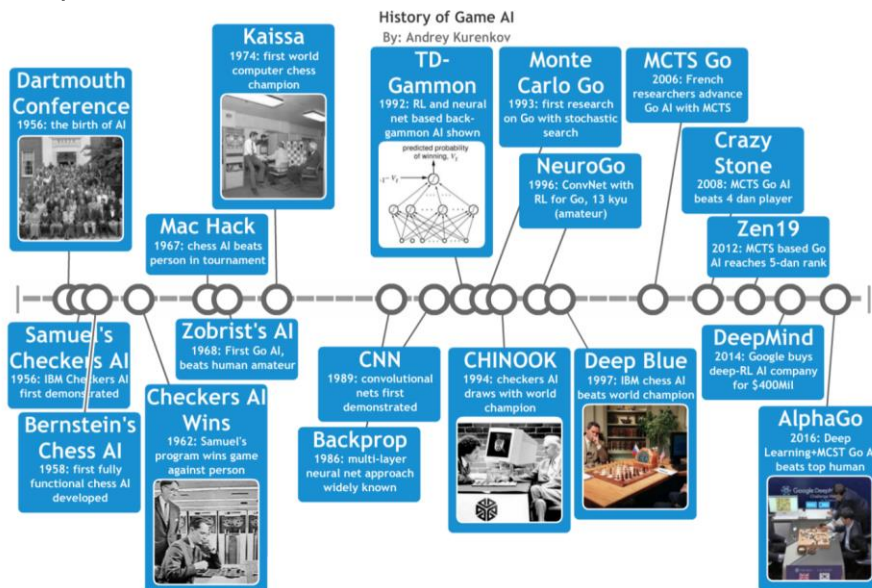


History of Learning in Games

History of Game AI

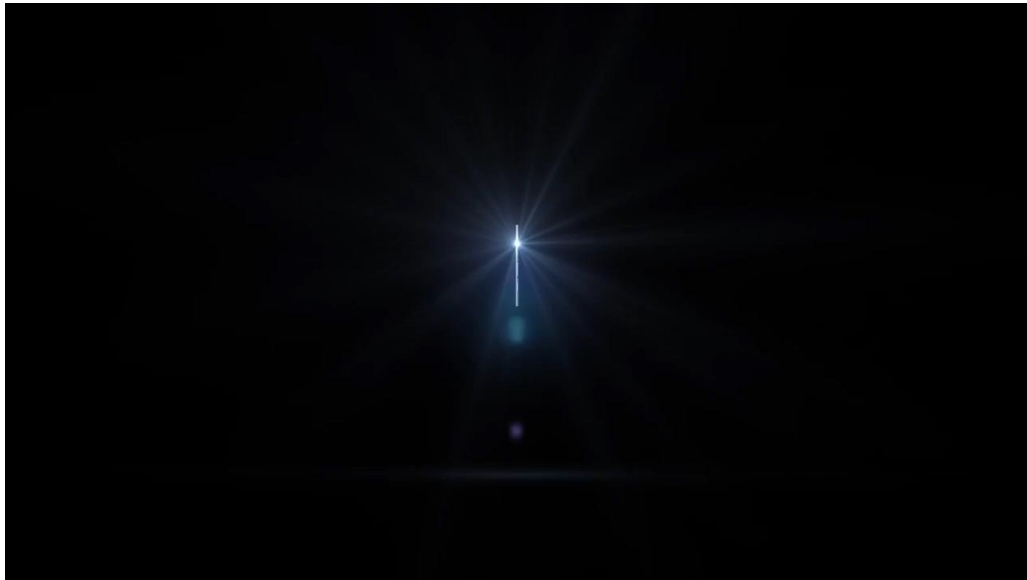


History has intertwined learning and programming

- Standard way to make a player
 - LMG
 - Eval function
 - Minimax (alpha-beta)
- Learning way to make a player
 - LMG
 - Parameterized eval function
 - Learn from human expert database
 - Learn from self-play
 - (optional Minimax or expectimax)

Menace

- In 1960 Donald Michie designed *MENACE*, the Matchbox Educable Noughts and Crosses Engine, a *collection of matchboxes that learned* to play a skilled game of tic-tac-toe.
- MENACE was a pioneering experiment in machine learning, demonstrating that a nonliving system could modify its behavior based on positive and negative reinforcement.
- MENACE did not involve a computer.



