# From Rosenbluth Sampling to PERM

rare event sampling with stochastic growth algorithms

Thomas Prellberg

School of Mathematical Sciences
Queen Mary, University of London

MCS11
August 15-26, 2011

# Topic Outline

# Outline

# Sampling Algorithms

Markov Chain Monte Carlo (MCMC) Methods

- Transition from state $i$ to state $j$ with probability $P_{i,j}$
- Ergodicity
- Detailed Balance Condition for Equilibrium

$$\pi_i P_{i,j} = \pi_j P_{j,i}$$

- Normally fixed system size (e.g. constant number of spins)

# Sampling Algorithms

Markov Chain Monte Carlo (MCMC) Methods

- Transition from state $i$ to state $j$ with probability $P_{i,j}$
- Ergodicity
- Detailed Balance Condition for Equilibrium

$$\pi_i P_{i,j} = \pi_j P_{j,i}$$

- Normally fixed system size (e.g. constant number of spins)

Stochastic Growth Methods

- No Markov Chain
- Grow independent configurations from scratch
- Incomplete enumeration

At each growth step, select one of all possible continuations

# Lattice paths

Relevant as model of polymer chains

# Lattice paths

Relevant as model of polymer chains

## Lattice Walks

- Physical space $\rightarrow$ simple cubic lattice $\mathbb{Z}^3$
- Ghost Polymer $\rightarrow$ $n$-step random walk

# Lattice paths

Relevant as model of polymer chains

## Lattice Walks

- Physical space $\rightarrow$ simple cubic lattice $\mathbb{Z}^3$
- Ghost Polymer $\rightarrow$ $n$-step random walk

## Self-Avoiding Walks

- Polymer with Excluded Volume $\rightarrow$ self-avoiding $n$-step random walk

# Lattice paths

Relevant as model of polymer chains

## Lattice Walks

- Physical space $\rightarrow$ simple cubic lattice $\mathbb{Z}^3$
- Ghost Polymer $\rightarrow$ $n$-step random walk

## Self-Avoiding Walks

- Polymer with Excluded Volume $\rightarrow$ self-avoiding $n$-step random walk

## Interacting Self-Avoiding Walks

- Quality of solvent $\rightarrow$ short-range interaction $\epsilon$, $E_n(\varphi) = m(\varphi)\epsilon$

# Lattice paths

Relevant as model of polymer chains

## Lattice Walks

- Physical space $\rightarrow$ simple cubic lattice $\mathbb{Z}^3$
- Ghost Polymer $\rightarrow$ $n$-step random walk

## Self-Avoiding Walks

- Polymer with Excluded Volume $\rightarrow$ self-avoiding $n$-step random walk

## Interacting Self-Avoiding Walks

- Quality of solvent $\rightarrow$ short-range interaction $\epsilon$, $E_n(\varphi) = m(\varphi)\epsilon$

(can add other/more parameters)

# Lattice paths

Relevant as model of polymer chains

## Lattice Walks

- Physical space $\rightarrow$ simple cubic lattice $\mathbb{Z}^3$
- Ghost Polymer $\rightarrow$ $n$-step random walk

## Self-Avoiding Walks

- Polymer with Excluded Volume $\rightarrow$ self-avoiding $n$-step random walk

## Interacting Self-Avoiding Walks

- Quality of solvent $\rightarrow$ short-range interaction $\epsilon$, $E_n(\varphi) = m(\varphi)\epsilon$

(can add other/more parameters)

## Goal of these Lectures

Present growth algorithms with *uniform sampling* for lattice walks

# Outline

# Outline

# Simple Random Walk in One Dimension

Model of directed polymer in $1+1$ dimensions

- Start at origin and step to left or right with equal probability

# Simple Random Walk in One Dimension

Model of directed polymer in $1+1$ dimensions

- Start at origin and step to left or right with equal probability
- $2^n$ possible random walks with $n$ steps
- each walk generated with equal weight

# Simple Random Walk in One Dimension

Model of directed polymer in $1 + 1$ dimensions

- Start at origin and step to left or right with equal probability
- $2^n$ possible random walks with $n$ steps
- each walk generated with equal weight

Distribution of endpoints

# Simple Random Walk in One Dimension

Model of directed polymer in $1+1$ dimensions

- Start at origin and step to left or right with equal probability
- $2^n$ possible random walks with $n$ steps
- each walk generated with equal weight

Distribution of endpoints

- walks end at position $k+(n-k)$ with probability

$$P_{n,k} = \frac{1}{2^n}\binom{n}{k}$$

($k$ steps to the right, $n-k$ steps to the left)

# Simple Sampling of Simple Random Walk

$s_{\cdot,\cdot} \leftarrow 0$
$Samples \leftarrow 0$
**while** $Samples < MaxSamples$ **do**
  $Samples \leftarrow Samples + 1$
  $n \leftarrow 0,\ k \leftarrow 0$
  $s_{0,0} \leftarrow s_{0,0} + 1$
  **while** $n < MaxLength$ **do**
    $n \leftarrow n + 1$
    Draw random number $r \in [0, 1]$
    **if** $r > 1/2$ **then**
      $k \leftarrow k + 1$
    **end if**
    $s_{n,k} \leftarrow s_{n,k} + 1$
  **end while**
**end while**

# Properties of Simple Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step walk is grown with equal probability
- Impossible to sample the tails of the distribution ($P_{n,0} = 2^{-n}$)

# Properties of Simple Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step walk is grown with equal probability
- Impossible to sample the tails of the distribution $(P_{n,0} = 2^{-n})$

How can we tweak the algorithm to reach the tails?

# Outline

# Biased Sampling

Introduce bias

- Jump to left with probability $p$
- Jump to right with probability $1 - p$

# Biased Sampling

Introduce bias

- Jump to left with probability $p$
- Jump to right with probability $1 - p$

Distribution of endpoints

# Biased Sampling

Introduce bias

- Jump to left with probability $p$
- Jump to right with probability $1-p$

Distribution of endpoints

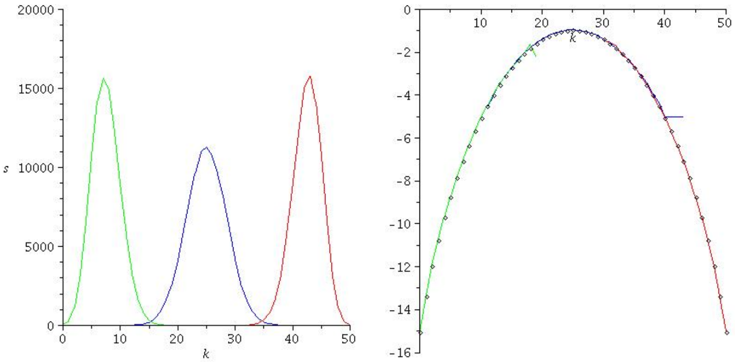
- Walks end at position $k+(n-k)$ with probability

$$P_{n,k} = \binom{n}{k} p^{n-k}(1-p)^k$$

($k$ steps to the right, $n-k$ steps to the left)

# Biased Sampling of Simple Random Walk

$s_{\cdot,\cdot} \leftarrow 0$, $w_{\cdot,\cdot} \leftarrow 0$
$Samples \leftarrow 0$
**while** $Samples < MaxSamples$ **do**
  $Samples \leftarrow Samples + 1$
  $n \leftarrow 0$, $k \leftarrow 0$, $Weight \leftarrow 1$
  $s_{0,0} \leftarrow s_{0,0} + 1$, $w_{0,0} \leftarrow w_{0,0} + Weight$
  **while** $n < MaxLength$ **do**
    $n \leftarrow n + 1$
    Draw random number $r \in [0, 1]$
    **if** $r > p$ **then**
      $k \leftarrow k + 1$, $Weight \leftarrow Weight/2(1 - p)$
    **else**
      $Weight \leftarrow Weight/2p$
    **end if**
    $s_{n,k} \leftarrow s_{n,k} + 1$, $w_{n,k} \leftarrow w_{n,k} + Weight$
  **end while**
**end while**




Biased sampling of simple random walk for $n = 50$ steps and bias $p = 0.85$ (green), $p = 0.5$ (blue), and $p = 0.15$ (red). For each simulation, 100000 samples were generated.

# Properties of Biased Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step walk ending at position $k$ is grown with equal probability
- Distributions concentrated around $k = pn$ with width $O(\sqrt{n})$

# Properties of Biased Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step walk ending at position $k$ is grown with equal probability
- Distributions concentrated around $k = pn$ with width $O(\sqrt{n})$

- To cover the whole distribution, need $O(\sqrt{n})$ individual simulations
- Need to choose different biases $p$ such that distributions overlap

# Properties of Biased Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step walk ending at position $k$ is grown with equal probability
- Distributions concentrated around $k = pn$ with width $O(\sqrt{n})$

- To cover the whole distribution, need $O(\sqrt{n})$ individual simulations
- Need to choose different biases $p$ such that distributions overlap

Can we tweak the algorithm to avoid several simulations?

