John Tromp and the Big Numbers of Go: The Possible Positions, Games and the Longest

By Peter Shotwell

Note: This is the "Final Solution" of the work of John Tromp and his colleagues whose progress was first reported on in 2010. After approximating the enormous number of possible Go games and the length of the longest one, they have finally reached their goal of precisely calculating the number of legal positions on the standard board size of 19x19.

Introduction: The State of Affairs Six Years Ago in 2010

Another mathematical fairyland in the world of Go was recently added to by Dutch computer scientist John Tromp and his colleagues. Their work on Go Combinatorics—the science of counting—join Elwyn Berlekamp and Bill Spight's combinatorial game theories and Monte Carlo methods in computer Go that have left little of our traditional thoughts about the inner mechanics of the game. [1]

In a private communication, Dr. Tromp discussed his early work on the longest possible game. He said that the number of possible games is exponential [2] in the length of the longest game, but nevertheless, I found the techniques and results to be remarkably interesting.

And what kind of game would this be?

The notion of eyes only arises in sensible play. For our results, forget all that you know about Go strategy and sensible play; we simply look at all possible move sequences . . .

The result is that most of the games are similar to the Stone Counting Method that I used in *Beginning Go* (Tuttle 2008) to explain the game and teach the concept of shape on small boards to children. Both players keep putting down stones until the loser has only two eyes left which leaves the winner with two eyes plus empty intersections that count as points. However, in Dr. Tromp's version the game doesn't end at that point—the "loser" can still play by filling in one eye and the game doesn't stop just because there are no stones left after everything is taken off by the opponent on the

[1] In 2016, Google's *Deepmind AlphaGo* that can beat top professionals has now been added to the list (http://deepmind.com/alpha-go.html) while a movie, *The Surrounding Game,* will soon explore the intensities of the also other-worldly training and development of professional Go players in the Far East. (http://www.surroundinggamemovie.com/en)

[2] See, for example, http://www.purplemath.com/modules/expofcns.htm

next move. They just keep playing until the positional Superko rule that no previous position can repeated forces two consecutive passes that end the game. The result is that the longest possible Go game has over 1^48 moves (10 followed by 48 zeroes). [3]

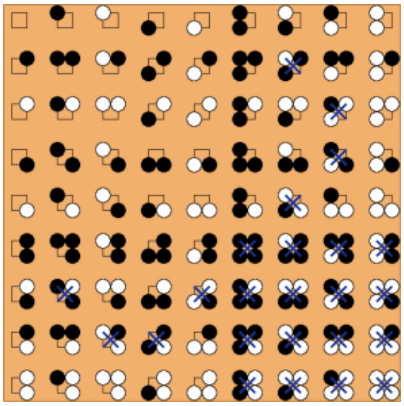
But who won?

At the end of the longest possible game, all moves lead to repetition of some earlier position. The board at this point may look like complete chaos however, with many groups in atari*. To ask who won is to miss the point. The scoring method is of no concern to counting games. You could even say the winner is the one who made the fewest passes; it doesn't change the number of possible move sequences. So we don't even specify any scoring method. It is of no concern to us.*

However, this game might run into time limits. According to one theory—the Big Freeze—if gravity fails to hold the universe together, there will only be another 1000 trillion years before all matter disappears. After the stars run out of fuel, all protons will plunge into Black Holes and evaporate, and all that will be left are electrons, positrons, and neutrinos floating aimlessly around, ready for the next stage, whatever that might be. With about 32,000,000 seconds in a year, the players would have to move very, very fast, since there would just be thirty-one sextillion seconds (31 followed by only 21 zeros) left to play in.

At the other end of the scale, since two consecutive passes at any point ends the games, the shortest possible one is 0 moves. Because no resignations are permitted, Black passes and then White passes. The next shortest game is Black plays, White passes, Black passes. Then the idea that there are 361 possible positions on a 19x19 board in this 1-move game leads into the question of the number of possible legal Go positions on other-sized boards.

On a 2x2 board, 57 out of 81 positions or about 70% are legal.



The illegal positions are X'd out

[3] This still conforms with the Logical Tromp-Taylor rules http://senseis.xmp.net/?TrompTaylorRules

However, as board-size increases, the percentage of legal positions naturally decreases with the result that on a 19x19 board, the figure would close to be about 1% positions. This then brings up the question of how many that would be and how this would compare to Chess with its 8x8 board.

