There are $7 \times 10^{26\,018\,276}$ self-avoiding walks of 38 797 311 steps on $\mathbb{Z}^3$

Nathan Clisby
MASCOS, The University of Melbourne

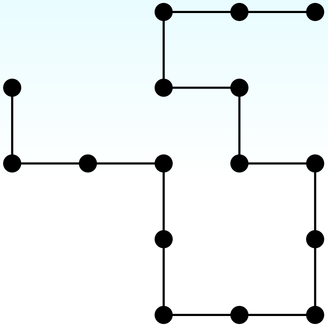
Institut für Theoretische Physik
Universität Leipzig
November 9, 2012

Outline

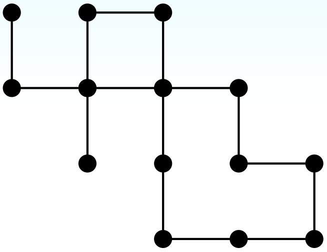
- Describe self-contained experiment to calculate connective constant μ for and count number of self-avoiding walks.
- Self-avoiding walks
- Global move (pivot)
- Efficient data structure (SAW-tree)
- Clever choice of observable
- Minimizing statistical error
- Results and conclusion

Self-avoiding walk model

- A walk on a lattice, step to neighbouring site provided it has not already been visited.
- Models polymers in good solvent limit.
- Exactly captures universal properties such as critical exponents.
- N-step SAW on $\mathbb{Z}^d$ is a mapping $\omega : \{0, 1, \ldots, N\} \to \mathbb{Z}^d$ with $|\omega(i+1) - \omega(i)| = 1$ for each i ($|x|$ denotes the Euclidean norm of x), and with $\omega(i) \neq \omega(j)$ for all $i \neq j$.
- For uniqueness, choose $\omega(0) = 0$.



SAW



Not a SAW

Asymptotic behaviour

- The number of SAW of length N, c_N, tells us about how many conformations are available to SAW of a particular length:

$$c_N \sim A\ N^{\gamma-1}\mu^N\left[1+\mathrm{corrections}\right]$$

- For $\mathbb{Z}^2$, $c_N = 1, 4, 12, 36, 100, 284, 780, 2172, \cdots$
- For $\mathbb{Z}^3$, $c_N = 1, 6, 30, 150, 726, 3534, \cdots$
- γ is a *universal* exponent.
- μ is the connective constant; lattice dependent.

- Direct enumeration does not get far, $c_N \sim \mu^N$ with $\mu \approx 2.64$ for $\mathbb{Z}^2$ and $\mu \approx 4.68$ for $\mathbb{Z}^3$.
- So, transform problem and count something else.
- For 2d lattices: finite lattice method extremely powerful. Count boundary states instead of walks, $O(1.3^n)$ (unfortunately, still exponential). Recently, Iwan Jensen found $c_{79} = 10194710293557466193787900071923676$ for $\mathbb{Z}^2$!
- For 3d lattices: most powerful method "length-doubling algorithm", combines brute force enumeration with inclusion-exclusion. $O(\mu^n) \to O((\sqrt{2\mu})^n)$[1].
- $c_{36} = 2941370856334701726560670$ for $\mathbb{Z}^3$.
- Series analysis used to extract information about asymptotic behaviour.

[1] R. Schram et al., J. Stat. Mech.: Theor. Exp., P060109 (2011)

- Wish to estimate c_N beyond limits accessible to exact enumeration.
- Obvious approach: simple sampling. Generate a simple random walk of length N, calculate probability that RW is self-avoiding. Probability $= c_N/(2d)^N \approx 4.68^N/6^N$ for $\mathbb{Z}^3$.
- Can improve slightly: forbid immediate reversals in the walk. Probability $= c_n/2d/(2d-1)^{N-1} \approx 4.68^N/6/5^{N-1}$ for $\mathbb{Z}^3$.
- For $N = 100$ only 1 in 1000 random walks with no immediate reversals is a SAW. Cannot push this much further.

- Rosenbluth sampling: only choose free edges.
- This introduces bias: compact walks which have few choices available are preferred.
- Correct bias by weighting walks.
- Weights provide an estimator of c_N, $c_N = \langle W_N \rangle$.
- Two issues:
  - High variance (poor estimator of c_N)
  - Attrition still occurs, since walks can become trapped. Can't sample truly long walks (ok up to N of the order of hundreds).

- PERM: Pruned Enriched Rosenbluth Sampling, a variant of sequential importance sampling.
- Prune: low weight walks, either discard with $P = 0.5$ or double weight.
- Enrich: high weight walks, make copies, ensure total weight remains the same.
- PERM: sensible choices for enrichment ensure attrition is eliminated, variance reduced.
- Dramatically better than Rosenbluth sampling, arbitrarily large N achievable.
- Sophisticated choices for pruning and enrichment algorithms can reduce correlations and variance.

Factors limiting the efficiency of PERM.

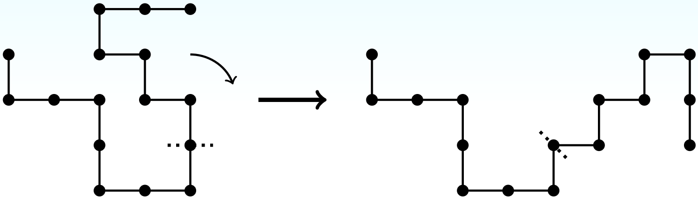
- Correlations introduced by enrichment.
- Variance of sample is reduced, but not eliminated. (In practice, variance can be essentially eliminated, at the expense of stronger correlation.)
- Intrinsic limit: CPU time $O(N)$ to produce a single walk. (Prohibitive for truly large N).
- Will now describe a method that overcomes each of these deficiencies.

To calculate c_N and μ efficiently we need to

- Utilise most efficient sampling method, rapidly move around state space.
- Utilise efficient data structures.
- Find a suitable observable, with low variance.
- Design computer experiment to minimise statistical error.
- Will see that working with *fixed length* walks confers dramatic advantage over growth algorithms.

Pivot algorithm

- Sample from the set of SAWs of a particular length.
- Markov chain:
  - Select a pivot site *uniformly at random*.
  - Randomly choose a lattice symmetry q (rotation or reflection).
  - Apply this symmetry to one of the two sub-walks created by splitting the walk at the pivot site.
  - If walk is self-avoiding: *accept* the pivot and update the configuration.
  - If walk is not self-avoiding: *reject* the pivot and keep the old configuration.
- Ergodic, samples SAWs uniformly at random.