# Outline

# Uniform Sampling

**Main Idea**

Allow for *local* biassing of random walk

# Uniform Sampling

## Main Idea

Allow for *local* biassing of random walk

- replace global bias $p$ by local bias

$$p_{n,k} = \frac{n+1-k}{n+2}$$

[EXERCISE]

# Uniform Sampling

## Main Idea

Allow for *local* biassing of random walk

- replace global bias $p$ by local bias

$$p_{n,k} = \frac{n+1-k}{n+2}$$

[EXERCISE]

- Change weight of configuration by factor

$$1/2p_{n,k} \quad \text{or} \quad 1/2(1-p_{n,k})$$

# Uniform Sampling of Simple Random Walk

Initialize local biases

**for** $n = 0$ to $MaxLength$ **do**
  **for** $k = 0$ to $n$ **do**
    $p_{n,k} \leftarrow (n+1-k)/(n+2)$
  **end for**
**end for**

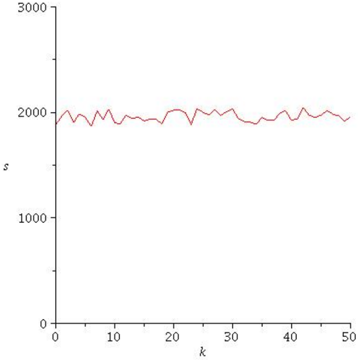
# Uniform Sampling of Simple Random Walk

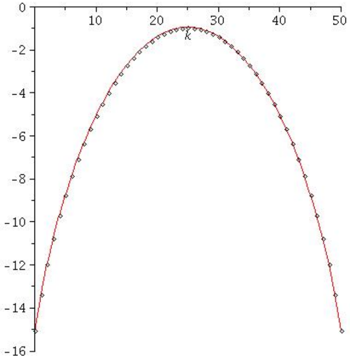
Initialize local biases

**for** $n = 0$ to $MaxLength$ **do**
  **for** $k = 0$ to $n$ **do**
    $p_{n,k} \leftarrow (n+1-k)/(n+2)$
  **end for**
**end for**

and change the algorithm slightly

**if** $r > p_{n,k}$ **then**
  $k \leftarrow k+1,\ Weight \leftarrow Weight/2(1-p_{n,k})$
**else**
  $Weight \leftarrow Weight/2p_{n,k}$
**end if**




Uniform sampling of simple random walk for $n = 50$ steps, with 100000 samples generated.

# Properties of Uniform Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step walk ending at position $k$ is grown with equal probability

$$P_{n,k} = \frac{1}{n+1}$$

and has weight

$$W_{n,k} = \frac{n+1}{2^n} \binom{n}{k}$$

[Exercise]

- Distribution is perfectly uniform

# Properties of Uniform Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step walk ending at position $k$ is grown with equal probability

$$P_{n,k} = \frac{1}{n+1}$$

and has weight

$$W_{n,k} = \frac{n+1}{2^n}\binom{n}{k}$$

[EXERCISE]

- Distribution is perfectly uniform
- One simulation suffices

# Properties of Uniform Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step walk ending at position $k$ is grown with equal probability

$$P_{n,k} = \frac{1}{n+1}$$

and has weight

$$W_{n,k} = \frac{n+1}{2^n}\binom{n}{k}$$

[Exercise]

- Distribution is perfectly uniform
- One simulation suffices

What if we don't know how to compute the biases $p_{n,k}$?

# Outline

# From Biases to Weights: A Change of View

- Correct choice of local biases $p_{n,k}$ achieves uniform sampling

# From Biases to Weights: A Change of View

- Correct choice of local biases $p_{n,k}$ achieves uniform sampling
- If local biases are incorrect, sampling will be non-uniform

# From Biases to Weights: A Change of View

- Correct choice of local biases $p_{n,k}$ achieves uniform sampling
- If local biases are incorrect, sampling will be non-uniform

**Idea**

Use this non-uniformity to iteratively tune biases

# From Biases to Weights: A Change of View

- Correct choice of local biases $p_{n,k}$ achieves uniform sampling
- If local biases are incorrect, sampling will be non-uniform

**Idea**

Use this non-uniformity to iteratively tune biases

Unfortunately, this is a *bad* idea, the resulting algorithm is inherently unstable. (Or maybe I haven't been smart enough - space for new ideas.)

# From Biases to Weights: A Change of View

- Correct choice of local biases $p_{n,k}$ achieves uniform sampling
- If local biases are incorrect, sampling will be non-uniform

**Idea**

Use this non-uniformity to iteratively tune biases

Unfortunately, this is a *bad* idea, the resulting algorithm is inherently unstable. (Or maybe I haven't been smart enough - space for new ideas.)

**Better Idea**

Use this non-uniformity to iteratively tune weights

# From Biases to Weights: A Change of View

- Correct choice of local biases $p_{n,k}$ achieves uniform sampling
- If local biases are incorrect, sampling will be non-uniform

**Idea**

Use this non-uniformity to iteratively tune biases

Unfortunately, this is a *bad* idea, the resulting algorithm is inherently unstable. (Or maybe I haven't been smart enough - space for new ideas.)

**Better Idea**

Use this non-uniformity to iteratively tune weights

This works!

# Pruned and Enriched Sampling

- For uniform sampling we need to achieve

$$W_{n,k} = \frac{n+1}{2^n} \binom{n}{k}$$

# Pruned and Enriched Sampling

- For uniform sampling we need to achieve

$$W_{n,k} = \frac{n+1}{2^n}\binom{n}{k}$$

- Simple sampling generates samples with weight $W_{n,k} = 2^{-n}$

# Pruned and Enriched Sampling

- For uniform sampling we need to achieve

$$W_{n,k} = \frac{n+1}{2^n}\binom{n}{k}$$

- Simple sampling generates samples with weight $W_{n,k} = 2^{-n}$

## Pruning and Enrichment Strategy

- **Pruning** If the weight is too small, remove the configuration probabilistically
- **Enrichment** If the weight is too large, make several copies of the configuration

# Pruning and Enrichment (ctd)

Suppose a walk has been generated with weight $w$, but ought to have target weight $W$.

# Pruning and Enrichment (ctd)

Suppose a walk has been generated with weight $w$, but ought to have target weight $W$.

- Compute ratio $R = w/W$

# Pruning and Enrichment (ctd)

Suppose a walk has been generated with weight $w$, but ought to have target weight $W$.

- Compute ratio $R = w/W$
- If $R = 1$, do nothing

# Pruning and Enrichment (ctd)

Suppose a walk has been generated with weight $w$, but ought to have target weight $W$.

- Compute ratio $R = w/W$
- If $R = 1$, do nothing
- If $R < 1$, stop growing with probability $1 - R$

# Pruning and Enrichment (ctd)

Suppose a walk has been generated with weight $w$, but ought to have target weight $W$.

- Compute ratio $R = w/W$
- If $R = 1$, do nothing
- If $R < 1$, stop growing with probability $1 - R$
- If $R > 1$, make $\lfloor R \rfloor + 1$ copies with probability $p = R - \lfloor R \rfloor$ and $\lfloor R \rfloor$ copies with probability $1 - p$

# Pruning and Enrichment (ctd)

Suppose a walk has been generated with weight $w$, but ought to have target weight $W$.

- Compute ratio $R = w/W$
- If $R = 1$, do nothing
- If $R < 1$, stop growing with probability $1 - R$
- If $R > 1$, make $\lfloor R \rfloor + 1$ copies with probability $p = R - \lfloor R \rfloor$ and $\lfloor R \rfloor$ copies with probability $1 - p$
- Continue growing with weight $w$ set to target weight $W$

# Pruning and Enrichment (ctd)

Suppose a walk has been generated with weight $w$, but ought to have target weight $W$.

- Compute ratio $R = w/W$
- If $R = 1$, do nothing
- If $R < 1$, stop growing with probability $1 - R$
- If $R > 1$, make $\lfloor R \rfloor + 1$ copies with probability $p = R - \lfloor R \rfloor$ and $\lfloor R \rfloor$ copies with probability $1 - p$
- Continue growing with weight $w$ set to target weight $W$

Pruning and enrichment leads to the generation of a tree-like structure of correlated walks. All walks grown from the same seed are called a *tour*

# Pruning and Enrichment (ctd)

Suppose a walk has been generated with weight $w$, but ought to have target weight $W$.