What we do know is that the number of Go positions is many, many, many orders of magnitude larger, a 171-digit number versus a roughly 46-digit number for Chess. This dramatic comparison can only show so many orders of magnitude in difference. Even comparing the size of the universe with the size of the nucleus of an atom, that's only a factor of 10^41, or 41 orders of magnitude. As you can see, the 125 magnitude difference that Go positions have over Chess positions dwarfs that. However, the number for Chess is necessarily inexact because deciding the legality of a single Chess position can be arbitrarily hard and is the subject of so-called retrograde analysis. These are active areas of research for me and several friends that is restricted by the enormity of the computer power that is needed, so it may take a number of years.

I asked about the oft-quoted Go adage that, "There are more possible Go games than there are atoms in the universe." [4]

*Actually, that statement is only about the number of legal Chess games and it vastly understates the situation. On a 2x2 Go board, despite the fact that there are only 57 legal positions, there are 386,356,909,593 possible games! For the number of possible 19x19 games, we must resort to double exponentiation. We have found that on 19x19 boards, the lower bound for that number is 10^(10^48) , while the upper bound is a whopping 10^(10^171). As for positions, there are 3*361x0.011957528698 x 10^170 possible positions that can appear on 19x19 boards while the number of atoms in the universe is only 10^80.* [In fact, on a 13x13 board, the number of possible positions is about the number of atoms in the universe, while the approximate number on a 19x19 board is more than 10^{90} times that figure.] *So now we are up to exact figures for the number of possible positions on 17x17 but it's not final until I can do the exact computation for 19x19. :-)*

When it is finished, will this work have any applications or uses?

None that I can think of . . .

[4] http://senseis.xmp.net/?NumberOfPossibleGoGames

Jan. 20, 2016: After 10 Years Enough Memory, Time, and Diskspace is Arranged so the Legal Number of Positions on 19x19 is Finally Reached

Each added line from 17x17 to 19x19 needed a five-fold increase in computer power, so on Jan 20, 2016 the number of legal positions called “L19” was at last determined to be 208168199381979984699478633344862770 286522453884530548425639456820927419612738015378525648451698 519643 907259916015628128546089888314427129715319317557736620397247064840935. Or, in a more compact form for the 171 digits:

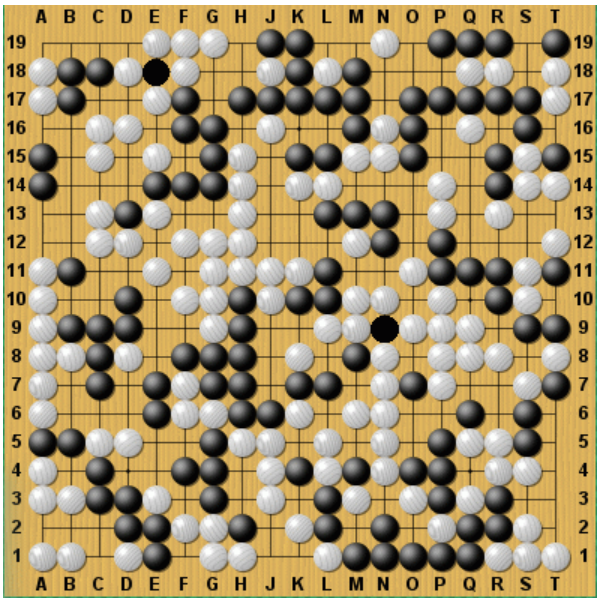
208168199381979984
699478633344862770
286522453884530548
425639456820927419
612738015378525648
451698519643907259
916015628128546089
888314427129715319
317557736620397247
064840935

This can be written more naturally in ternary. [5]

