complexity of Korean chess and Chinese chess". arXiv:1507.06401 (https://arxiv.org/abs/1507.06401) [math.GM (https://arxiv.org/archive/math.GM)].
33. Chorus, Pascal. "Implementing a Computer Player for Abalone Using Alpha-Beta and Monte-Carlo Search" (https://project.dke.maastrichtuniversity.nl/games/files/msc/pcreport.pdf) (PDF). Dept of Knowledge Engineering, Maastricht University. Retrieved 2012-03-29.
34. Kopczynski, Jacob S (2014). *Pushy Computing: Complexity Theory and the Game Abalone* (Thesis). Reed College.
35. Joosten, B. "Creating a Havannah Playing Agent" (https://project.dke.maastrichtuniversity.nl/games/files/bsc/bscHavannah.pdf) (PDF). Retrieved 2012-03-29.
36. E. Bonnet; F. Jamain; A. Saffidine (March 25, 2014). "Havannah and TwixT are PSPACE-complete". arXiv:1403.6518 (https://arxiv.org/abs/1403.6518) [cs.CC (https://arxiv.org/archive/cs.CC)].
37. Kevin Moesker (2009). *Txixt: Theory, Analysis, and Implementation* (https://project.dke.maastrichtuniversity.nl/games/files/msc/Thesis_Moesker.pdf) (PDF) (Thesis). Faculty of Humanities and Sciences of Maastricht University.
38. Lisa Glendenning (May 2005). *Mastering Quoridor* (https://web.archive.org/web/20120315192840/http://hyperion.cs.washington.edu/attachments/15/glendenning_ugrad_thesis.pdf) (PDF). Computer Science (B.Sc. thesis). University of New Mexico. Archived from the original (http://hyperion.cs.washington.edu/attachments/15/glendenning_ugrad_thesis.pdf) (PDF) on 2012-03-15.
39. Cathleen Heyden (2009). *Implementing a Computer Player for Carcassonne* (https://project.dke.maastrichtuniversity.nl/games/files/msc/MasterThesisCarcassonne.pdf) (PDF) (Thesis). Maastricht University, Dept of Knowledge Engineering.
40. The lower branching factor is for the second player.
41. Kloetzer, Julien; Iida, Hiroyuki; Bouzy, Bruno (2007). "The Monte-Carlo approach in Amazons" (https://helios2.mi.parisdescartes.fr/~Bouzy/publications/KIB-MCAmazons-CGW07.pdf) (PDF). *Computer Games Workshop, Amsterdam, the Netherlands, 15-17 June 2007*. pp. 185–192.

42. P. P. L. M. Hensgens (2001). "A Knowledge-Based Approach of the Game of Amazons" (https://project.dke.maastrichtuniversity.nl/games/files/msc/Hensgens_thesis.pdf) (PDF). Universiteit Maastricht, Institute for Knowledge and Agent Technology.
43. R. A. Hearn (February 2, 2005). "Amazons is PSPACE-complete". arXiv:cs.CC/0502013 (https://arxiv.org/abs/cs.CC/0502013).
44. Hiroyuki Iida; Makoto Sakuta; Jeff Rollason (January 2002). "Computer shogi" (https://doi.org/10.1016%2FS0004-3702%2801%2900157-6). *Artificial Intelligence*. **134** (1–2): 121–144. doi:10.1016/S0004-3702(01)00157-6 (https://doi.org/10.1016%2FS0004-3702%2801%2900157-6).
45. H. Adachi; H. Kamekawa; S. Iwata (1987). "Shogi on n × n board is complete in exponential time". *Trans. IEICE*. J70-D: 1843–1852.
46. F.C. Schadd (2009). *Monte-Carlo Search Techniques in the Modern Board Game Thurn and Taxis* (https://web.archive.org/web/20210114164554/https://project.dke.maastrichtuniversity.nl/games/files/msc/Fschadd_thesis.pdf) (PDF) (Thesis). Maastricht University. Archived from the original (https://project.dke.maastrichtuniversity.nl/games/files/msc/Fschadd_thesis.pdf) (PDF) on 2021-01-14.
47. John Tromp; Gunnar Farnebäck (2007). "Combinatorics of Go" (https://tromp.github.io/go/gostate.ps). This paper derives the bounds 48<log(log(*N*))<171 on the number of possible games *N*.
48. John Tromp (2016). "Number of legal Go positions" (https://tromp.github.io/go/legal.html).
49. J. M. Robson (1983). "The complexity of Go". *Information Processing; Proceedings of IFIP Congress*. pp. 413–417.
50. Christ-Jan Cox (2006). "Analysis and Implementation of the Game Arimaa" (https://project.dke.maastrichtuniversity.nl/games/files/msc/Cox_thesis1.pdf) (PDF).
51. David Jian Wu (2011). "Move Ranking and Evaluation in the Game of Arimaa" (http://icosahedral.net/downloads/djwuthesis.pdf) (PDF).
52. Brian Haskin (2006). "A Look at the Arimaa Branching Factor" (http://arimaa.janzert.com/bf_study/).
53. A.F.C. Arts (2010). *Competitive Play in Stratego* (https://project.dke.maastrichtuniversity.nl/games/files/msc/Arts_thesis.pdf) (PDF) (Thesis). Maastricht.
54. CDA Evans and Joel David Hamkins (2014). "Transfinite game values in infinite chess". arXiv:1302.4377 (https://arxiv.org/abs/1302.4377) [math.LO (https://arxiv.org/archive/math.LO)].
55. Stefan Reisch, Joel David Hamkins, and Phillipp Schlicht (2012). "The mate-in-n problem of infinite chess is decidable". *Conference on Computability in Europe*: 78–88. arXiv:1201.5597 (https://arxiv.org/abs/1201.5597).
56. Alex Churchill, Stella Biderman, and Austin Herrick (2020). "Magic: the Gathering is Turing Complete". arXiv:1904.09828 (https://arxiv.org/abs/1904.09828) [cs.AI (https://arxiv.org/archive/cs.AI)].
57. Stella Biderman (2020). "Magic: the Gathering is as Hard as Arithmetic". arXiv:2003.05119 (https://arxiv.org/abs/2003.05119) [cs.AI (https://arxiv.org/archive/cs.AI)].
58. Lokshtanov, Daniel; Subercaseaux, Bernardo (May 14, 2022). "Wordle is NP-hard". arXiv:2203.16713 (https://arxiv.org/abs/2203.16713) [cs.CC (https://arxiv.org/archive/cs.CC)].

See also

- Go and mathematics
- Solved game
- Solving chess
- Shannon number
- list of NP-complete games and puzzles
- list of PSPACE-complete games and puzzles

External links

- David Eppstein's Computational Complexity of Games and Puzzles (http://www.ics.uci.edu/~eppstein/cgt/hard.html)