- Compute ratio $R = w/W$
- If $R = 1$, do nothing
- If $R < 1$, stop growing with probability $1 - R$
- If $R > 1$, make $\lfloor R \rfloor + 1$ copies with probability $p = R - \lfloor R \rfloor$ and $\lfloor R \rfloor$ copies with probability $1 - p$
- Continue growing with weight $w$ set to target weight $W$

Pruning and enrichment leads to the generation of a tree-like structure of correlated walks. All walks grown from the same seed are called a *tour*

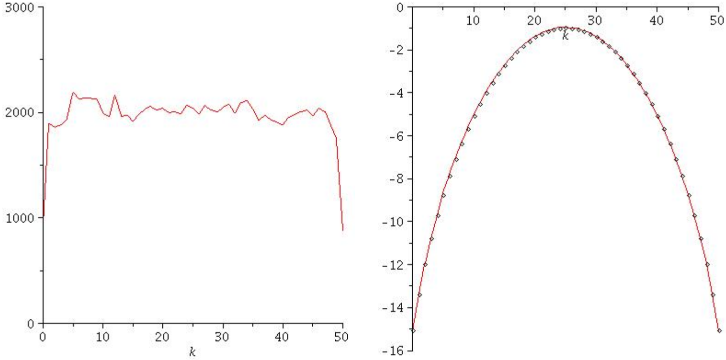
## Drawback of Pruning and Enrichment

- Need to deal with correlated data
- No a priori error analysis available
- A posteriori error analysis very difficult (only heuristics)

$Tours \leftarrow 0$, $n \leftarrow 0$, $k_0 \leftarrow 0$, $Weight_0 \leftarrow 1$
$s_{0,k_0} \leftarrow s_{0,k_0} + 1$, $w_{0,k_0} \leftarrow w_{0,k_0} + Weight_0$
**while** $Tours < MaxTours$ **do** {Main loop}
  **if** $n = MaxLength$ **then** {Maximal length reached: don't grow}
    $Copy_n \leftarrow 0$
  **else** {pruning/enrichment by comparing with target weight}
    $Ratio \leftarrow Weight_n / W_{n,k_n}$
    $p \leftarrow Ratio \mod 1$
    Draw random number $r \in [0,1]$
    **if** $r < p$ **then**
      $Copy_n \leftarrow \lfloor Ratio \rfloor + 1$
    **else**
      $Copy_n \leftarrow \lfloor Ratio \rfloor$
    **end if**
    $Weight_n \leftarrow w_{n,k_n}$
  **end if**
  **if** $Copy_n = 0$ **then** {Shrink to last enrichment point or to size zero}
    **while** $n > 0$ and $Copy_n = 0$ **do**
      $n \leftarrow n - 1$
    **end while**
  **end if**
  **if** $n = 0$ and $Copy_0 = 0$ **then** {start new tour}
    $Tours \leftarrow Tours + 1$, $n \leftarrow 0$, $k_0 \leftarrow 0$, $Weight_0 \leftarrow 1$
    $s_{n,k_n} \leftarrow s_{n,k_n} + 1$, $w_{n,k_n} \leftarrow w_{n,k_n} + Weight_n$
  **else** {Grow by one step}
    $Copy_n \leftarrow Copy_n - 1$
    Draw random number $r \in [0,1]$
    **if** $r > 1/2$ **then**
      $k_{n+1} \leftarrow k_n + 1$
    **else**
      $k_{n+1} \leftarrow k_n$
    **end if**
    $Weight_{n+1} \leftarrow Weight_n$, $n \leftarrow n + 1$
    $s_{n,k_n} \leftarrow s_{n,k_n} + 1$, $w_{n,k_n} \leftarrow w_{n,k_n} + Weight_n$
  **end if**
**end while**

# Pruned and Enriched Sampling of Simple Random Walk

**repeat**
    **if** maximal lenght reached **then**
        set number of enrichment copies to zero
    **else**
        prune/enrich step: compute number of enrichment copies
    **end if**
    **if** number of enrichment copies is zero **then**
        prune: shrink to previous enrichment
    **end if**
    **if** configuration shrunk to zero **then**
        start new tour
        store data for new configuration
    **else**
        decrease number of enrichment copies
        grow new step
        store data for new configuration
    **end if**
**until** enough data is generated




Pruned and enriched sampling of simple random walk for $n = 50$ steps, with 100000 tours generated.

# Outline

- So far, we still needed the target weights to be known

- So far, we still needed the target weights to be known
- For a truly blind algorithm, need to also estimate target weights

- So far, we still needed the target weights to be known
- For a truly blind algorithm, need to also estimate target weights

**Key idea**

Compute target weights on the fly

- So far, we still needed the target weights to be known
- For a truly blind algorithm, need to also estimate target weights

**Key idea**

Compute target weights on the fly

Replace exact $W_{n,k}$ by estimate $\langle W_{n,k} \rangle$ generated from data

- So far, we still needed the target weights to be known
- For a truly blind algorithm, need to also estimate target weights

**Key idea**

Compute target weights on the fly

Replace exact $W_{n,k}$ by estimate $\langle W_{n,k}\rangle$ generated from data

- insert

  $W_{n,k_n} \leftarrow (n+1)w_{n,k_n}/\sum_l w_{n,l}$

  just before

  $Ratio \leftarrow Weight_n/W_{n,k_n}$

- So far, we still needed the target weights to be known
- For a truly blind algorithm, need to also estimate target weights

**Key idea**

Compute target weights on the fly

Replace exact $W_{n,k}$ by estimate $\langle W_{n,k}\rangle$ generated from data

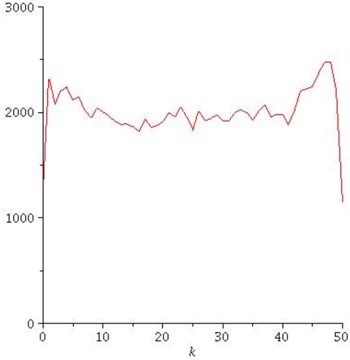
- insert

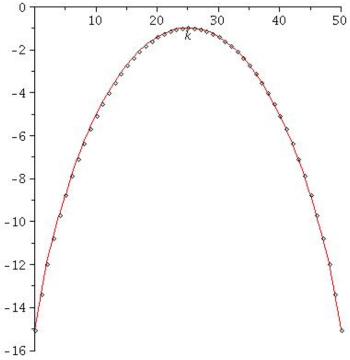
  $$W_{n,k_n} \leftarrow (n+1)w_{n,k_n}/\sum_l w_{n,l}$$

  just before

  $$Ratio \leftarrow Weight_n/W_{n,k_n}$$

That's all!




Blind pruned and enriched sampling of simple random walk for $n = 50$ steps, with 100000 tours generated.

# Algorithm Analysis Needed!

- Any fixed choice of the target weight $W_{n,k}$ gives an algorithm that samples correctly (just maybe not that well)

# Algorithm Analysis Needed!

- Any fixed choice of the target weight $W_{n,k}$ gives an algorithm that samples correctly (just maybe not that well)
- Replacing the optimal choice of $W_{n,k}$ by an estimate should converge to the optimal choice, hence lead to uniform sampling

# Algorithm Analysis Needed!

- Any fixed choice of the target weight $W_{n,k}$ gives an algorithm that samples correctly (just maybe not that well)
- Replacing the optimal choice of $W_{n,k}$ by an estimate should converge to the optimal choice, hence lead to uniform sampling
- This is confirmed by simulations

# Algorithm Analysis Needed!

- Any fixed choice of the target weight $W_{n,k}$ gives an algorithm that samples correctly (just maybe not that well)
- Replacing the optimal choice of $W_{n,k}$ by an estimate should converge to the optimal choice, hence lead to uniform sampling
- This is confirmed by simulations

The algorithm ought to be simple enough for a rigorous analysis

# Algorithm Analysis Needed!

- Any fixed choice of the target weight $W_{n,k}$ gives an algorithm that samples correctly (just maybe not that well)
- Replacing the optimal choice of $W_{n,k}$ by an estimate should converge to the optimal choice, hence lead to uniform sampling
- This is confirmed by simulations

The algorithm ought to be simple enough for a rigorous analysis

Anyone interested?

# Summary of First Lecture

Sampling of Random Walks

- Simple Sampling
- Biased Sampling
- Uniform Sampling
- Pruned and Enriched Sampling
- Blind Pruned and Enriched Sampling

# Summary of First Lecture

Sampling of Random Walks

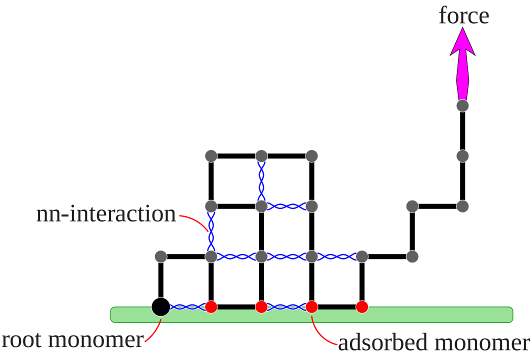
- Simple Sampling
- Biased Sampling
- Uniform Sampling
- Pruned and Enriched Sampling
- Blind Pruned and Enriched Sampling

Slightly unrealistic situation because

- Random walks never trap - no attrition!

