The number of legal Go positions

John Tromp
http://tromp.github.io/

65 year annivesary Go Club Amsterdam, Bergen aan Zee 2023

Overview

- Numbers
- Positions
- Legal
- Why
- History
- Algorithm
- Verifiability
- Concluding

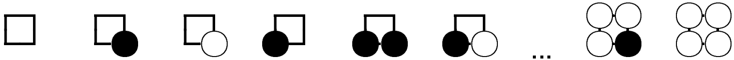
Numbers

- positional notation, a.k.a. place-value, or base-*b* notation
- decimal (base 10) notation uses the 10 digits 0,1,2,3,4,5,6,7,8,9
  $4125 = 4125_{10} = 4 \cdot 1000 + 1 \cdot 100 + 2 \cdot 10 + 5 \cdot 1$
  $= 4 \cdot 10^3 + 1 \cdot 10^2 + 2 \cdot 10^1 + 5 \cdot 10^0$
- binary (base 2) notation uses the 2 binary digits 0,1
  $1010_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0$
  $= 1 \cdot 8 + 0 \cdot 4 + 1 \cdot 2 + 0 \cdot 1$
  $= 8 + 2 = 10$
- ternary (base 3) notation uses the 3 ternary digits 0,1,2
  $2010_3 = 2 \cdot 3^3 + 0 \cdot 3^2 + 1 \cdot 3^1 + 0 \cdot 3^0$
  $= 2 \cdot 27 + 0 \cdot 9 + 1 \cdot 3 + 0 \cdot 1$
  $= 54 + 3 = 57$
- Go gives a new meaning to "positional notation"

Go positions

- Go-ternary notation uses the symbols $+$, ●, ○ instead of 0,1,2
- Go positions denote first $3^{n \times n}$ numbers

0	1	2	3	4	5	...	79	80
$\begin{matrix}0\cdot0\\0\cdot0\end{matrix}$	$\begin{matrix}0\cdot0\\0\cdot1\end{matrix}$	$\begin{matrix}0\cdot0\\0\cdot2\end{matrix}$	$\begin{matrix}0\cdot0\\1\cdot0\end{matrix}$	$\begin{matrix}0\cdot0\\1\cdot1\end{matrix}$	$\begin{matrix}0\cdot0\\1\cdot2\end{matrix}$	...	$\begin{matrix}2\cdot2\\2\cdot1\end{matrix}$	$\begin{matrix}2\cdot2\\2\cdot2\end{matrix}$



Legal positions

Definition

A position is *legal* iff all strings have liberties

Theorem

A position is legal iff it's reachable in a game

Proof.

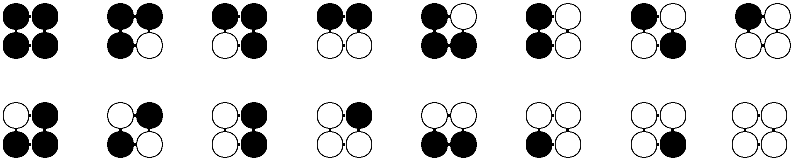
$\Rightarrow$ have White pass while placing black stones, then have Black pass while placing white stones

$\Leftarrow$ game rules ensure strings have liberties

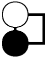
□

Illegal and legal 2×2

- 16 illegal 4-stone positions

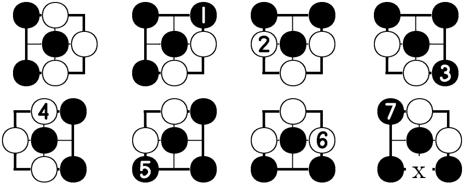


- 8 illegal 3-stone positions

- all positions with at most 2 stones are legal
- leaving $81 - 24 = 57$ legal 2×2 positions
- L2 =  in Go-ternary