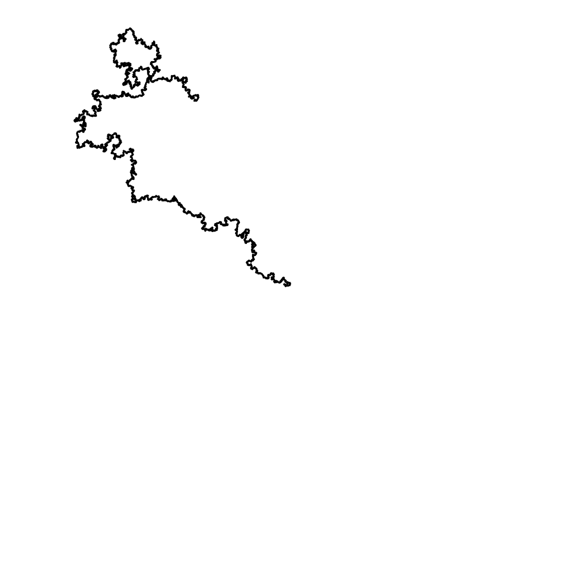


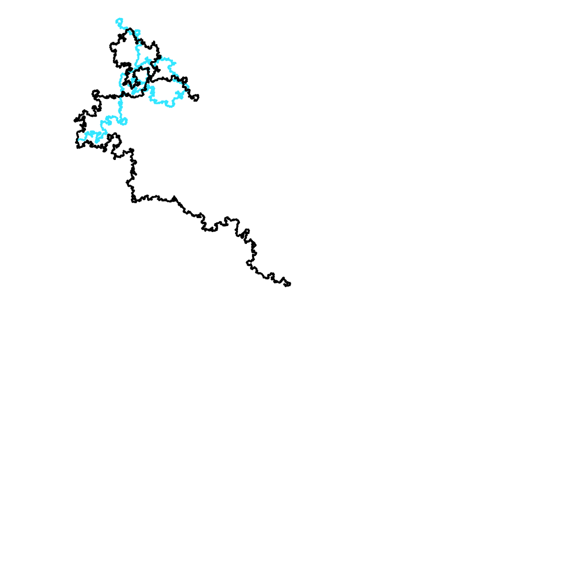
Example pivot move

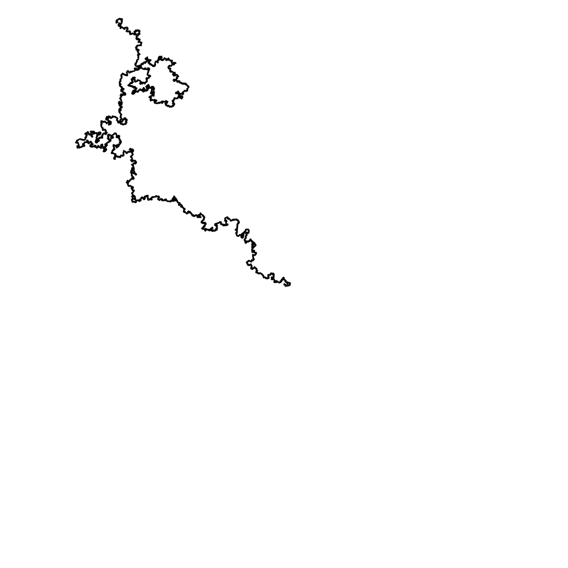
Why is it so effective?

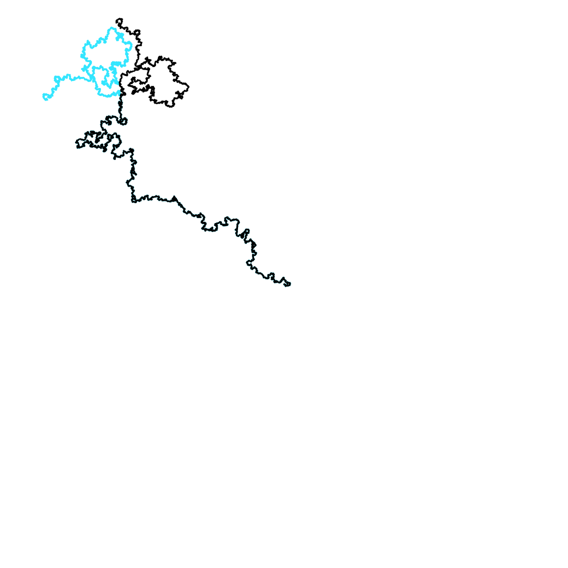
- Pivots are rarely successful, $\Pr = O(N^{-p})$, $p \approx 0.11$ for $\mathbb{Z}^3$.
- Every time a pivot attempt *is* successful there is a large change in global observables.
- Only need $O(1)$ successful pivots before we have an *essentially new* configuration.
- $\Rightarrow \tau_{\mathrm{int}} = O(N^p)$
- Best case for local moves gives $\tau_{\mathrm{int}} = O(N^2)$, dramatic improvement.

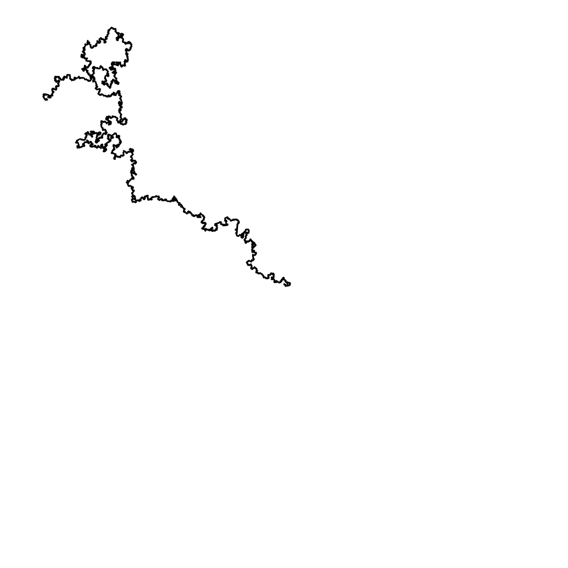
Will now show a sequence of *successful* pivots applied to an $n = 65536$ site SAW on the square lattice.