# Outline

# "Realistic" Lattice Models of Polymers




A self-avoiding walk lattice model of a polymer tethered to a sticky surface under the influence of a pulling force.

# The Effect of Self-Avoidance

Physically,

- Excluded volume changes the universality class

[1] N Clisby, PRL **104** 055702 (2010), using the Pivot Algorithm

# The Effect of Self-Avoidance

Physically,

- Excluded volume changes the universality class
- Different critical exponents, e.g. length scale exponent changes

$$R \sim n^{\nu}$$

where $\nu = 0.5$ for RW and $\nu = 0.587597(7)\ldots$[1] for SAW in d=3

[1] N Clisby, PRL **104** 055702 (2010), using the Pivot Algorithm

# The Effect of Self-Avoidance

Physically,

- Excluded volume changes the universality class
- Different critical exponents, e.g. length scale exponent changes

$$R \sim n^{\nu}$$

where $\nu = 0.5$ for RW and $\nu = 0.587597(7)\ldots$[1] for SAW in d=3

and mathematically,

- Self-avoidance turns a simple Markovian random walk without memory into a complicated non-Markovian random walk with infinite memory

[1] N Clisby, PRL **104** 055702 (2010), using the Pivot Algorithm

# The Effect of Self-Avoidance

Physically,

- Excluded volume changes the universality class
- Different critical exponents, e.g. length scale exponent changes

$$R \sim n^{\nu}$$

where $\nu = 0.5$ for RW and $\nu = 0.587597(7)\ldots$[1] for SAW in d=3

and mathematically,

- Self-avoidance turns a simple Markovian random walk without memory into a complicated non-Markovian random walk with infinite memory
- When growing a self-avoiding walk, one needs to test for self-intersection with all previous steps

[1] N Clisby, PRL **104** 055702 (2010), using the Pivot Algorithm

# Outline

# Simple Sampling of Self-Avoiding Walk

From now on, consider Self-Avoiding Walks (SAW) on $\mathbb{Z}^2$

# Simple Sampling of Self-Avoiding Walk

From now on, consider Self-Avoiding Walks (SAW) on $\mathbb{Z}^2$

- Simple sampling of SAW works just like simple sampling of random walks

# Simple Sampling of Self-Avoiding Walk

From now on, consider Self-Avoiding Walks (SAW) on $\mathbb{Z}^2$

- Simple sampling of SAW works just like simple sampling of random walks
- But now walks get removed if they self-intersect

# Simple Sampling of Self-Avoiding Walk

$s_{.} \leftarrow 0$
$Samples \leftarrow 0$
**while** $Samples < MaxSamples$ **do**
  $Samples \leftarrow Samples + 1$
  $n \leftarrow 0$, Start at origin
  $s_0 \leftarrow s_0 + 1$
  **while** $n < MaxLength$ **do**
    Draw one of the neigboring sites uniformly at random
    **if** Occupied **then**
      Reject entire walk and exit loop
    **else**
      Step to new site
      $n \leftarrow n + 1$
      $s_n \leftarrow s_n + 1$
    **end if**
  **end while**
**end while**

# Properties of Simple Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step self-avoiding walk is grown with equal probability

# Properties of Simple Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step self-avoiding walk is grown with equal probability

However

Generating SAW with simple sampling is very inefficient

# Properties of Simple Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step self-avoiding walk is grown with equal probability

However

Generating SAW with simple sampling is very inefficient

- There are $4^n$ $n$-step random walks, but only about $2.638^n$ $n$-step SAW

# Properties of Simple Sampling

- Samples grown independently from scratch
- Each sample of an $n$-step self-avoiding walk is grown with equal probability

However

Generating SAW with simple sampling is very inefficient

- There are $4^n$ $n$-step random walks, but only about $2.638^n$ $n$-step SAW
- The probability of successfully generating an $n$-step SAW decreases exponentially fast

# Properties of Simple Sampling

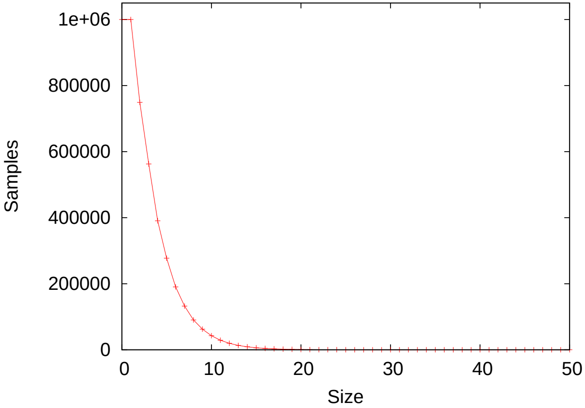
- Samples grown independently from scratch
- Each sample of an $n$-step self-avoiding walk is grown with equal probability

However

Generating SAW with simple sampling is very inefficient

- There are $4^n$ $n$-step random walks, but only about $2.638^n$ $n$-step SAW
- The probability of successfully generating an $n$-step SAW decreases exponentially fast

This is called exponential attrition




Attrition of started walks generated with Simple Sampling. From $10^6$ started walks none grew more than 35 steps.

# Outline

# Rosenbluth Sampling

A slightly improved sampling algorithm was proposed in 1955 by Rosenbluth and Rosenbluth[2].

[2] M. N. Rosenbluth and A. W. Rosenbluth, J. Chem. Phys. **23** 356 (1955)

# Rosenbluth Sampling

A slightly improved sampling algorithm was proposed in 1955 by Rosenbluth and Rosenbluth[2].

- Avoid self-intersections by only sampling from the steps that don't self-intersect

[2] M. N. Rosenbluth and A. W. Rosenbluth, J. Chem. Phys. **23** 356 (1955)

# Rosenbluth Sampling

A slightly improved sampling algorithm was proposed in 1955 by Rosenbluth and Rosenbluth[2].

- Avoid self-intersections by only sampling from the steps that don't self-intersect
- The growth only terminates if the walk is trapped and cannot continue growing

[2] M. N. Rosenbluth and A. W. Rosenbluth, J. Chem. Phys. **23** 356 (1955)

# Rosenbluth Sampling

A slightly improved sampling algorithm was proposed in 1955 by Rosenbluth and Rosenbluth[2].

- Avoid self-intersections by only sampling from the steps that don't self-intersect
- The growth only terminates if the walk is trapped and cannot continue growing

Still exponential attrition (albeit less)

[2] M. N. Rosenbluth and A. W. Rosenbluth, J. Chem. Phys. **23** 356 (1955)

# Rosenbluth Sampling

A slightly improved sampling algorithm was proposed in 1955 by Rosenbluth and Rosenbluth[2].

- Avoid self-intersections by only sampling from the steps that don't self-intersect
- The growth only terminates if the walk is trapped and cannot continue growing

Still exponential attrition (albeit less)

Configurations are generated with varying probabilities, depending on the number of ways they can be continued

[2] M. N. Rosenbluth and A. W. Rosenbluth, J. Chem. Phys. **23** 356 (1955)

# The "Atmosphere" of a configuration

We call *atmosphere* $a$ of a configuration the number of ways in which a configuration can continue to grow.

# The "Atmosphere" of a configuration

We call *atmosphere a* of a configuration the number of ways in which a configuration can continue to grow.

- For simple random walks, the atmosphere is constant

# The “Atmosphere” of a configuration

We call *atmosphere a* of a configuration the number of ways in which a configuration can continue to grow.

- For simple random walks, the atmosphere is constant
- For 2-dim SAW, the atmosphere varies between $a = 4$ (seed) and $a = 0$ (trapped)

# The "Atmosphere" of a configuration

We call *atmosphere* $a$ of a configuration the number of ways in which a configuration can continue to grow.

- For simple random walks, the atmosphere is constant
- For 2-dim SAW, the atmosphere varies between $a = 4$ (seed) and $a = 0$ (trapped)

If a configuration has atmosphere $a$ then there are $a$ different possibilities of growing the configuration, and each of these can get selected with probability $p = 1/a$, therefore the weight gets multiplied by $a$.

# The “Atmosphere” of a configuration

We call *atmosphere* $a$ of a configuration the number of ways in which a configuration can continue to grow.

- For simple random walks, the atmosphere is constant
- For 2-dim SAW, the atmosphere varies between $a = 4$ (seed) and $a = 0$ (trapped)

If a configuration has atmosphere $a$ then there are $a$ different possibilities of growing the configuration, and each of these can get selected with probability $p = 1/a$, therefore the weight gets multiplied by $a$.