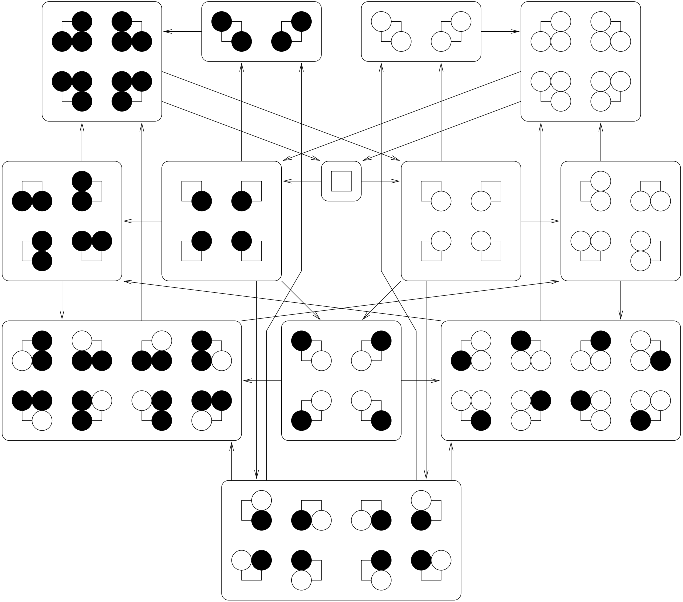
Symmetry

- Rotations and reflections generally produce *different* positions
- Identity of positions matters in *superko*
- Pinwheel ko



- Superko prevents White 8 at 'x' but not White 2

386,356,909,593 simple paths



Why?

- "Because it's there" – George Mallory
- "more than the number of atoms in the universe" is a horrible understatement
- interesting computational challenge
- major open problem left in "Combinatorics of Go" by myself and Gunnar Farnebäck from CG2006 http://tromp.github.io/go/gostate.pdf

Why?

- "Because it's there" – George Mallory
- "more than the number of atoms in the universe" is a horrible understatement
- interesting computational challenge
- major open problem left in "Combinatorics of Go" by myself and Gunnar Farnebäck from CG2006 `http://tromp.github.io/go/gostate.pdf`

Why?

- "Because it's there" – George Mallory
- "more than the number of atoms in the universe" is a horrible understatement
- interesting computational challenge
- major open problem left in "Combinatorics of Go" by myself and Gunnar Farnebäck from CG2006 `http://tromp.github.io/go/gostate.pdf`

Why?

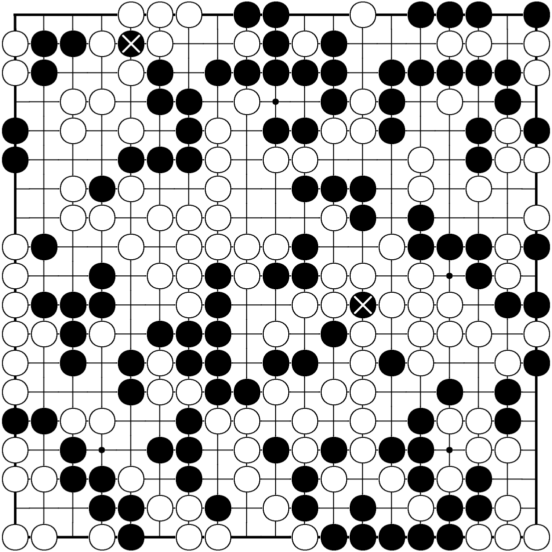
- "Because it's there" – George Mallory
- "more than the number of atoms in the universe" is a horrible understatement
- interesting computational challenge
- major open problem left in "Combinatorics of Go" by myself and Gunnar Farnebäck from CG2006 `http://tromp.github.io/go/gostate.pdf`

Why?

- "Because it's there" – George Mallory
- "more than the number of atoms in the universe" is a horrible understatement
- interesting computational challenge
- major open problem left in "Combinatorics of Go" by myself and Gunnar Farnebäck from CG2006 `http://tromp.github.io/go/gostate.pdf`

The number of legal 19x19 positions is ...