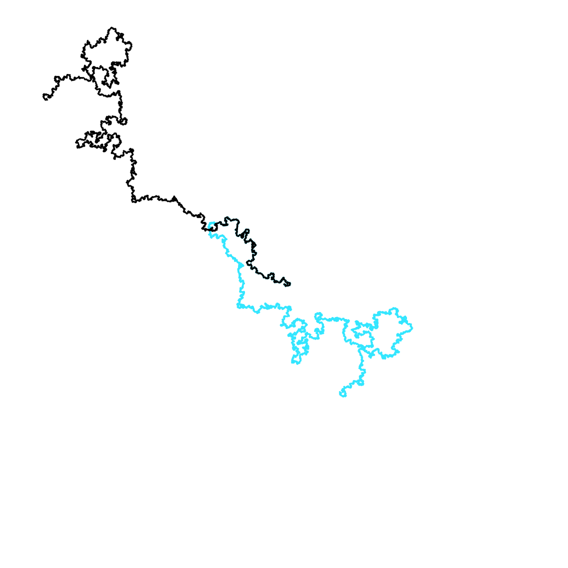


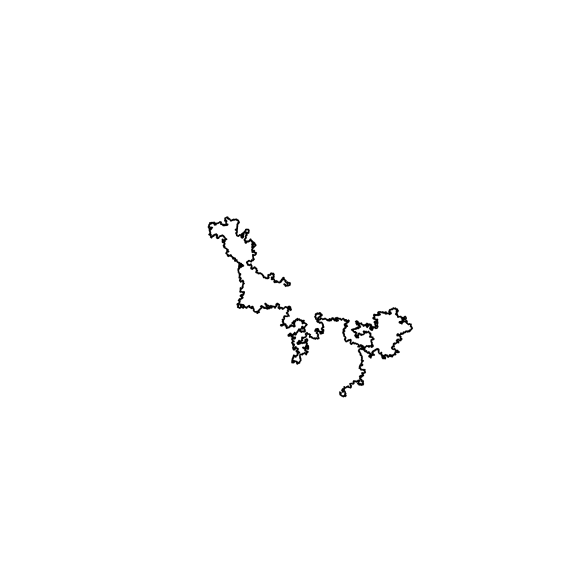


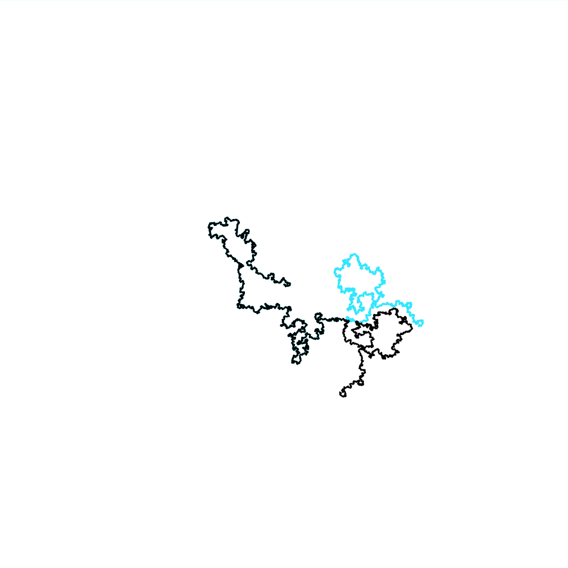


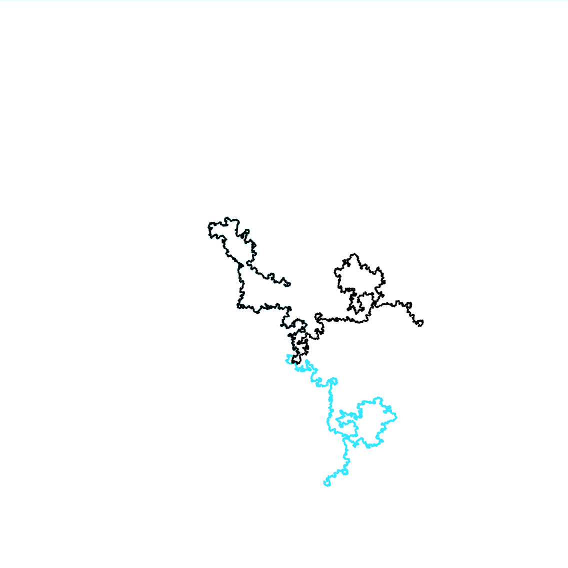


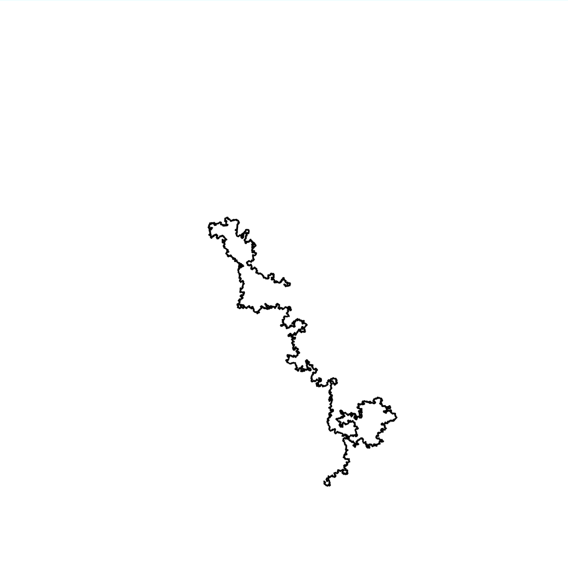


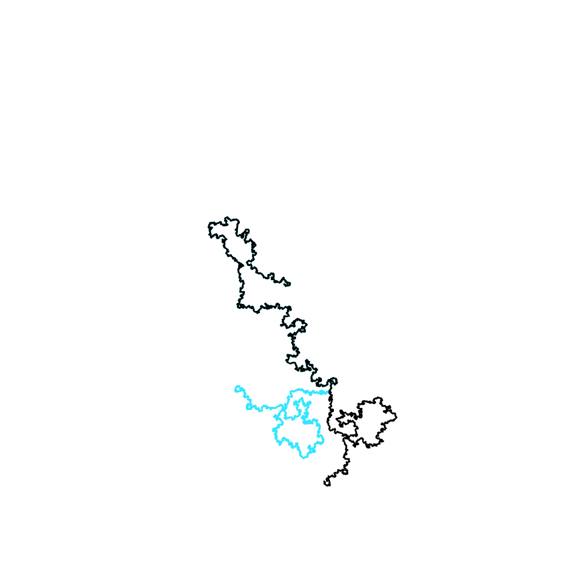


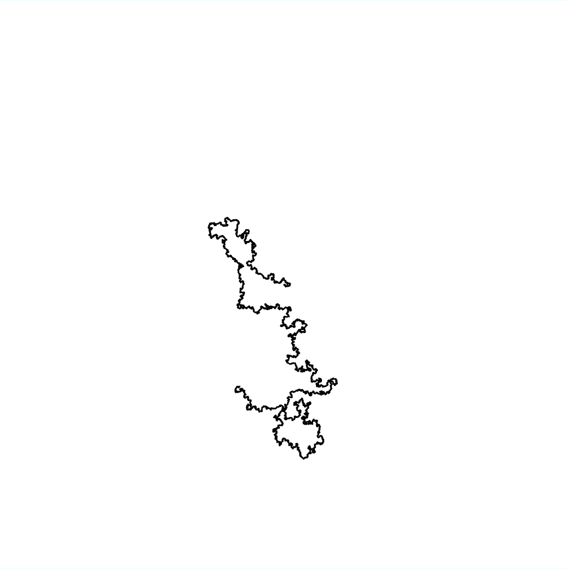


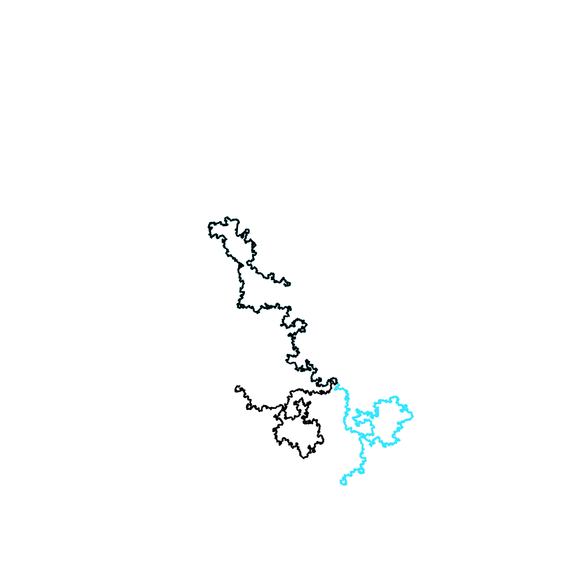


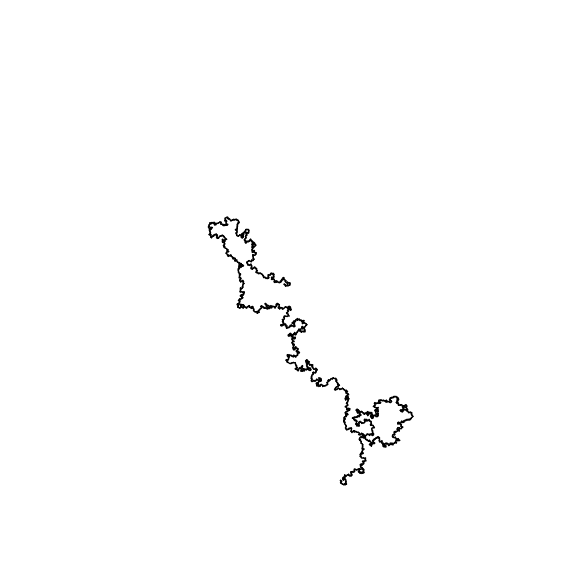


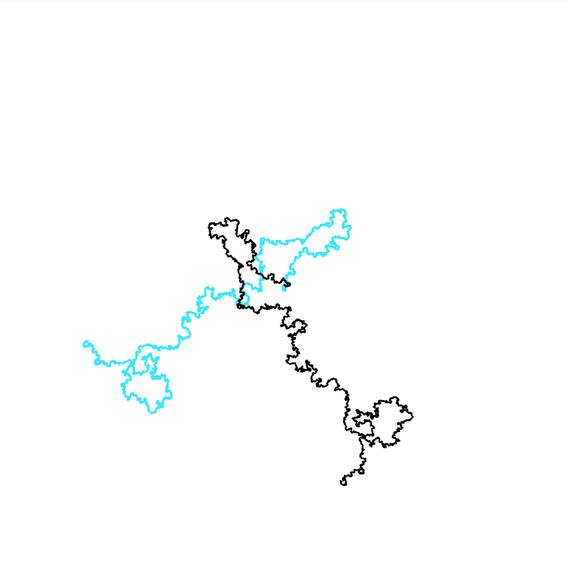


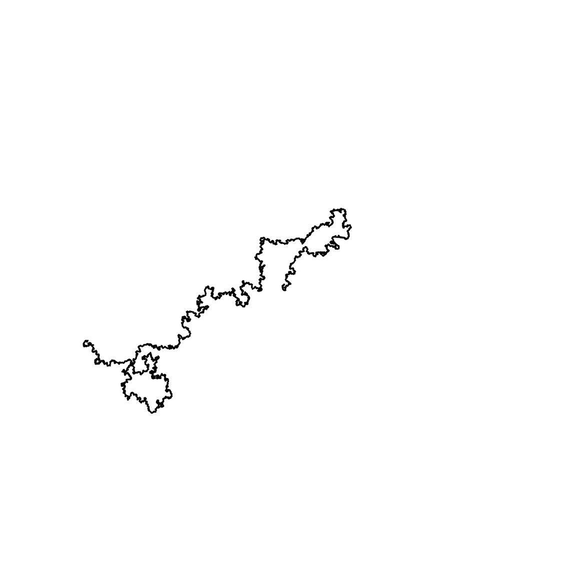


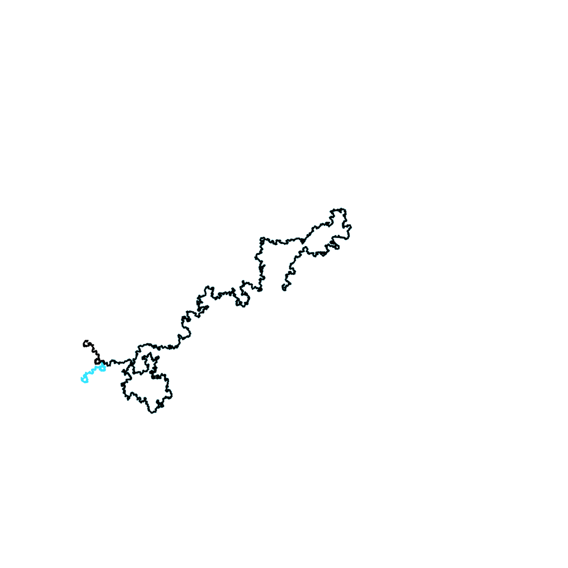


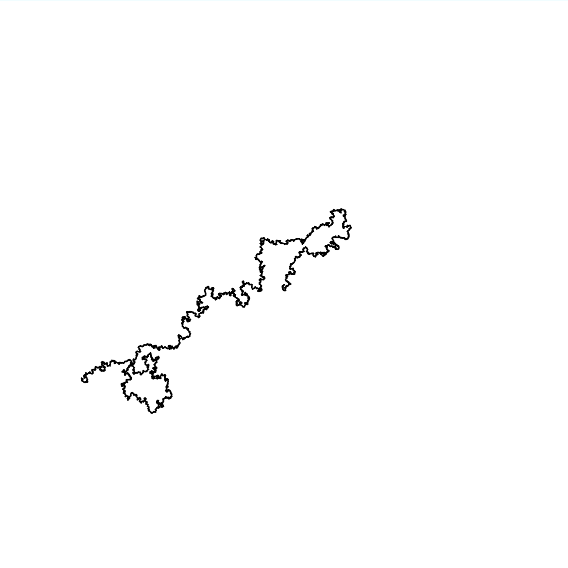


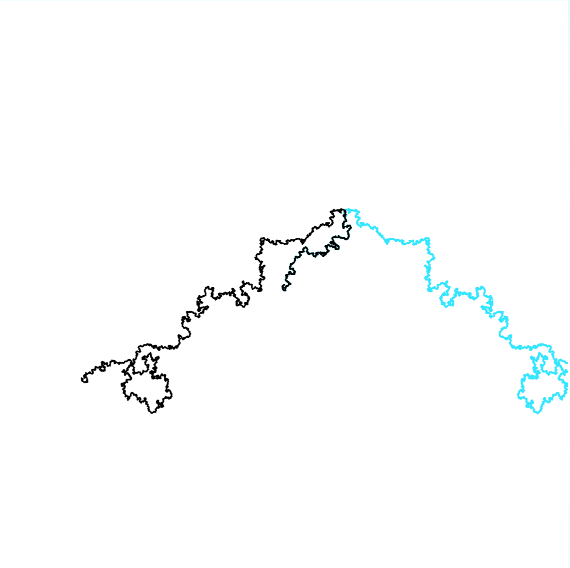


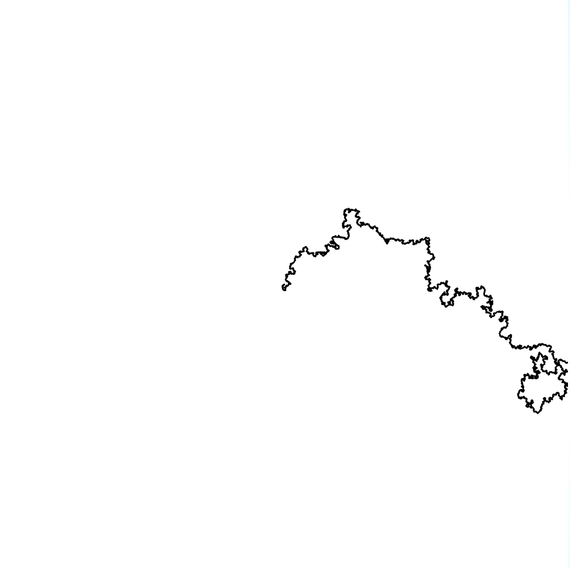


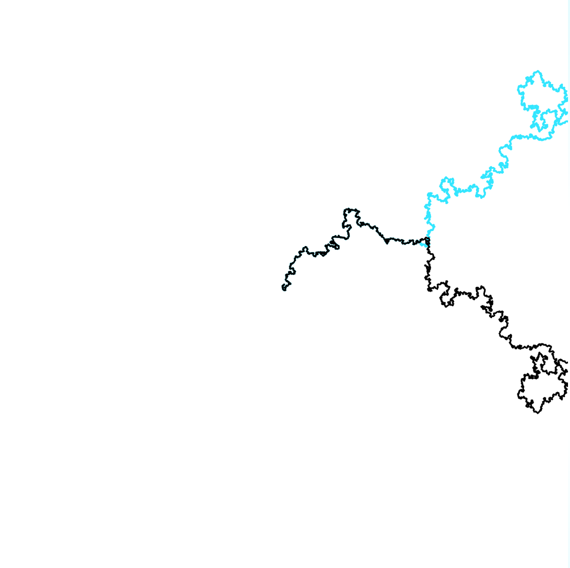


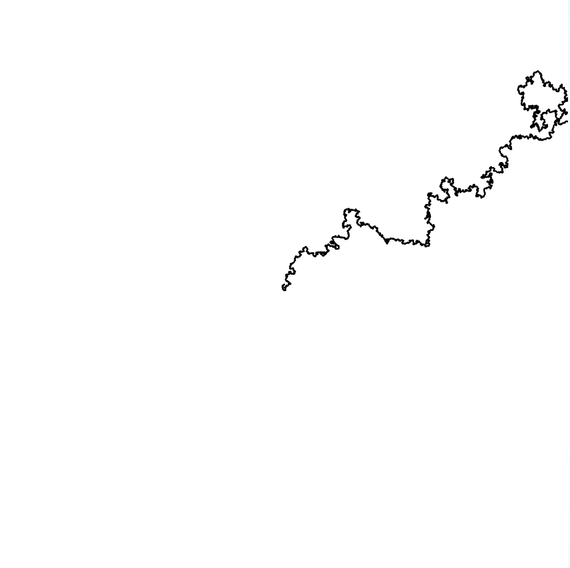


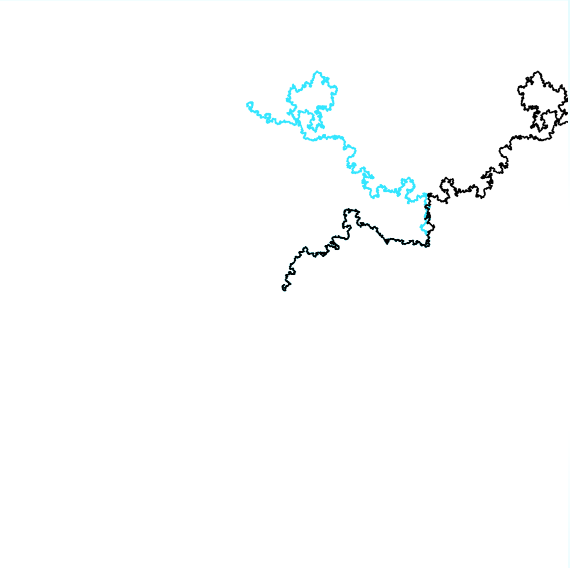


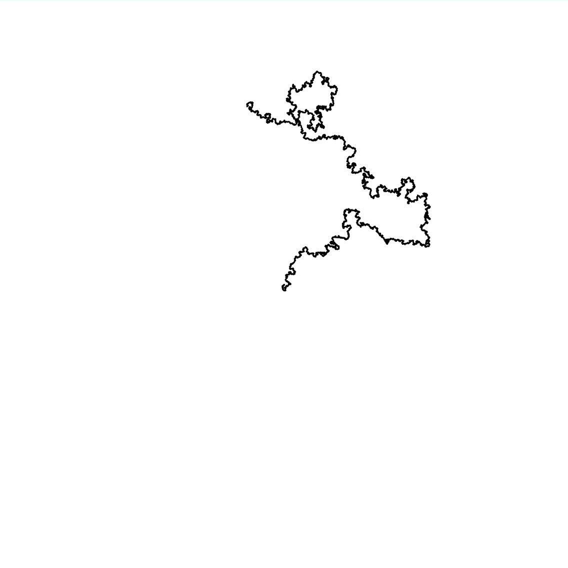