TTT is solved

- To be solved, a program will essentially have a policy of a move for every position.
 - TTT 5478
 - TTT 765 (with 8-way compression of positions)

A Checkered History

- Arthur Samuel, 1959
 - Learning algorithm used to build computer checkers player
 - Beat a 'master'
- Patrick Prosser, 1970s
 - Undergraduate project
 - Probably state of the art at that time
- Jonathan Schaeffer, 1990s
 - **Chinook** is the World Man-Machine champion
- Jonathan Schaeffer, 2007
 - Checkers is SOLVED

7

Example: Samuel's Checker-Playing Program

- It uses a linear evaluation function

$$f(n) = a_1x_1(n) + a_2x_2(n) + \dots + a_mx_m(n)$$

For example: $f = 6K + 4M + U$

- **K** = King Advantage
- **M** = Man Advantage
- **U** = Undenied Mobility Advantage (number of moves that Max has that Min can't jump after)

8

Samuel's Checker Player

- In learning mode
 - Computer acts as 2 players: **A** and **B**
 - **A** adjusts its coefficients after every move
 - **B** uses the static utility function
 - If **A** wins, its function is given to **B**

9

Samuel's Checker Player

- How does **A** change its function?
 1. Coefficient replacement
 - $\Delta(\text{node}) = \text{backed-up value}(\text{node}) - \text{initial value}(\text{node})$
 - if $\Delta > 0$ then terms that contributed **positively** are given more weight and terms that contributed negatively get less weight
 - if $\Delta < 0$ then terms that contributed **negatively** are given more weight and terms that contributed positively get less weight

10

Samuel's Checker Player

- **How does A change its function?**

- 2. Term Replacement

- 38 terms altogether

- 16 used in the utility function at any one time

Terms that **consistently correlate low** with the function value are removed and added to the end of the term queue.

They are replaced by terms from the front of the term queue.

11

What Happened in Checkers

- Samuels got a faster computer
 - 15k ops/minute!
- IBM held an exhibition match against human master
- The Computer Won
- Media claimed Checkers was SOLVED!

12

SOLVED versus Superhuman

- Solving a game means storing or computing correct win/loss values for all possible positions...
- Human-competitive means just that
- Superhuman means consistently beating the best human players.
 - With more CPU time for plies, a competitive player can go superhuman,

13

SCIENCE, 2007

Checkers Is Solved

Jonathan Schaeffer^{*}, Neil Burch, Yngvi Björnsson[†], Akihiro Kishimoto[‡], Martin Müller, Robert Lake, Paul Lu, Steve Sutphen

^{*}† Present address: Department of Computer Science, Reykjavik University, Reykjavik, Kringlan 1, IS-103, Iceland.

[‡] Present address: Department of Media Architecture, Future University, Hakodate, 116-2 Kamedanakano-cho Hakodate Hokkaido, 041-8655, Japan.

• See all authors and affiliations

Science 14 Sep 2007;
Vol. 317; Issue 5844, pp. 1518-1522
DOI: 10.1126/science.1144079

[Article](#)
[Figures & Data](#)
[Info & Metrics](#)
[eLetters](#)
[PDF](#)

Abstract

The game of checkers has roughly 500 billion billion possible positions (5×10^{20}). The task of solving the game, determining the final result in a game with no mistakes made by either player, is daunting. Since 1989, almost continuously, dozens of computers have been working on solving checkers, applying state-of-the-art artificial intelligence techniques to the proving process. This paper announces that checkers is now solved: Perfect play by both sides leads to a draw. This is the most challenging popular game to be solved to date, roughly one million times as complex as Connect Four. Artificial intelligence technology has been used to generate strong heuristic-based game-playing programs, such as Deep Blue for chess. Solving a game takes this to the next level by replacing the heuristics with perfection.

Jonathan Schaeffer

TOP AIGAME SCIENTIST



15

Berliner BKG 1978

- [I. Why Yet Another Game?](#)
- [II. The Structure of BKG](#)
- [III. Basic Procedures](#)
 - [A. The Move Generator](#)
 - [B. Special Program Functions](#)
 - [1. Doubling](#)
 - [2. Resigning](#)
 - [C. Evaluation](#)
- [IV. The Evaluation Procedures](#)
 - [A. Blot Danger Calculation](#)
 - [B. Blockading Factor](#)
 - [C. The Running Game](#)
 - [D. Bearing Off](#)
 - [E. Spacing of Men Around the Board](#)
 - [F. Other Variables](#)
- [V. The Evaluation Process](#)
- [VI. State-Classes and Their Utilization](#)
- [VII. Testing of BKG](#)
- [VIII. Use of Simulation Facility](#)
- [IX. A Game](#)
- [References](#)

Why Backgammon?