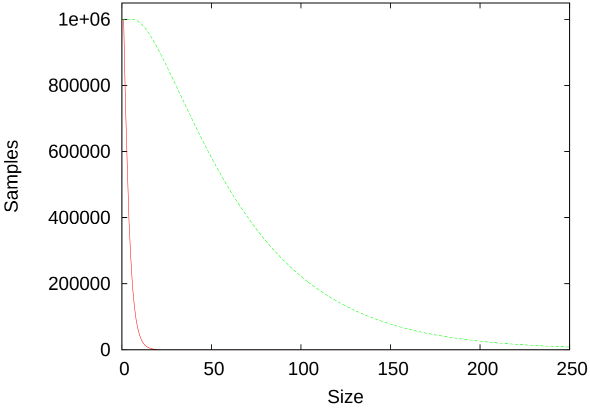
An $n$-step walk grown by Rosenbluth sampling therefore has weight

$$W_n = \prod_{i=0}^{n-1} a_i$$

and is generated with probability $P_n = 1/W_n$, so that $P_n W_n = 1$ as required.

# Rosenbluth Sampling of Self-Avoiding Walk

$Samples \leftarrow 0$
**while** $Samples < MaxSamples$ **do**
  $Samples \leftarrow Samples + 1$
  $n \leftarrow 0$, $Weight \leftarrow 1$, Start at origin
  $s_0 \leftarrow s_0 + 1$, $w_0 \leftarrow w_0 + Weight$
  **while** $n < MaxLength$ **do**
    Create list of neighboring unoccupied sites
    Determine the atmosphere $a$
    **if** $a = 0$ (walk cannot continue) **then**
      Reject entire walk and exit loop
    **else**
      Step to one of the unoccupied sites
      $n \leftarrow n + 1$, $Weight \leftarrow Weight \times a$
      $s_n \leftarrow s_n + 1$, $w_n \leftarrow w_n + Weight$
    **end if**
  **end while**
**end while**




Attrition of started walks generated with Rosenbluth Sampling compared with Simple Sampling.

# Outline

# Pruned and Enriched Rosenbluth Sampling

- No significant improvements for four decades

[3] P. Grassberger, Phys. Rev E **56** 3682 (1997)
[4] E. J. Janse van Rensburg, J. Phys. A **42** 323001 (2009)

# Pruned and Enriched Rosenbluth Sampling

- No significant improvements for four decades
- In 1997 Grassberger augmented Rosenbluth sampling with pruning and enrichment strategies[3]

[3] P. Grassberger, Phys. Rev E **56** 3682 (1997)
[4] E. J. Janse van Rensburg, J. Phys. A **42** 323001 (2009)

# Pruned and Enriched Rosenbluth Sampling

- No significant improvements for four decades
- In 1997 Grassberger augmented Rosenbluth sampling with pruning and enrichment strategies[3]
- Grassberger's Pruned and Enriched Rosenbluth Method (PERM) uses somewhat different strategies from those presented here

[3] P. Grassberger, Phys. Rev E **56** 3682 (1997)
[4] E. J. Janse van Rensburg, J. Phys. A **42** 323001 (2009)

# Pruned and Enriched Rosenbluth Sampling

- No significant improvements for four decades
- In 1997 Grassberger augmented Rosenbluth sampling with pruning and enrichment strategies[3]
- Grassberger's Pruned and Enriched Rosenbluth Method (PERM) uses somewhat different strategies from those presented here
- For details, and several enhancements of PERM see review paper[4]

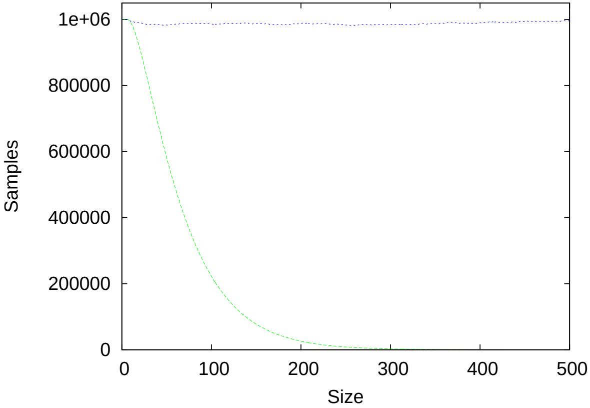
[3] P. Grassberger, Phys. Rev E **56** 3682 (1997)

[4] E. J. Janse van Rensburg, J. Phys. A **42** 323001 (2009)

# Pruned and Enriched Rosenbluth Sampling

**repeat**
  **if** zero atmosphere or maximal lenght reached **then**
    set number of enrichment copies to zero
  **else**
    prune/enrich step: compute number of enrichment copies
  **end if**
  **if** number of enrichment copies is zero **then**
    prune: shrink to previous enrichment
  **end if**
  **if** configuration shrunk to zero **then**
    start new tour and store data for new configuration
  **else**
    decrease number of enrichment copies
    **if** positive atmosphere **then**
      grow new step and store data for new configuration
    **end if**
  **end if**

$Tours \longleftarrow 0$, $n \longleftarrow 0$, $Weight_0 \longleftarrow 1$
Start new walk with step size zero
$a \longleftarrow 0$, $Copy_0 \longleftarrow 1$
$s_0 \longleftarrow s_0 + 1$, $w_0 \longleftarrow w_0 + Weight$
**while** $Tours < MaxTours$ **do** {Main loop}
  **if** $n = MaxLength$ or $a = 0$ **then** {Maximal length reached or atmosphere zero: don't grow}
    $Copy_n \longleftarrow 0$
  **else** {pruning/enrichment by comparing with target weight}
    $Ratio \longleftarrow Weight_n / w_n$
    $p \longleftarrow Ratio \mod 1$
    Draw random number $r \in [0, 1]$
    **if** $r < p$ **then**
      $Copy_n \longleftarrow \lfloor Ratio \rfloor + 1$
    **else**
      $Copy_n \longleftarrow \lfloor Ratio \rfloor$
    **end if**
    $Weight_n \longleftarrow w_n$
  **end if**
  **if** $Copy_n = 0$ **then** {Shrink to last enrichment point or to size zero}
    **while** $n > 0$ and $Copy_n = 0$ **do**
      Delete last site of walk
      $n \longleftarrow n - 1$
    **end while**
  **end if**
  **if** $n = 0$ and $Copy_0 = 0$ **then** {start new tour}
    $Tours \longleftarrow Tours + 1$,
    Start new walk with step size zero
    $a \longleftarrow 0$, $Copy_0 \longleftarrow 1$
    $s_0 \longleftarrow s_0 + 1$, $w_0 \longleftarrow w_0 + Weight$
  **else**
    Create list of neighboring unoccupied sites, determine the atmosphere $a$
    **if** $a > 0$ **then**
      $Copy_n \longleftarrow Copy_n - 1$
      Draw one of the neigboring unoccupied sites uniformly at random
      Step to new site
      $n \longleftarrow n + 1$, $Weight_n \longleftarrow Weight_n \times a$
      $s_n \longleftarrow s_n + 1$, $w_n \longleftarrow w_n + Weight_n$
    **end if**
  **end if**
**end while**




Attrition of started walks with PERM compared with Rosenbluth Sampling.

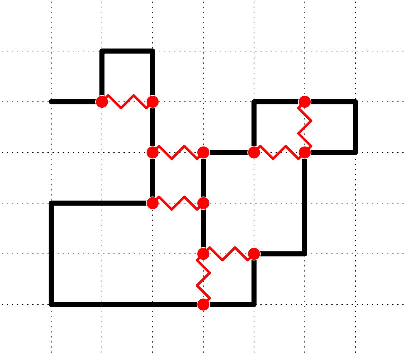
# Outline

# Interacting Self-Avoiding Walks

Consider sampling with respect to an extra parameter, for example the number of nearest-neighbour contacts

# Interacting Self-Avoiding Walks

Consider sampling with respect to an extra parameter, for example the number of nearest-neighbour contacts



An interacting self-avoiding walk on the square lattice with $n = 26$ steps and $m = 7$ contacts.