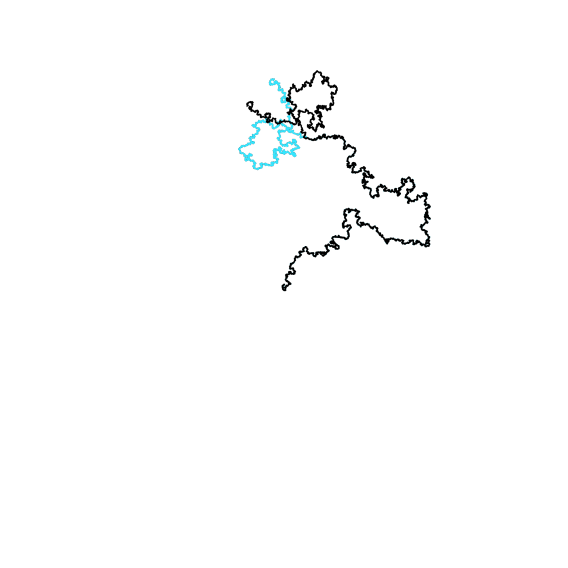


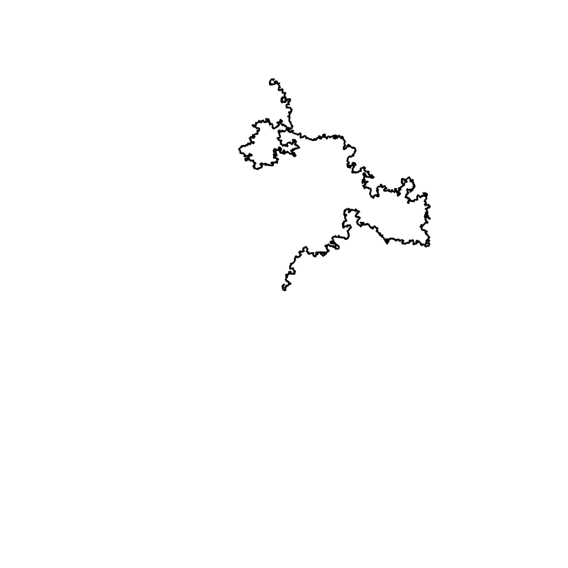








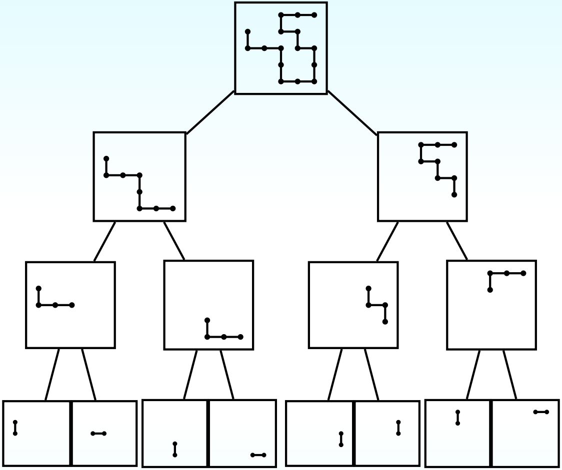




An efficient data structure for polymers

- Represent SAW as a binary tree.
- Enables global moves like pivots to be performed in CPU time $T(N) = O(\log N)$.
- c.f. $O(N^{1-p})$ for hash table implementation[2].
- Dramatic improvement for large N.
- New version in development: allows arbitrary lattices, wide variety of moves and symmetry operations ("GUM-tree").

[2] Neal Madras and Alan D. Sokal. "The Pivot Algorithm: A Highly Efficient Monte Carlo Method for the Self-Avoiding Walk". In: *J. Stat. Phys.* 50 (1988), pp. 109–186.



SAW-tree representation of a walk.

New SAW-tree implementation exploits geometric properties of walks.

- Represent the SAW as binary trees, with “bounding box” information for sub-walks.