0 0 0 0 2 2 2 0 1 1 0 0 2 0 1 1 1 0 1
2 1 1 2 1 2 0 0 2 1 2 1 0 0 0 2 2 0 2
2 1 0 0 2 1 0 1 1 1 1 1 0 1 1 1 1 1 2
0 0 2 2 0 1 1 0 2 0 0 1 2 1 0 2 0 1 0
1 0 2 0 2 0 1 2 0 1 1 2 2 1 0 0 1 2 1
1 0 0 0 1 1 1 2 0 2 2 0 0 0 2 0 1 2 2
0 0 2 1 2 0 0 2 0 0 1 1 1 0 2 0 2 0 0
0 0 2 2 0 2 2 2 0 0 0 2 1 0 1 0 0 0 2
2 1 0 0 2 0 2 2 2 2 1 0 0 2 1 1 1 2 1
2 0 0 1 0 2 2 1 2 1 1 2 2 0 2 0 1 2 0
2 1 1 1 0 0 2 1 0 0 2 2 1 2 2 2 0 1 1
2 2 1 2 0 1 1 1 0 2 0 1 2 0 2 2 2 0 2
2 0 1 0 1 2 1 1 0 1 1 0 2 1 2 0 0 2 1
2 0 0 0 1 2 2 1 1 2 0 2 2 0 0 1 0 1 0
1 1 2 2 0 0 1 2 2 0 2 0 2 0 1 2 2 1 0
2 0 1 0 0 1 1 0 2 1 2 1 2 1 1 0 2 2 0
2 2 1 1 2 0 1 0 2 0 1 2 0 2 1 2 1 0 0
0 0 0 1 1 2 2 1 0 2 1 0 1 0 2 1 1 2 0
2 2 0 2 1 0 2 2 0 0 2 1 1 1 1 1 2 2 2

[5] In decimal notation every position is weighted by a power of 10, binary uses powers of 2, and ternary uses powers of 3. For example, 201 denotes 2x9+0*3+1*1=19.

This is not only because the numbers used are simpler (0,1 and 2), but also because each point on the board can have three stages. Thus, for fun, by transposing the 0s into empty intersections, the 1s into black stones and the 2s into white stones, Dr. Tromp made a Go position out the possible number and noted that, not surprisingly, it would be an illegal one.



Black stones at E18 and N9 lack "liberties," making the position illegal which is no surprise since the probability of any position being legal is about 0.0000222 in ternary, close to 3^-4, or 1 in 81. [6] *This 1.2% chance had already been computed by random sampling back in 1992 and the better approximation L19* ~ 2.081681994 10^170 *has been known since 2006. But it took 10 years of asking universities and big companies like Google and Amazon to get enough computer availability to nail it down to the last digit.*

A Computational Challenge is Helped by the Chinese Remainder Theory

It is fun and customary to factor interesting numbers to see if they might be even more interesting. So what are the prime factors of L19 and how would we find them? Let's start with the latter questions. The dumbest possible approach would have been repeated trial division. But this takes time exponential in the number of digits, making it completely infeasible for numbers of more than, say, 10 digits. Even allowing ourselves a million trillion steps, this would get us no further than the number 10^18, far short of L19. Instead, we needed to use an advanced algorithm like the General

[6] That is, 0.0000222 is almost 0.0001 in ternary, which is 3^{-4} (the 1 is at position -4, so weighted by that power of 3), 3^{-4} = 1/3^4 = 1/81.

Number Field Sieve [7] or the Elliptic Curve Method (ECM). [8] These still needed exponential time—but not in the number of digits itself, but rather in the cube root or square root thereof, thereby vastly extending the range of inputs on which they're feasible in practice. [9]

Using an ECM implementation [10] courtesy of Dario Alejandro Alpern, I was able to factorize L19 in mere hours on my laptop. This yielded eight unique prime factors that when multiplied together gave L19 in 1,3,7,9,19,34,35, and 65 digits respectively:

*5 **
*401 **
*4821637 **
*964261621 **
*2824211368611548437 **
*219846696500237600175961330792275 7 **
*659486468368075679414404343174041 97 **
*545365406033465952117220614213780728204593769853147073453174700 47 **

The challenge of constructing rather than deconstructing L19 is surprisingly similar. Individually testing 3^361 positions for legality is as insane as doing trial divisions (while detecting illegalities early on during position generation is easy with this C source code, [11] it offers only slight improvements). Just as with factoring, we needed an advanced algorithm that was exponential not in the number of points, but in the square root of the number of board points which is its width on a square board.

Gunnar Farnebäck and I developed just such an algorithm in the early 2000s. As described in detail in our 2016 paper "Combinatorics of Go," it essentially reduces computing L19 to taking the 361st power of a very sparse matrix of 363 billion rows and columns [12] but the computational power required for this only became available to me last year.