# Uniform Sampling

Renewed interest in Uniform Sampling Algorithms

F. Wang and D. P. Landau, PRL **86** 2050 (2001)

# Uniform Sampling

Renewed interest in Uniform Sampling Algorithms

F. Wang and D. P. Landau, PRL **86** 2050 (2001)

- Multicanonical PERM

  M. Bachmann and W. Janke, PRL **91** 208105 (2003)

# Uniform Sampling

Renewed interest in Uniform Sampling Algorithms

F. Wang and D. P. Landau, PRL **86** 2050 (2001)

- Multicanonical PERM

  M. Bachmann and W. Janke, PRL **91** 208105 (2003)
- Flat Histogram PERM

  T. Prellberg and J. Krawczyk, PRL **92** 120602 (2004)

# Uniform Sampling

Renewed interest in Uniform Sampling Algorithms

F. Wang and D. P. Landau, PRL **86** 2050 (2001)

- Multicanonical PERM

  M. Bachmann and W. Janke, PRL **91** 208105 (2003)
- Flat Histogram PERM

  T. Prellberg and J. Krawczyk, PRL **92** 120602 (2004)

Incorporating uniform sampling into PERM is straightforward, once one observes that PERM already samples uniformly *in system size*

# Flat Histogram Rosenbluth Sampling

Extension of PERM to a microcanonical version

# Flat Histogram Rosenbluth Sampling

Extension of PERM to a microcanonical version

- Distinguish configurations of size $n$ by some additional parameter $m$ (e.g. energy)

# Flat Histogram Rosenbluth Sampling

Extension of PERM to a microcanonical version

- Distinguish configurations of size $n$ by some additional parameter $m$ (e.g. energy)
- Bin data with respect to $n$ and $m$

  $s_{n,m} \leftarrow s_{n,m} + 1,\ w_{n,m} \leftarrow w_{n,m} + Weight_n$

# Flat Histogram Rosenbluth Sampling

Extension of PERM to a microcanonical version

- Distinguish configurations of size $n$ by some additional parameter $m$ (e.g. energy)
- Bin data with respect to $n$ and $m$

  $$s_{n,m} \leftarrow s_{n,m} + 1,\; w_{n,m} \leftarrow w_{n,m} + Weight_n$$

- Enrichment ratio for pruning/enrichment becomes

  $$Ratio \leftarrow Weight_n / W_{n,m}$$

# Flat Histogram Rosenbluth Sampling

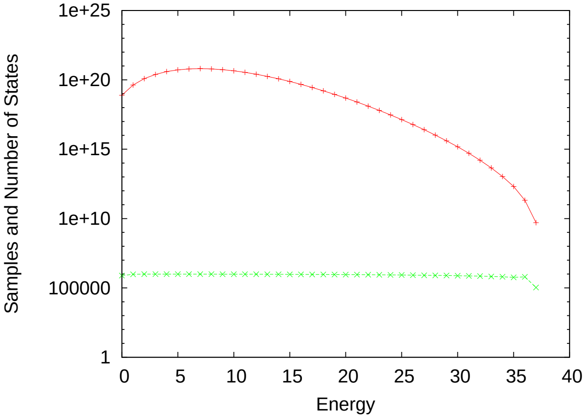
Extension of PERM to a microcanonical version

- Distinguish configurations of size $n$ by some additional parameter $m$ (e.g. energy)
- Bin data with respect to $n$ and $m$

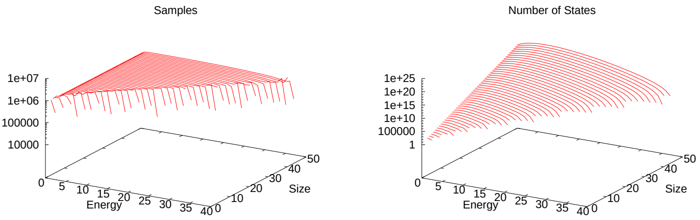
  $s_{n,m} \leftarrow s_{n,m} + 1, \; w_{n,m} \leftarrow w_{n,m} + Weight_n$
- Enrichment ratio for pruning/enrichment becomes

  $Ratio \leftarrow Weight_n / W_{n,m}$

And that is all!




Generated samples and estimated number of states for ISAW with 50 steps estimated from $10^6$ flatPERM tours.




Generated samples and estimated number of states for ISAW with up to 50 steps generated with flatPERM.

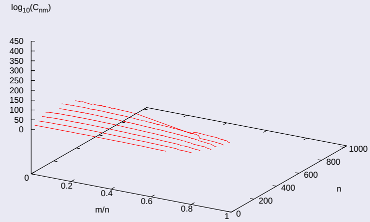
# Outline

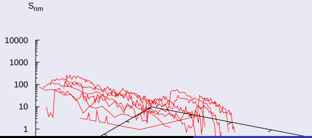
# 2d ISAW simulation up to $N = 1024$

To stabilise algorithm (avoid initial overflow/underflow):
Delay growth of large configurations
Here: after $t$ tours growth up to length $10t$