- The thing that makes backgammon an interesting object of study of AI is that in any given position (of which are 10^{20} ([Levner 1976](#))), there are 21 possible combinations that the throw of two dice can produce. Each of these can be played legally in the average board position about 40 different ways. Thus if one were to investigate a backgammon position by tree searching it would be necessary to deal with a branching factor of more than 800 (!!) at every node. Clearly, this is completely impractical.

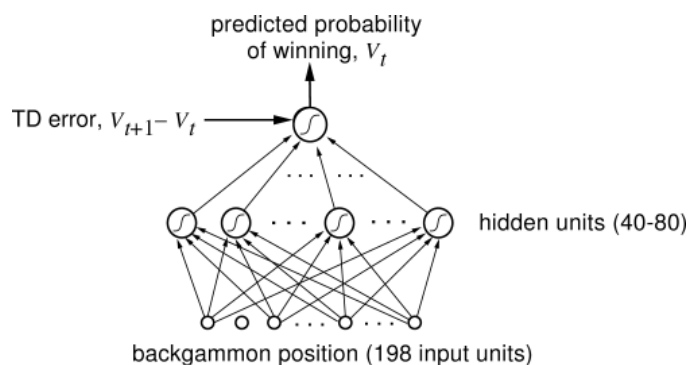
Neurogammon (Tesauro 1990)

- Neurogammon faced five opponents in the backgammon competition at the Olympiad: three commercial programs (Video Gammon [USA], Saitek Backgammon [Netherlands], and Mephisto Backgammon [West Germany]) and two non-commercial programs (Backbrain [Sweden] and AI Backgammon [USA]). Hans Berliner's BKG program was not entered in the competition. All five programs apparently used single-ply search algorithms with conventional hand-crafted evaluation functions. In matches to 11 points, Neurogammon defeated Video Gammon by 12-7, Mephisto by 12-5, Saitek by 12-9, Backbrain by 11-4, and AI Backgammon by 16-1. With five match victories and no losses, Neurogammon convincingly won the gold medal for backgammon at the Computer Olympiad.

Neurogammon Predicted TD Gammon

- Another direction which seems very promising is to abandon supervised learning altogether, and instead to learn by playing out a sequence of moves against an opponent and observing the outcome. Such a learning system could learn on its own, without the aid of an intelligent teacher, and in principle could continue to improve until it surpassed even the best human experts. Connectionist algorithms such as Sutton's Temporal-Difference algorithm [8] could be used in such an approach. It is not known whether such algorithms are efficient enough to tackle a problem of the scale and complexity of backgammon.

Neurogammon and TD Gammon



How many Backgammon Positions?

- Calculated in 1976 by David Lavner:

111,171,500,447,973,849,600

- ◆ How could a Neural Network with 4000 weights interpolate 100 quintillion values???

I didn't believe in TD Gammon

- I was superior player since high-school
 - Winning a money contests in bars and hustling in college
- I felt there were 7 +/- "styles" where humans excelled
 - Not "one" to rule them all.
- Self-play only worked sometimes and usually petered out
- Why would it work for Backgammon?

Setup for Pollack & Blair HCGammon

- 197 Inputs, 20 hidden units, 1 output
- 4 bits * 2 players * 24 pips, 2bar, 2off, race
- Generate Legal Moves, Apply network
- Use Competitive fitness HILLCLIMBING
- champion = Random Weights
- Forever do:
 - Challenger = Champion + noise
 - If challenger > champion (~20 games)
 - then champion = challenger

Results of First Trial: Buster Douglas Effect

- Tested (off-line) against PUBEVAL.C, Tesauro's "Average Player" LINEAR Evaluation function.
- TDGAMMON was protected IBM Intellectual Property!
- Observed rapid Improvement, then catastrophic wandering.
- Need to either increase #games to prevent LN, or decrease its effect!