- Other things to keep track of, so results in a complicated data structure.
- Most operations take $O(\log N)$, including intersection testing after a pivot attempt.

CPU time per attempted pivot, for SAWs of length N:

Lattice	Madras and Sokal	Kennedy	SAW-tree
Square	$O(N^{0.81})$	$O(N^{0.38})$	$o(\log N)$
Cubic	$O(N^{0.89})$	$O(N^{0.74})$	$O(\log N)$

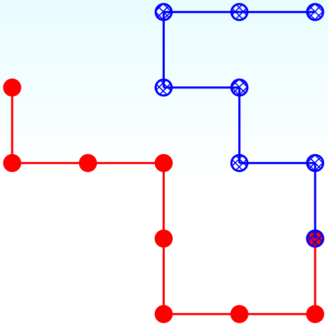
CPU time per attempted pivot, for SAWs of length N:

	$\mathbb{Z}^2$			$\mathbb{Z}^3$		
N	S-t (μs)	M&S/S-t	K/S-t	S-t (μs)	M&S/S-t	K/S-t
31	0.41	0.894	1.06	0.59	0.981	1.37
1023	0.87	5.15	1.90	1.71	6.31	3.75
32767	1.27	68.6	4.92	3.36	79.2	21.5
1048575	2.91	2510	32.2	7.53	3830	385
33554431	4.57	35200	134	12.58	61700	7130

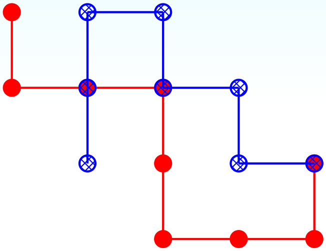
How to calculate c_N and μ?