L19 in Go-ternary



L19 in decimal

208168199381979984699478633344862770286522453884530548425
639456820927419612738015378525648451698519643907259916015
628128546089888314427129715319317557736620397247064840935

confirming 2006 estimate of $2.081681994 \cdot 10^{170}$

History

1992 Achim Flammenkamp estimates L19 at 1.2% of 3^{361}

1994 Jonathan Cano computes up to L(4,5) by brute force enumeration

2000 Les Fables describes a dynamic programming method

2005 Gunnar Farnebäck computes up to L(10,10)

2005 Tromp computes up to L(13,13) with optimized implementation

2006 Tromp+Koucky compute up to L(17,17) on CWI cluster

2014 Piet Hut at Princeton Institute for Advanced Studies (IAS)

2015 Tromp computes L(18,18); requests more computing power

2015 Michael Di Domenico at Princeton Institute for Defense Analysis (IDA)

Chinese Remainder Theorem

- Given some relatively prime numbers $m_1, \ldots, m_k$
- any number $n < \Pi m_i$ is uniquely determined by its remainders $n \bmod m_1, \ldots, n \bmod m_k$
- Example with $m_1 = 5, m_2 = 8$

	0	1	2	3	4	5	6	7
0	0	25	10	35	20	5	30	15
1	16	1	26	11	36	21	6	31
2	32	17	2	27	12	37	22	7
3	8	33	18	3	28	13	38	23
4	24	9	34	19	4	29	14	39

- error detection

Independent jobs

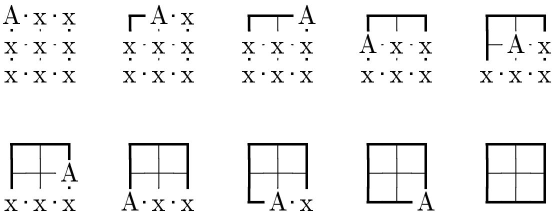
- for d in $\{0, 3, 5, 7, 9, 11, 15, 45, 83\}$
- compute L19 modulo $2^{64} - d$

where	d	$2^{64} - d$	L19 $\bmod (2^{64} - d)$
IAS	0	18446744073709551616	8090796072333351655
IDA	3	18446744073709551613	2915461546443917794
IDA	5	18446744073709551611	7586469474294957788
IDA	7	18446744073709551609	6473614947737753186
IDA	9	18446744073709551607	10169697560205166237
IDA	11	18446744073709551605	8330618849129880355
IDA	15	18446744073709551601	15770133769769565723
IAS	45	18446744073709551571	18086767044943672066
IAS	83	18446744073709551533	4954386835027564217

- 9×64 bits $= 576$ bits, enough for 566 bit answer
- compute L19 using Chinese Remainder Theorem

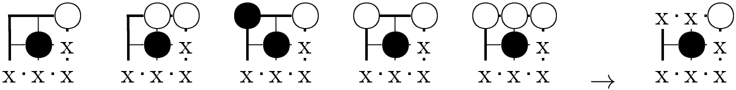
Count on partial boards

- partial board of i points has 3^i partial positions
- one count for all illegal partial positions
- other counts for legal partial positions
- extend partial positions one point at a time

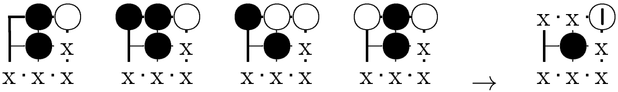


Abstract from irrelevant details

- Only the last *n* points of the partial position matter



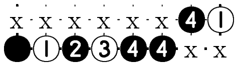
- Distinguish stones with and without liberties



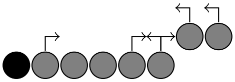
- Identify connections between libertyless stones