Buster Douglas Effect: Reset the championship level

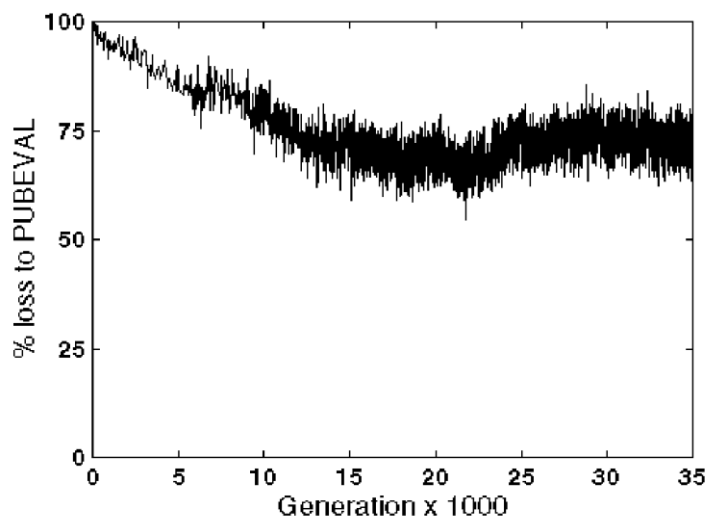
- Upset victory over Mike Tyson in Feb 1990
- Lost the title in oct 1990 to Evander Holyfield



Second Trial of HCGammon

- Played 2 pairs of symmetric games, challenger had to win 3/4
- used momentum to avoid catastrophe:
- Champion=random weights
- Forever do
- $\text{Challenger} = \text{Champion} + \text{noise}$
- If challenger > champion then
- $\text{champ} = .95 \text{ champ} + .05 \text{ challenger}$

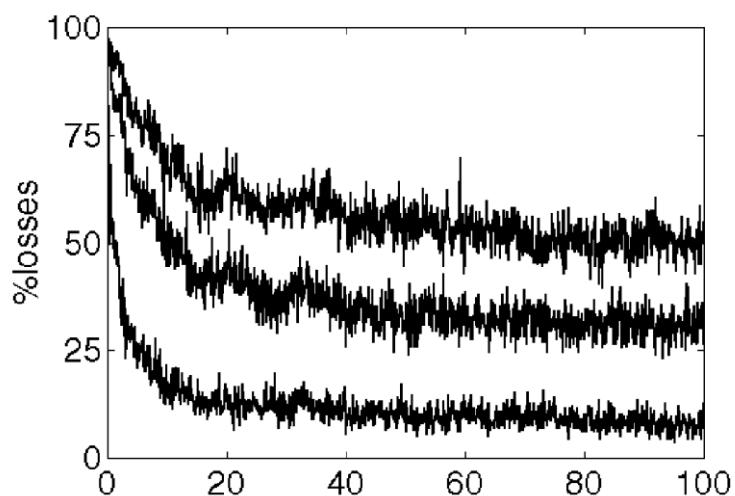
Evaluation against PUBEVAL for 35,000 cycles



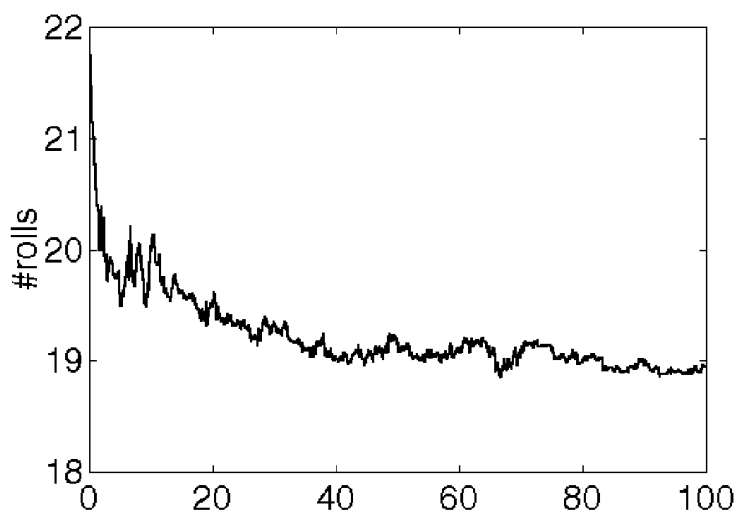
Third pass: increase challenges manually