- Would like to apply pivot algorithm in canonical ensemble.
- Approach: measure probability that object from larger set is a SAW, $|S| = P(x \in S | x \in T)|T|$, with $|T|$ known.
- Obvious choice: concatenating pairs of SAWs. Every $M + N$-step walk can be split into M and N step subwalks $\Rightarrow c_{M+N} \leq c_M c_N$ for all M, N.
- S_N set of walks of length N.
- $|S_{M+N}| = P(\omega_1 \circ \omega_2 \in S_{M+N} | (\omega_1, \omega_2) \in S_M \times S_N)|S_M||S_N|$
- Indicator function for successful concatenation is our observable, and

$$B(\omega_1, \omega_2) = \begin{cases} 0 & \text{if } \omega_1 \circ \omega_2 \text{ not self-avoiding} \\ 1 & \text{if } \omega_1 \circ \omega_2 \text{ self-avoiding} \end{cases}$$

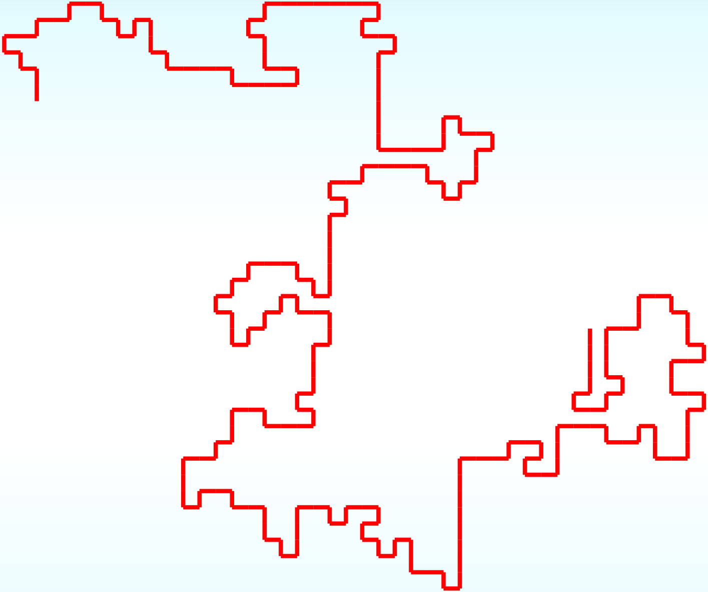


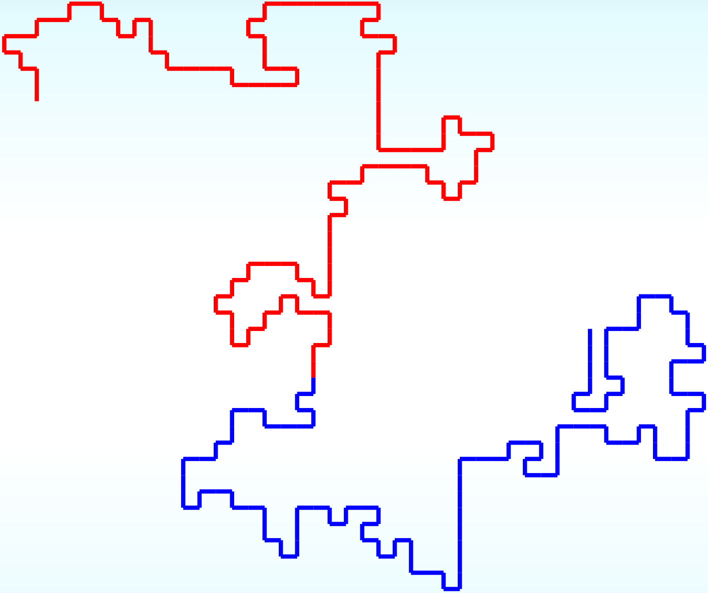
$B(\omega_1, \omega_2) = 1$

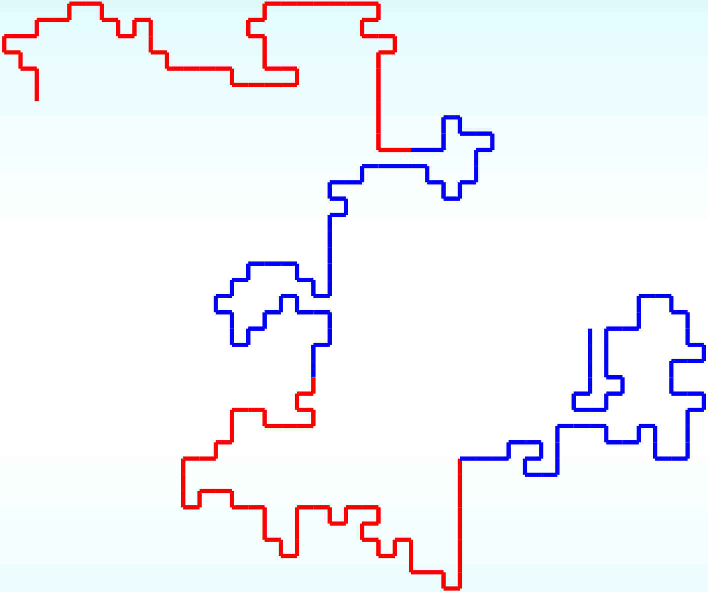


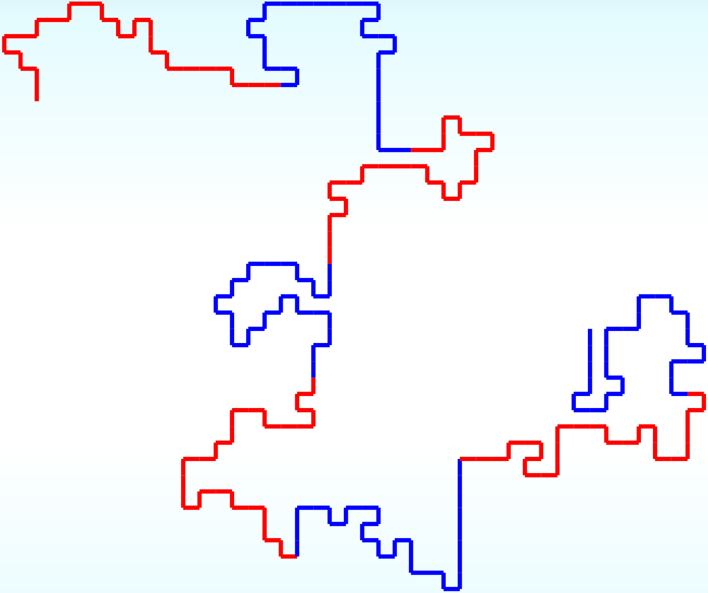
$B(\omega_1, \omega_2) = 0$

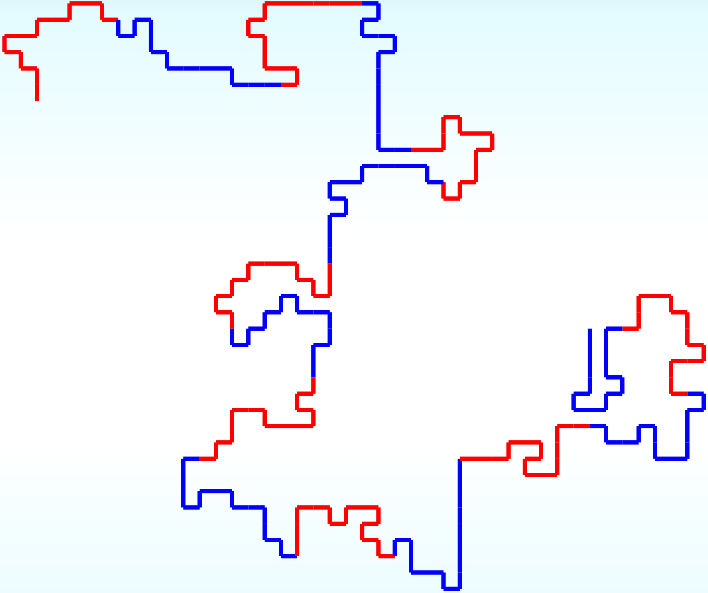
A long N step walk can be successively subdivided into smaller pieces.