Efficient border state encoding

encode



as

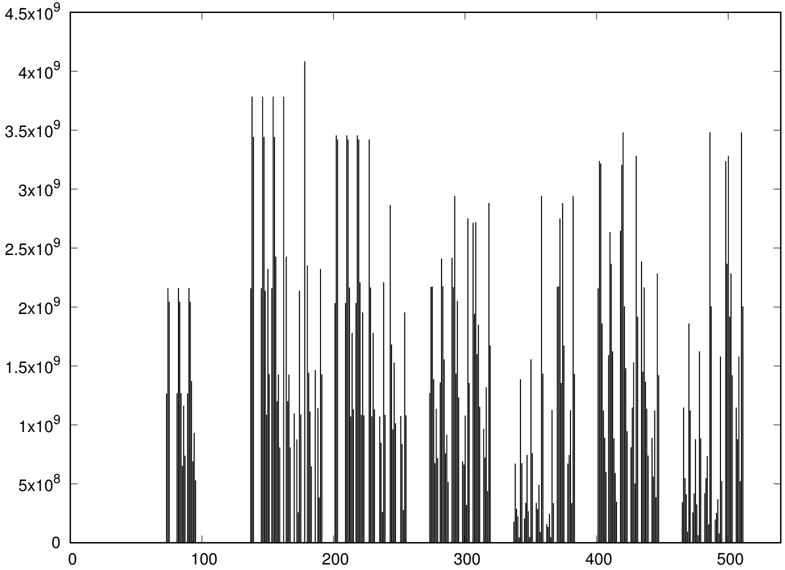


- infer string connections from left/right flags
- infer color of gray liberty less stones
- also exploit color symmetry (swapping all black/white)
- uses only 3 bits per point; 57 bits for L19

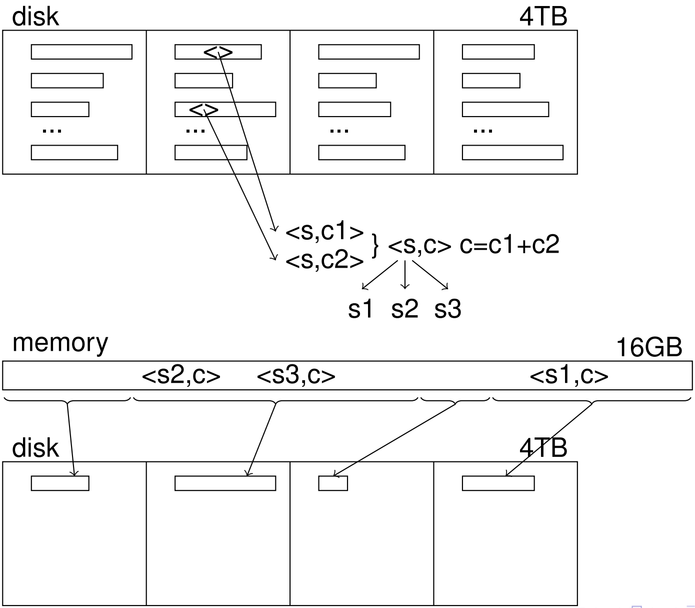
Huge datasets

- L19 has over 363 billion border states
- 16 bytes per <state,count> pair
- set of partial board counts takes 5.8TB of pairs
- ordered pairs allow delta state encoding
- redundancy unavoidable
- over 4TB per dataset
- partitioning allows for parallel processing

State density in code space



Extending all states



Verifiability

- Software available at `https://github.com/tromp/golegal`
- Use beefy server for additional congruences
  - 15TB of fast scratch diskspace
  - 8 to 16 cores
  - 192GB of RAM
  - a few months of running time
- checksums on all files
- check summing to 3^i
- check that $L(19, n) = L(n, 19)$ computed earlier
- check against estimate $2.081681994 \cdot 10^{170}$ extrapolated from earlier results

Approximation formula

- $L(m,n) \approx \alpha\beta^{m+n}L^{mn}$ for some constants α, β, and L.
- $$L = \lim_{n\to\infty} \frac{L(n,n)L(n+1,n+1)}{L(n,n+1)^2}$$
- $$B = \lim_{n\to\infty} \frac{L(n,n+1)}{L(n,n)L^n} = \lim_{n\to\infty} \frac{L(n,n)}{L(n,n-1)L^n}$$
- $$A = \lim_{n\to\infty} \frac{L(n,n)}{B^{2n}L^{n^2}}$$

Base of Liberties

n	$L(n,n)L(n+1,n+1)/L(n,n+1)^2$
3	2.979
4	2.9756
5	2.975732
6	2.9757343
7	2.9757341927
8	2.9757341918
9	2.975734192044
10	2.975734192044
11	2.97573419204350
12	2.97573419204355
13	2.9757341920435727
14	2.97573419204357255
15	2.975734192043357249 32
16	2.9757341920433572493 62
17	2.97573419204335724938 11
18	2.975734192043357249380 97