- Can see negative trends 18k 25k
- To correct negative trends, we increase the difficulty to challenger:
- 10,000 generations: must win 5/6 games
- 70,000 generations: must win 7/8 games
- *After 100,000 generations we seem to have a strong player*

regression testing: 1k,10k,100k players



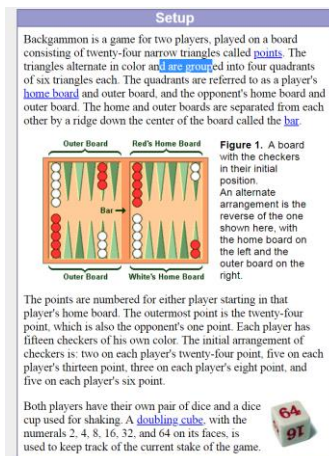
Improvement in Bearing Off



Why did HC Gammon work at all? Hillclimbing sucks.

- I needed a "toy" version of backgammon for experimentation

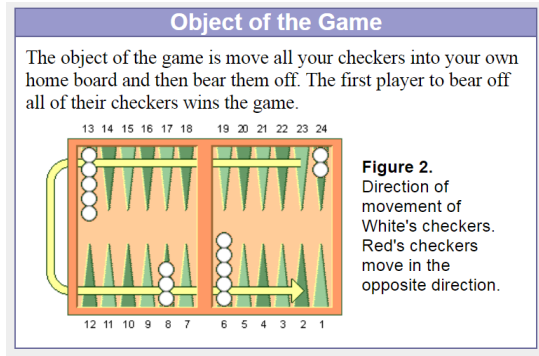
Setup



- Six positions
- Start on 0 1 2
- One dice per player

objective

- Same thing



Movement

- Opening roll: winner gets difference
- Only one checker per point
- Adjacency protects from hitting

Movement of the Checkers

To start the game, each player throws a single die. This determines both the player to go first and the numbers to be played. If equal numbers come up, then both players roll again until they roll different numbers. The player throwing the higher number now moves his checkers according to the numbers showing on both dice. After the first roll, the players throw two dice and alternate turns.

The roll of the dice indicates how many points, or pips, the player is to move his checkers. The checkers are always moved forward, to a lower-numbered point. The following rules apply:

1. A checker may be moved only to an open point, one that is not occupied by two or more opposing checkers.
2. The numbers on the two dice constitute separate moves. For example, if a player rolls 5 and 3, he may move one checker five spaces to an open point, or he may move the one checker a total of eight spaces to an open point, but only if the intermediate point (either three or five spaces from the starting point) is also open.
3. A player who rolls doubles plays the numbers shown on the dice twice. A roll of 6 and 6 means that the player has four sixes to use, and he may move any combination of checkers he feels appropriate to complete this requirement.
4. A player must use both numbers of a roll if this is legally possible (or all four numbers of a double). When only one number can be played, the player must play that number. Or if either number can be played but not both, the player must play the larger one. When neither number can be used, the player loses his turn. In the case of doubles, when all four numbers cannot be played, the player must play as many numbers as he can.

Figure 3. Two ways that White can play a roll of 13.

Hitting back, reentering

Hitting and Entering

A point occupied by a single checker of either color is called a **blot**. If an opposing checker lands on a blot, the blot is **hit** and placed on the **bar**.

Any time a player has one or more checkers on the bar, his first obligation is to **enter** those checker(s) into the opposing home board. A checker is entered by moving it to an open point corresponding to one of the numbers on the rolled dice.

For example, if a player rolls 4 and 6, he may enter a checker onto either the opponent's four point or six point, so long as the prospective point is not occupied by two or more of the opponent's checkers.

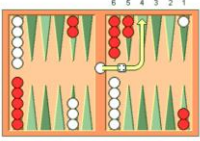


Figure 4. If White rolls 4-6 with a checker on the bar, he must enter the checker onto Red's four point since Red's six point is not open.

If neither of the points is open, the player loses his turn. If a player is able to enter some but not all of his checkers, he must enter as many as he can and then forfeit the remainder of his turn.

After the last of a player's checkers has been entered, any unused numbers on the dice must be played, by moving either the checker that was entered or a different checker.

- Same thing
- No need to have all checkers on board to escape

Bearing Off

Bearing Off

Once a player has moved all of his fifteen checkers into his home board, he may commence **bearing off**. A player bears off a checker by rolling a number that corresponds to the point on which the checker resides, and then removing that checker from the board. Thus, rolling a 6 permits the player to remove a checker from the six point.