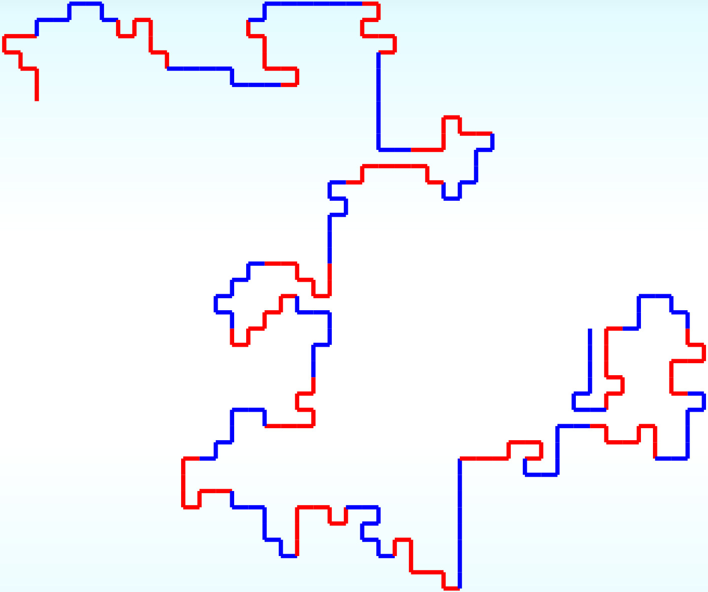


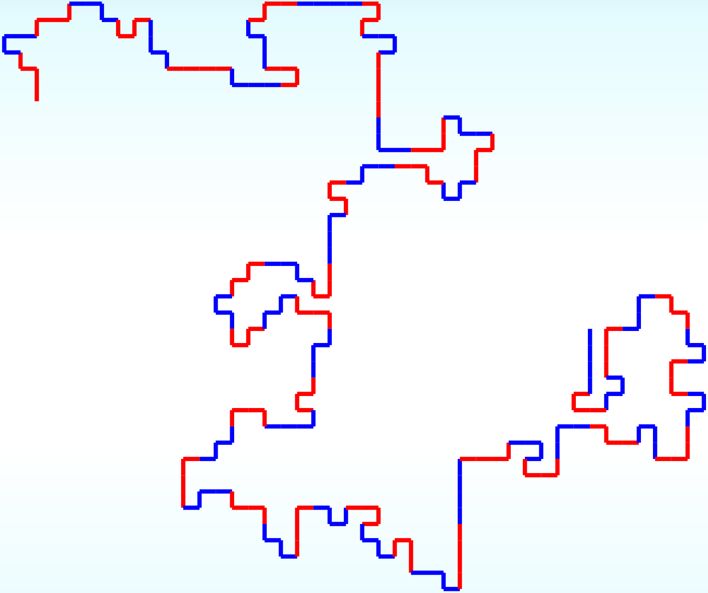


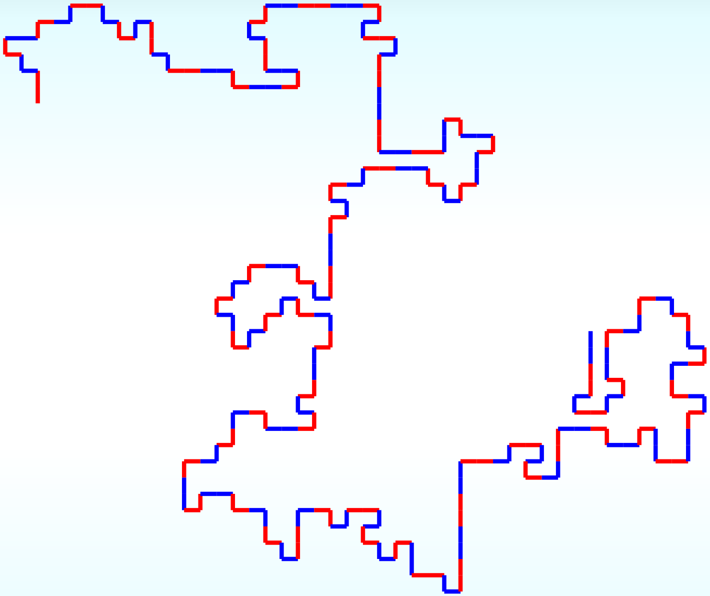


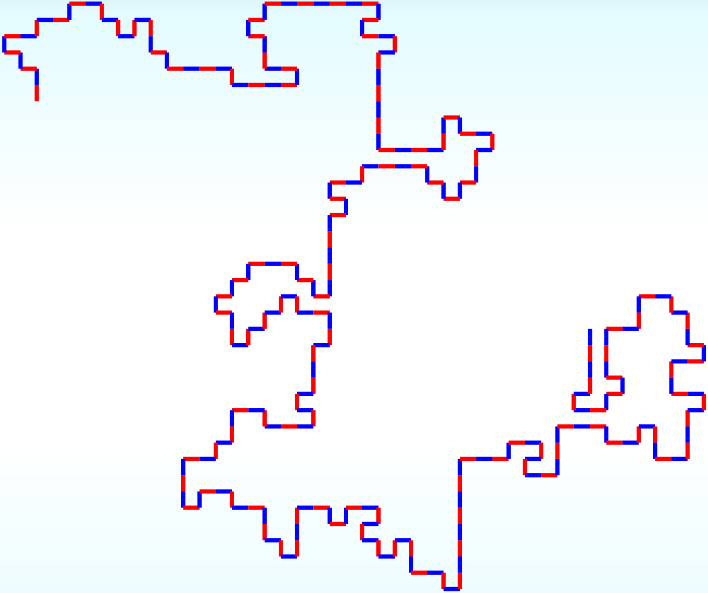












- Could choose $m, n = 36$ (longest known for $\mathbb{Z}^3$):

$$\langle B_{36,36} \rangle = \frac{c_{72}}{c_{36} c_{36}}$$

- Iterate to obtain estimates for c_N for longer walks.

$$\begin{aligned} c_N &= \frac{c_N}{c_{N/2}^2} \cdot \frac{c_{N/2}^2}{c_{N/4}^4} \cdots \frac{c_{2k}^{N/2k}}{c_k^{N/k}} c_k^{N/k} \\ &= \langle B_{N/2,N/2} \rangle \langle B_{N/4,N/4} \rangle^2 \cdots \langle B_{k,k} \rangle^{N/2k} c_k^{N/k} \\ \log c_N &= \log\langle B_{N/2,N/2} \rangle + 2 \log\langle B_{N/4,N/4} \rangle + \cdots \\ &\cdots + \frac{N}{2k} \log\langle B_{k,k} \rangle + \frac{N}{k} \log c_k \end{aligned}$$

where c_k is known.

- Telescoping, with length doubling at each iteration.
- C.f. sequential growth, N steps, product of N factors.