Concluding Remarks

- Dynamic Programming reduces an problem exponential in n^2 (impossible) to a problem exponential in n (feasible). For factoring numbers as big as L19, similar improvements are possible over trial division.
- answered
  Ultimate question of liberties, the universe, and everystring
- newly computed $L(19, 19)$, $L(19, 18)$, and $L(18, 18)$ improve accuracy in approximation formula
  $L(m, n) \equiv 2.9757341920433572493 81^{mn} \times 0.96553505933837387^{m+n} \times 0.8506399258457145$
- Go counting could make nice server benchmark
- The king of games versus the game of kings

Concluding Remarks

- Dynamic Programming reduces an problem exponential in n^2 (impossible) to a problem exponential in n (feasible). For factoring numbers as big as L19, similar improvements are possible over trial division.
- answered
  Ultimate question of liberties, the universe, and everystring
- newly computed $L(19,19)$, $L(19,18)$, and $L(18,18)$ improve accuracy in approximation formula $L(m,n) \equiv 2.9757341920433572493^{mn} \times 0.96553505933837387^{m+n} \times 0.8506399258457145$
- Go counting could make nice server benchmark
- The king of games versus the game of kings

Concluding Remarks

- Dynamic Programming reduces an problem exponential in n^2 (impossible) to a problem exponential in n (feasible). For factoring numbers as big as L19, similar improvements are possible over trial division.
- answered
  Ultimate question of liberties, the universe, and everystring
- newly computed $L(19, 19)$, $L(19, 18)$, and $L(18, 18)$ improve accuracy in approximation formula
  $L(m, n) \equiv 2.975734192043357249381^{mn} \times 0.965535059338373 87^{m+n} \times 0.8506399258457145$
- Go counting could make nice server benchmark
- The king of games versus the game of kings

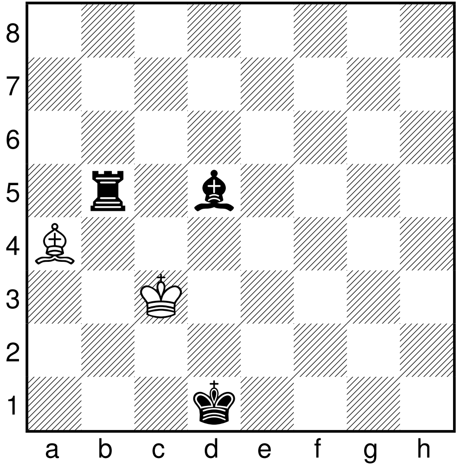
Concluding Remarks

- Dynamic Programming reduces an problem exponential in n^2 (impossible) to a problem exponential in n (feasible). For factoring numbers as big as L19, similar improvements are possible over trial division.
- answered
  Ultimate question of liberties, the universe, and everystring
- newly computed $L(19, 19)$, $L(19, 18)$, and $L(18, 18)$ improve accuracy in approximation formula
  $L(m, n) \equiv 2.9757341920433572493\,81^{mn} \times 0.96553505933837387^{m+n} \times 0.850639925845714 5$
- Go counting could make nice server benchmark
- The king of games versus the game of kings

Concluding Remarks

- Dynamic Programming reduces an problem exponential in n^2 (impossible) to a problem exponential in n (feasible). For factoring numbers as big as L19, similar improvements are possible over trial division.
- answered
  Ultimate question of liberties, the universe, and everystring
- newly computed $L(19,19)$, $L(19,18)$, and $L(18,18)$ improve accuracy in approximation formula
  $L(m,n) \equiv 2.9757341920433572493\,81^{mn} \times$
  $0.96553505933837387^{m+n} \times 0.850639925845 7145$
- Go counting could make nice server benchmark
- The king of games versus the game of kings

Chess counting



Server benchmark

- Task is well defined, easily understood, and non-artificial
- Program code is small and self-contained
- Generated data sets are huge
- Problem is a typical instance of map-reduce, and thus representative of a wide class of popular problems
- Computation requires a good balance of multi-core processing power, memory for sorting, and disk-IO
- Board size parameter gives family of benchmarks, in 5x effort increments