If there is no checker on the point indicated by the roll, the player must make a legal move using a checker on a higher-numbered point. If there are no checkers on higher-numbered points, the player is permitted (and required) to remove a checker from the highest point on which one of his checkers resides. A player is under no obligation to bear off if he can make an otherwise legal move.


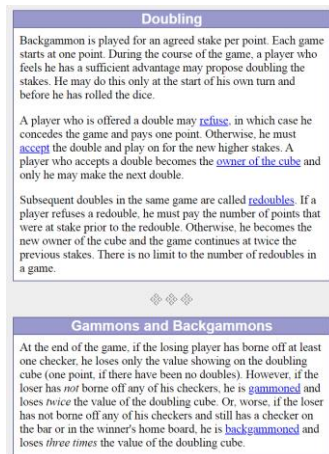


Figure 5. White rolls 4-6 and bears off two checkers.

A player must have all of his active checkers in his home board in order to bear off. If a checker is hit during the bear-off process, the player must bring that checker back to his home board before continuing to bear off. The first player to bear off all fifteen checkers wins the game.

- No need to get to home board
- Any checker can bear off with high-enough roll

Doubling (<http://bit.do/ubling>)

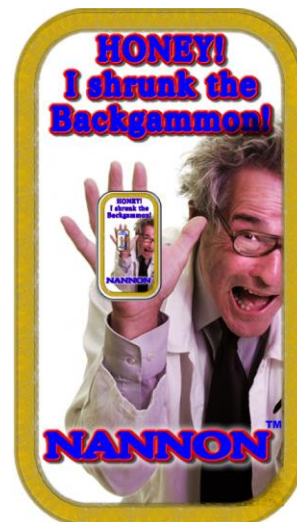


- Doubling works the same way, but I didn't want to teach it.
- "takes a minute to learn, a lifetime to master"
- Shutting out your opponent counts as a "Nannon!" and pays double (but I didn't want to teach it).

Nannon is a new family of games with a "complexity" knob

- N Number of Board spots
- K Number of checkers per side
- D sided die to roll
- To count the number of game states:

$$\sum_{i=0}^k \sum_{j=0}^k \binom{n}{i} \binom{n-i}{j} (k+1-i)(k+1-j)$$



Bellman 1957 (Part 1)

- 1) Every position in two player markov game like Backgammon has a payoff value:
- 2) Payoff correct when we both make optimal moves going forward
- 3) If I make less than optimal moves, the value goes down
- 4) If you make less than optimal moves, the value goes up.
- 5) QED: values in table are upper and lower bounded, thus convergent!

DEMO.CS.BRANDEIS.EDU

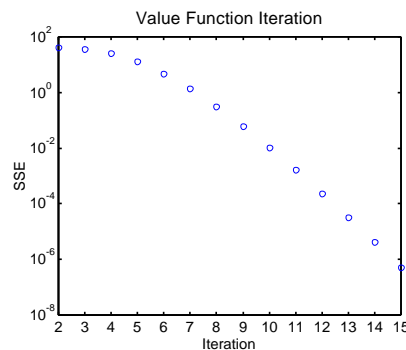
21

Bellman 1957 (Part 2)

- We can find the value updating all values with a 1 ply expectimax lookahead.
- QED in 1992 Tesauro stumbled upon Bellman's Value Iteration Convergence principle with his NN interpolating a "Value Function", leading to a renaissance.
- *The 1995 paper did not mention value iteration nor Richard Bellman!*

Nannon Falls under Bellman. So?

- Playing one-ply Expectimax with converged (solved) values leads to “superintelligent” player
 - No AI Rules, No AI Logic, No LISP necessary, just numbers.



Longer game favors intelligence “Against Nature” (random play)

board	chex	states	IQ Advantage
6	1	57	0.5
6	2	630	0.5932
6	3	2530	0.6732
10	3	31425	0.7404
12	3	86147	0.7766
8	4	31826	0.7773
10	4	137104	0.8313
12	4	517526	0.8644

- What is the advantage of intelligence in Nannon with k checkers?

- When k is 1, a game is all luck!
- When k > 1 there is choice
 - The longer the game, the more opportunity to apply intelligence
 - The better a High-IQ player does “Against Nature”