# 2d ISAW simulation up to $N = 1024$

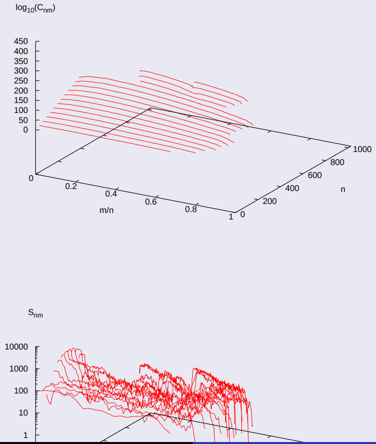
## Total sample size: $1,000,000$



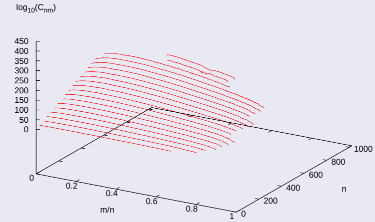
# 2d ISAW simulation up to $N = 1024$

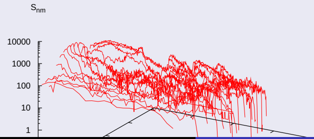
## Total sample size: $10,000,000$



# 2d ISAW simulation up to $N = 1024$

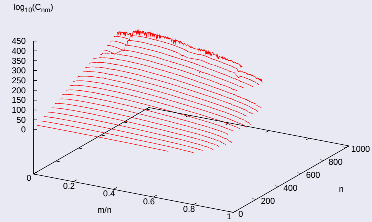
## Total sample size: $20,000,000$

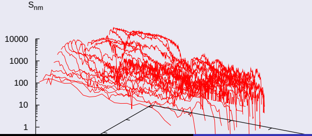


# 2d ISAW simulation up to $N = 1024$

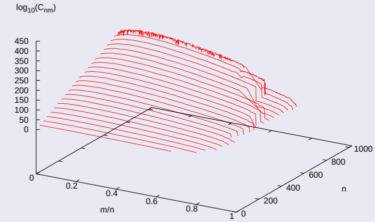
## Total sample size: $30,000,000$

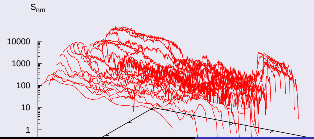


# 2d ISAW simulation up to $N = 1024$

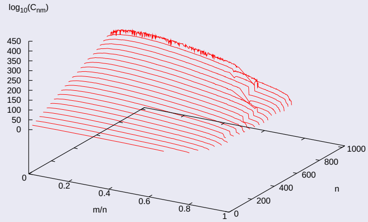
## Total sample size: $40,000,000$

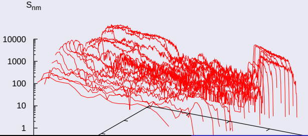


# 2d ISAW simulation up to $N = 1024$

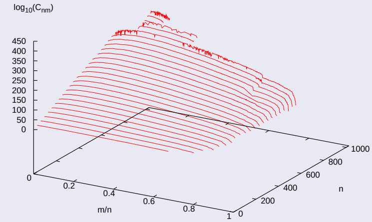
## Total sample size: $50,000,000$

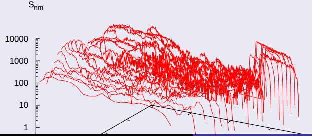


# 2d ISAW simulation up to $N = 1024$

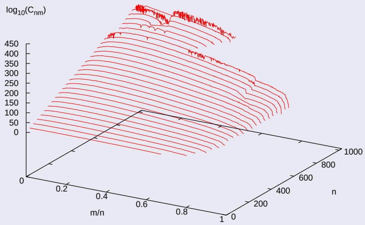
## Total sample size: $60,000,000$

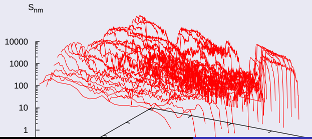


# 2d ISAW simulation up to $N = 1024$

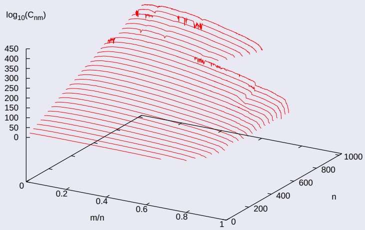
## Total sample size: $70,000,000$

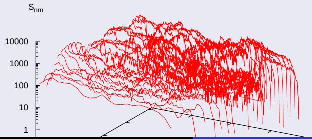


# 2d ISAW simulation up to $N = 1024$

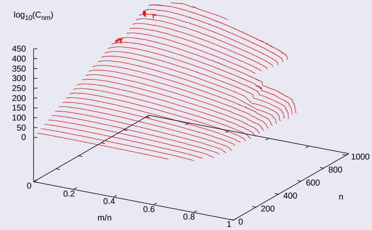
## Total sample size: $80,000,000$

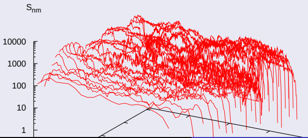


# 2d ISAW simulation up to $N = 1024$

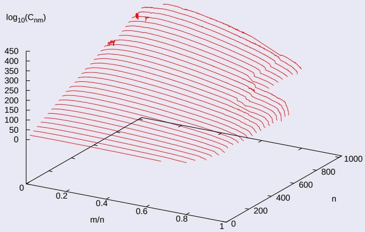
## Total sample size: $90,000,000$

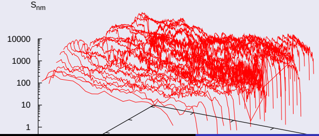


# 2d ISAW simulation up to $N = 1024$

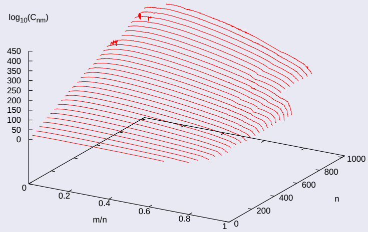
## Total sample size: $100,000,000$

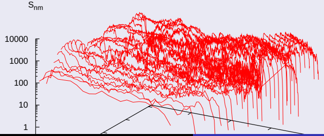


# 2d ISAW simulation up to $N = 1024$

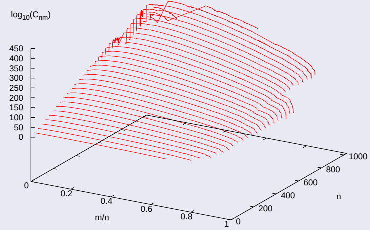
## Total sample size: $110,000,000$

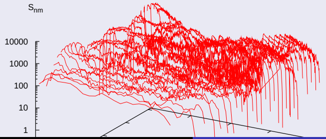


# 2d ISAW simulation up to $N = 1024$

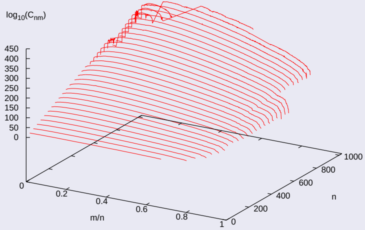
## Total sample size: $120,000,000$

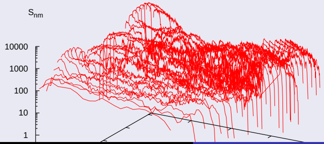


# 2d ISAW simulation up to $N = 1024$

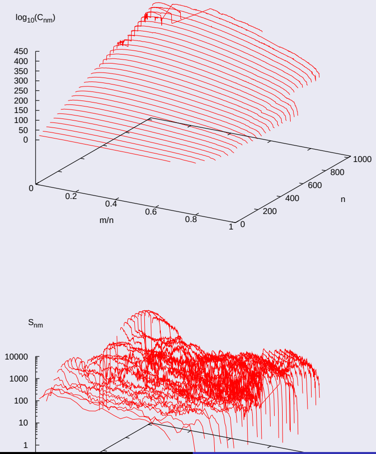
## Total sample size: $130,000,000$



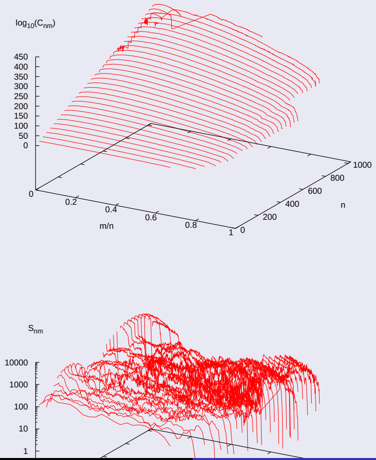
# 2d ISAW simulation up to $N = 1024$

## Total sample size: $140,000,000$



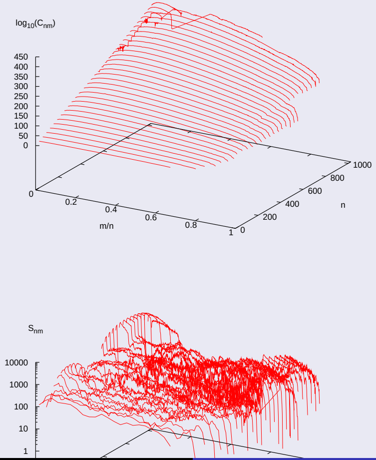
# 2d ISAW simulation up to $N = 1024$

## Total sample size: $150,000,000$



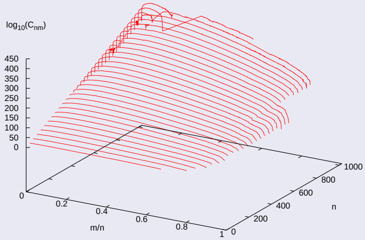
# 2d ISAW simulation up to $N = 1024$

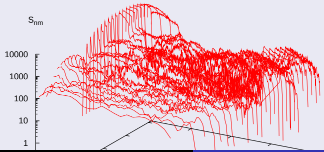
## Total sample size: $160,000,000$



# 2d ISAW simulation up to $N = 1024$

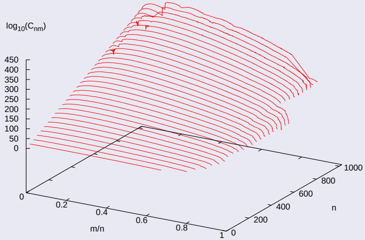
## Total sample size: $170,000,000$

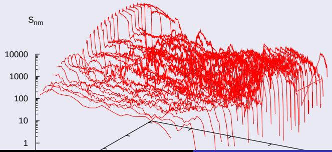


# 2d ISAW simulation up to $N = 1024$

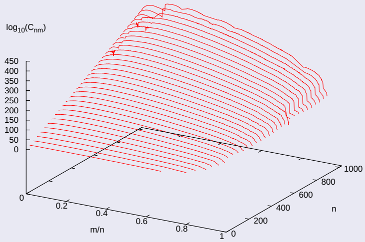
## Total sample size: $180,000,000$

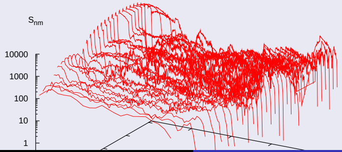


# 2d ISAW simulation up to $N = 1024$

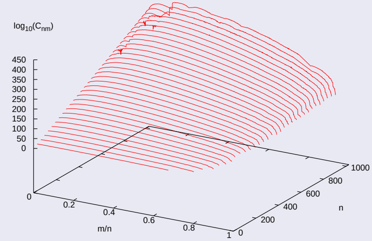
## Total sample size: $190,000,000$

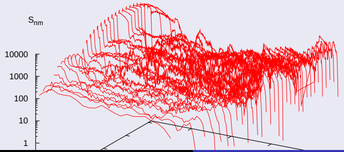


# 2d ISAW simulation up to $N = 1024$

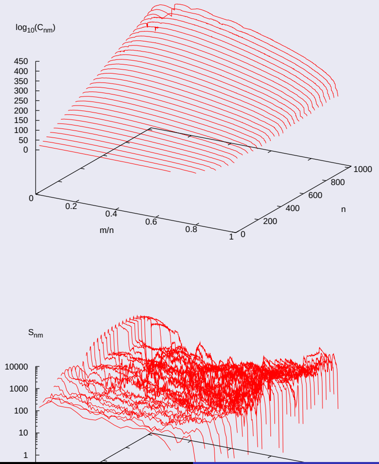