- Can then use $c_N \sim A\mu^N N^{\gamma-1}$ to estimate μ:

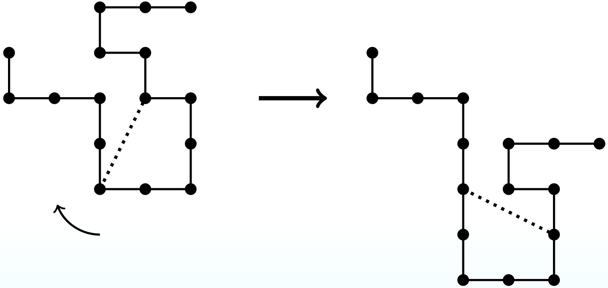
$$\begin{aligned}\log \mu_N &\equiv \frac{1}{N}\log c_N \\ &= \frac{1}{k}\log c_k + \frac{1}{2k}\log\langle B_{k,k}\rangle + \frac{1}{4k}\log\langle B_{2k,2k}\rangle + \cdots \\ &\qquad \cdots + \frac{1}{N}\log\langle B_{N/2,N/2}\rangle \\ &= \log\mu + \frac{(\gamma-1)\log N}{N} + \frac{\log A}{N} + \text{corrections}\end{aligned}$$

- Corrections vanish with increasing N! In limit of large N systematic error of estimator $\to 0$.

Scale free moves

- Need to calculate $\langle B_{k,k} \rangle$, $\langle B_{2k,2k} \rangle$, $\cdots$
- Use pivot algorithm / SAW-tree.
- How many pivots must be completed before two walks are "essentially new" configurations with respect to observable B?
- Shape of walks close to the joint clearly important.
- Uniform pivot sites: $\widetilde{\tau}_{\mathrm{int}} = \Omega(N)$ (due to trapped configurations).
- Choose distance from joint uniformly from all distance scales, i.e. $u = \log(\text{distance})$ chosen uniformly at random.
- When successful moves have been made at all length scales walks are essentially new with respect to B.
- Now: $\widetilde{\tau}_{\mathrm{int}} = N^p \log^2 N$, only a penalty of $\log N$!

- Scale free moves: idea should be useful for any polymer system with intermediate length scales, e.g. confined polymers, star polymers, polymer knotting.
- Perform cut-and-paste moves at intermediate length scales - system will self-select for optimal length scale.



Example cut-and paste move; note that only the middle walk is rotated.

Error estimate

- Expected error, for same CPU time, diminishes as a power law for higher order terms in the sum!

$$\log \mu_N = \frac{1}{k} \log c_k + \frac{1}{2k} \log \langle B_{k,k} \rangle + \cdots + \frac{1}{N} \log \langle B_{N/2,N/2} \rangle$$

- Partition CPU time amongst different terms to minimize overall statistical error (short test run).

$$\sigma^2 = \sum \frac{a_i^2}{t_i} \qquad \text{Total time } t = \sum t_i$$

$$\Rightarrow t_i = \frac{a_i}{\sum a_i} t, \qquad \sigma = \frac{\sum a_i}{\sqrt{t}}$$

- Can accurately predict error on estimate for c_N prior to start of computer experiment.
- Dominated by low k contribution, appropriate partitioning of effort reduced error by $O(\sqrt{\log N})$. Relative error in c_N proportional to $1/k$.

- Can make unbiased estimates of c_N, for N up to 10^9 or so.
- Can push calculation to sufficiently large N s.t. asymptotic corrections for μ completely eliminated.
- $\Rightarrow$ Systematic error for μ negligible, error purely statistical.

Results for c_N

- Calculated $\log c_N$ with relative error of approximately 4×10^{-9} up to $N = 38797311$ (about 60000 CPU hours).
- Concentrated on $\mathbb{Z}^3$ because asymptotic behaviour for $\mathbb{Z}^2$ well understood from series.
- $c_{9471} = 1.43323(8) \times 10^{6352}$
- $c_{38797311} = 7 \times 10^{26018276}$. Confidence interval of mantissa is $(6.6, 8.2)$.
- For comparison, see[3]. Relative error from PERM and related algorithms of the order of 10^{-3} for short walks of 100 steps. Not a fair comparison:
  - Not much CPU time used, i.e. not serious computer experiments.
  - Estimates would degrade for large N. Best case: error increasing as $O(\sqrt{N})$ for fixed CPU time.

[3] E. J. Janse van Rensburg. "Approximate Enumeration of Self-Avoiding Walks". In: *Algorithmic Probability and Combinatorics* 520 (2010), pp. 127–151.

n	improved μ_n	σ	$\sigma(\gamma)\log n/n$	$\sigma(A)/n$	$n^{-1-\Delta_1}$
73	4.683732537	1.7×10^{-8}	2.8×10^{-6}	0.00013	0.0076
147	4.683926585	2.1×10^{-8}	1.6×10^{-6}	6.4×10^{-5}	0.0026
295	4.684000343	2.4×10^{-8}	9.1×10^{-7}	3.2×10^{-5}	0.00093
591	4.684026833	2.5×10^{-8}	5.1×10^{-7}	1.6×10^{-5}	0.00033
1183	4.684035895	2.6×10^{-8}	2.8×10^{-7}	7.9×10^{-6}	0.00012
2367	4.684038838	2.6×10^{-8}	1.5×10^{-7}	4.0×10^{-6}	4.1×10^{-5}
4735	4.684039717	2.7×10^{-8}	8.4×10^{-8}	2.0×10^{-6}	1.4×10^{-5}
9471	4.684039941	2.7×10^{-8}	4.5×10^{-8}	9.9×10^{-7}	5.1×10^{-6}
18943	4.684039976	2.7×10^{-8}	2.4×10^{-8}	4.9×10^{-7}	1.8×10^{-6}
37887	4.684039966	2.7×10^{-8}	1.3×10^{-8}	2.5×10^{-7}	6.4×10^{-7}
75775	4.684039954	2.7×10^{-8}	6.9×10^{-9}	1.2×10^{-7}	2.2×10^{-7}
151551	4.684039944	2.7×10^{-8}	3.7×10^{-9}	6.2×10^{-8}	7.9×10^{-8}
303103	4.684039937	2.7×10^{-8}	2.0×10^{-9}	3.1×10^{-8}	2.8×10^{-8}
606207	4.684039934	2.7×10^{-8}	1.0×10^{-9}	1.5×10^{-8}	9.9×10^{-9}
1212415	4.684039932	2.7×10^{-8}	5.4×10^{-10}	7.7×10^{-9}	3.5×10^{-9}
2424831	4.684039931	2.7×10^{-8}	2.8×10^{-10}	3.9×10^{-9}	1.2×10^{-9}
4849663	4.684039931	2.7×10^{-8}	1.5×10^{-10}	1.9×10^{-9}	4.4×10^{-10}
9699327	4.684039931	2.7×10^{-8}	7.8×10^{-11}	9.7×10^{-10}	1.6×10^{-10}
19398655	4.684039931	2.7×10^{-8}	4.0×10^{-11}	4.8×10^{-10}	5.5×10^{-11}
38797311	4.684039931	2.7×10^{-8}	2.1×10^{-11}	2.4×10^{-10}	1.9×10^{-11}

Results for μ

- For $\mathbb{Z}^3$ we have:
- PERM: $\mu = 4.684038(6)$ (Hsu and Grassberger, "Polymers confined between two parallel plane walls")
- Series: $\mu = 4.68404(1)$ (Clisby, Liang, and Slade, "Self-avoiding walk enumeration via the lace expansion")
- Series: $\mu = 4.684040(5)$ (Schram, Barkema, and Bisseling, "Exact enumeration of self-avoiding walks")
- Pivot: $\mu = 4.68403993(3)$, almost 200 times more accurate than previous best ($\sigma = 2.7 \times 10^{-8}$).

Conclusion

- Simple computer experiment.
- Different ingredients fit together to produce extremely accurate estimates.
- Choose a Monte Carlo scheme which enables efficient sampling (large jumps in state space)
- Efficient data structures help.
- Choose observable and design of computer experiment carefully.
- Scale free moves may be useful for other problems.