The matrix rows and columns represent so-called border states that describe all pertinent information about a cross section of the board, such as the presence of stones still needing liberties and how they are connected. Matrix powers represent counts of the number of paths from one border state to another. Each legal position uniquely corresponds to a path from the edge border state to one without stones with no liberties, as illustrated for this position on a 3x3 board:

---

[7] https://en.wikipedia.org/wiki/Lenstra_elliptic_curve_factorization

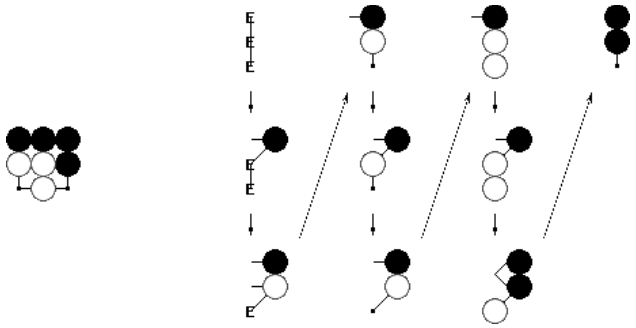
[8] https://en.wikipedia.org/wiki/General_number_field_sieve

[9] If "feasible" computations are, for example, at most a million trillion steps (10^18), then the largest number that can be handled with an algorithm that is exponential in the number of digits is just 10^18. However, an algorithm that is exponential in the square-root of the number of digits can handle numbers as large as 10^324.

[10] From "Get the source code" at https://www.alpertron.com.ar/ECM.HTM

[11] See, for example, http://tromp.github.io/go/approx.c

[12] John Tromp & Gunnar Farnebäck; "Combinatorics of Go"; *Lecture Notes in Computer Science* Vol. 4630; Springer Books; 2016 http://tromp.github.io/go/gostate.pdf



Clever use of the Chinese Remainder Theorem [13] allowed us to split the computation into 9 independent parts, each computing L19 modulo 2^64 minus some small number, contributing 64 bits of the 566 bit result. [14]

We started on March 6, 2015 by running on big servers at three places:

- *The IAS School of Natural Sciences in Princeton, where the L18 computation was recently performed, generously provided by Piet Hut and administered by Lee Colbert*
- *The IDA Center for Communications Research, on the other side of Princeton, generously offered by Michael Di Domenico, who spent many hours setting up, monitoring, streamlining, and patching the jobs, as well as helping improve the job flow scripts*
- *The HP Helion Cloud, generously provided by HP courtesy of Eric Moore*

Suffering many hiccups and a few catastrophes, after generating an estimated 30 petabytes [15] of disk IO (Input-Output), we now had the full 171 digits of precision and finished on December 26, 2015. It was a nice Day-After-Christmas event! It also took a few weeks into the New Year for me to get the log files, and so, it was a huge "Holiday Present" from all these people who helped make the computations possible!

Many thanks also to Gunnar and Michal Koucký [16] for their contributions. Gunnar wrote a legal counting program in pike, [17] while Michal suggested the use of Chinese Remaindering, implemented a file-based program, and was a partner in obtaining the results for n=14,15,16 and 17.

Thanks so much to all of you!

[13] https://en.wikipedia.org/wiki/Chinese_remainder_theorem
[14] http://stackoverflow.com/questions/2664301/how-does-modulus-divison-work
[15] https://en.wikipedia.org/wiki/Petabyte
[16] http://iuuk.mff.cuni.cz/~koucky/
[17] http://www.lysator.liu.se/~gunnar/legal.pike.txt

The Number of Legal 19 X n Go Positions in the On-Line Encyclopedia of Integer Sequences #AO94777

1. 1

2. 7

3. 12675

4. 24318165

5. 414295148741

6. 62567386502084877

7. 83677847847984287628595

8. 990966953618170260281935463385

9. 10391914879129383431898309043879879346 9

10. 964984285019096545896308879788350980881481778 57

11. 79347486681658226682093667179018913232167338311218515189 9

12. 57774258489513238998237970307483999327287210756991189655942651331169

13. 37249792307686396442294904767024517674249157948208717533254799550970595875237705

14. 12667732900366224249789357650440598098805861083269127196623872213228196352455447575029701325

15. 07514643083613831187684137548661238097337888203278444027646016628708836017112983093392398689983378015094 91

16. 81306696382275541642905602248429964648687410096724926394471959997560745985050222203959114933143180552465546745306704237 7

17. 90793889196281992046057261818504652201510583381479222439672692319440591872147679971059923417352092306672884621790900736597125832620874 37

18. 9723114288829212892740188841706543509937780640178732810318337696945624428547218105214326012774371397184848890970111836283470468812827907149926502347633

19. 0816819938197998469947863334486277028652245388453054842563945682092741961273801537852564845169851964390725991601562812854608988831442712971531931755773662039724706484093 5

The entire table of the Legal Counts for 1-20 x n and for some rectangular board sizes can be found on Dr. Tromp's website. [18]

A Googolplex of Possible 19x19 Games

In addition, the lower bounds of the longest game were refined by Dr. Tromp and Mathieu Walaraet as well as the number of possible 19x19 games. [19] This is now over a googolplex, [20] a number that cannot be written with ordinary numbers because it would take more space than is available in the known universe. [21]

What is Next for Dr. Tromp?

A writer for Vice's *Motherboard* e-Magazine asked Dr. Tromp this question. [22]

. . . So after solving a mystery that's been unsolved [in the more than 2,000 years since the ancient Chinese first thought about it]*, what's next? Tromp tells me he plans to continue work on his "Cuckoo Cycle" [23] proof-of-work system and solve large-scale Connect Four [24] problems, but he's especially interested in improving his similar work on chess. [25] Chess, though, is a different beast.*

"The situation is very different with chess, in that we'll never know the exact number," he says. "Right now we don't even know the order of magnitude. I want to get to the point where we can at least say: the number is between 10^44 and 10^45."

---

[18] http://tromp.github.io/go/legal.html#table.

[19] http://matthieuw.github.io/go-games-number/GoGamesNumber.pdf and Dr. Thorp's paper.

[20] The term was invented by a 9-year nephew of a mathematician. When asked what he thought such a "silly" number as 10^{100} (1 followed by 100 zeros) should be called, after a short pause, he replied that is was a "Googol." Thus, later on, a "Googolplex" became 10 to the power of a googol (1 followed by 10 to the power of 100 zeros). As for "writing" it on a computer, since processor power doubles about every 1 to 2 years, any attempt would be overtaken by faster and faster processors which would go on for hundreds of years before they finally caught up. Looking for a domain name for their new company, one of Larry Page and Sergey Brin's friends mistyped "Google" which was available, so corporate headquarters came to be called the GooglePlex.

[21] https://en.wikipedia.org/wiki/Googolplex#In_the_physical_universe

[22] http://motherboard.vice.com/author/LeifJohnson?Article_page=6

[23] https://github.com/tromp/cuckoo

[24] https://en.wikipedia.org/wiki/Connect_Four

[25] http://tromp.github.io/chess/chess.html

Which leaves the question of why?

"Having the ability to determine the state complexity of the greatest abstract board game, and not doing it, that just doesn't sit right with me," Tromp says. "Or, as the great mathematician David Hilbert [26] proclaimed, 'We must know—we will know!'"

Verifiability of the Go Figures for Do-It-Yourselfers

The software used for these computations is available at Dr. Tromp's github repository. [27] Running "make" should compute L(3,3) in about a second. For running an L19 job, a beefy server with 15TB of fast scratch disk space, 8 to 16 cores, and 192GB of RAM is recommended. Expect a few months of running time. Please use a modulus index outside the set {0,1,2,3,4,5,6,11,19} that we used. The computation has many built-in checks to guard against memory and disk corruption. Values L(m,n) where n<m, are checked for matching L(n,m) (computed in a very different way), while all values L(m,n) have to closely match the approximation formula L(m,n) ~ 0.8506399258457 * 0.965535059338374^{m+n} * 2.97573419204335 72493^{m*n} that is derived from earlier results. Finally, application of the Chinese Remainder Theorem provides an important safeguard in that a tiny change in any of the inputs results in a huge change of output.

A Good Server Benchmark?

In 2010, Dr. Tromp did not think there would be any uses for his work and interests but now that the summit has been climbed, there indeed seems to be several.

For example, Go counting would make a decent server benchmark:

- The task is well defined, easily understood, and non-artificial.
- The program code is small and self-contained.
- The generated data sets are huge.
- The problem is a typical instance of map-reduce, and thus representative of a wide class of popular problems.
- The computation requires a good balance of multi-core processing power, memory for sorting, and disk-IO.
- The board size parameter gives an entire family of benchmarks, where each increment corresponds to a factor of about 5 in effort.

[26] https://en.wikipedia.org/wiki/David_Hilbert

[27] https://github.com/tromp/golegal