## Total sample size: $200,000,000$



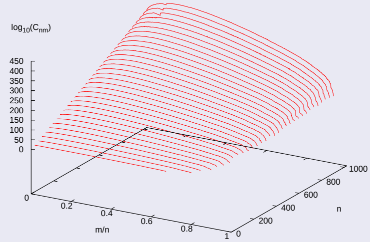
# 2d ISAW simulation up to $N = 1024$

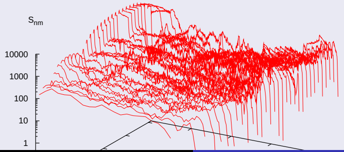
## Total sample size: $210,000,000$



# 2d ISAW simulation up to $N = 1024$

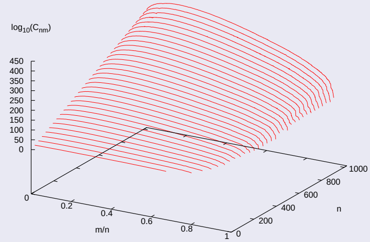
## Total sample size: $220,000,000$

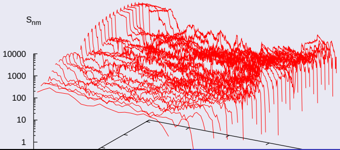


# 2d ISAW simulation up to $N = 1024$

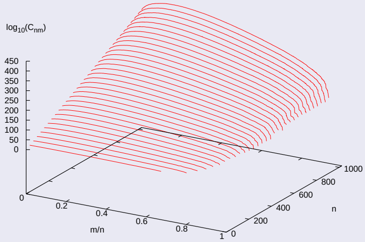
## Total sample size: $230,000,000$

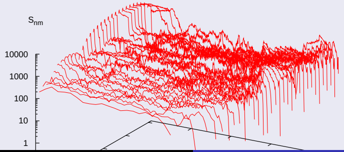


# 2d ISAW simulation up to $N = 1024$

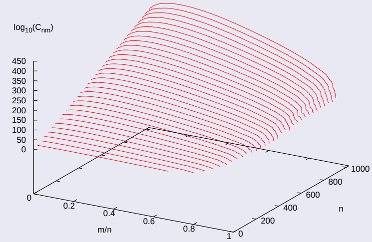
## Total sample size: $240,000,000$

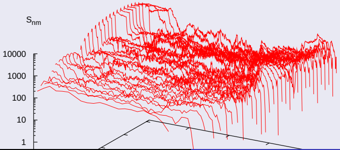


# 2d ISAW simulation up to $N = 1024$

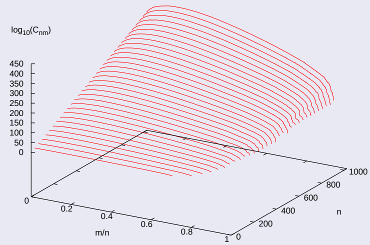
## Total sample size: $250,000,000$

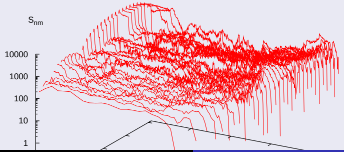


# 2d ISAW simulation up to $N = 1024$

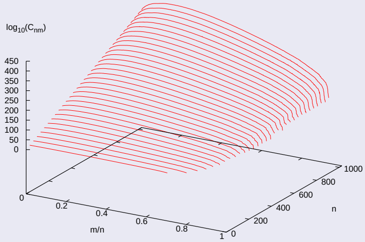
## Total sample size: $260,000,000$

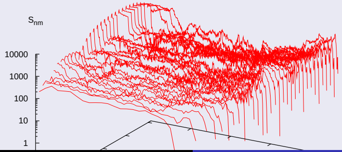


# 2d ISAW simulation up to $N = 1024$

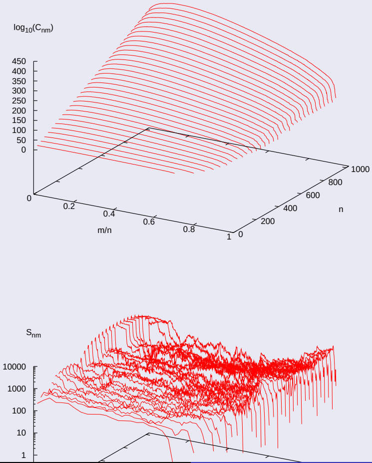
## Total sample size: $270,000,000$



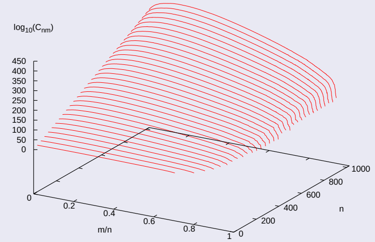
# 2d ISAW simulation up to $N = 1024$

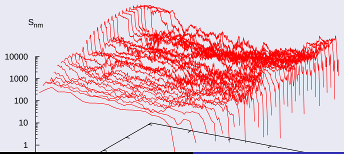
## Total sample size: $280,000,000$



# 2d ISAW simulation up to $N = 1024$

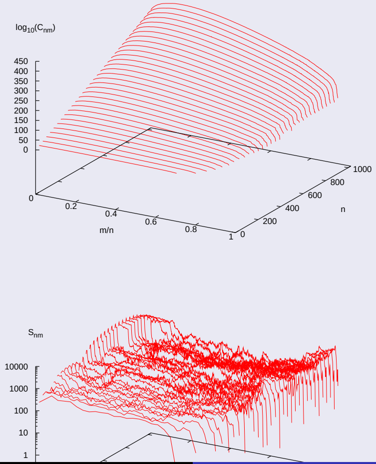
## Total sample size: $290,000,000$



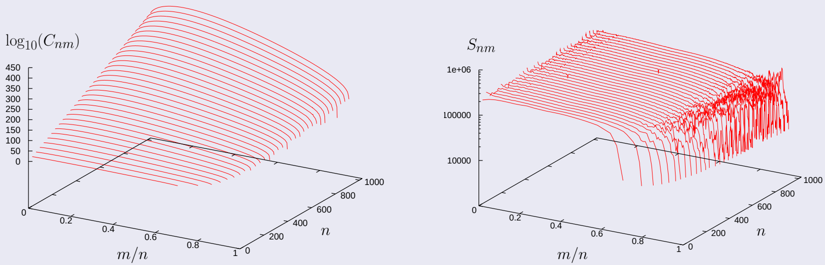
# 2d ISAW simulation up to $N = 1024$

## Total sample size: $300,000,000$



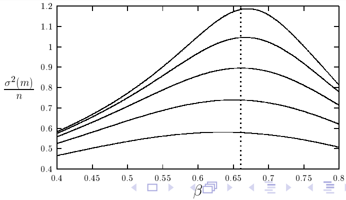
# ISAW simulations

T Prellberg and J Krawczyk, PRL 92 (2004) 120602




- 2d ISAW up to $n = 1024$
- One simulation suffices
- 400 orders of magnitude

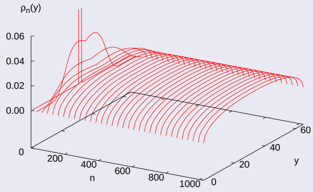
(only 2d shown, 3d similar)

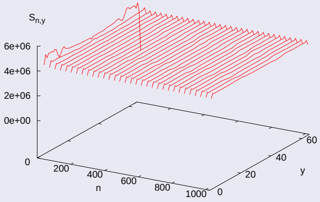


# Simulation results: SAW in a strip

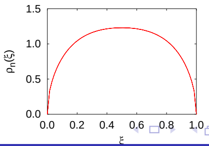
T Prellberg et al, in: Computer Simulation Studies in Condensed Matter Physics XVII, Springer Verlag, 2006

- 2d SAW in a strip: strip width 64, up to $n = 1024$



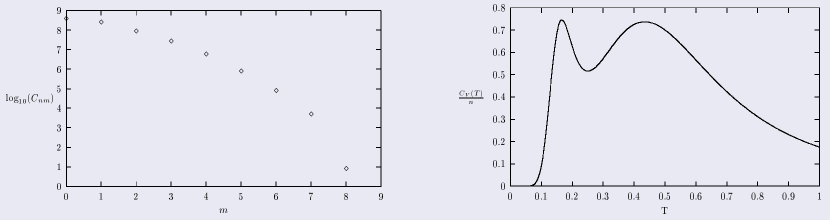

- Scaled endpoint density



# HP model simulations

T Prellberg et al, in: Computer Simulation Studies in Condensed Matter Physics XVII, Springer Verlag, 2006

- Engineered sequence HPHPHHPHPHHPPH in $d = 3$:




- Investigated other sequences up to $N \approx 100$ in $d = 2$ and $d = 3$
- Collapsed regime accessible
- Reproduced known ground state energies
- Obtained density of states $C_{n,m}$ over large range ($\approx 10^{30}$)

# Outline

# Extensions

PERM (and its flat histogram version) can be applied objects that are grown in a unique way

- prime example: linear polymers

# Extensions

PERM (and its flat histogram version) can be applied objects that are grown in a unique way

- prime example: linear polymers
- but also: permutations (insert $n+1$ into a permutation of $\{1, 2, \ldots, n\}$)

# Extensions

PERM (and its flat histogram version) can be applied objects that are grown in a unique way

- prime example: linear polymers
- but also: permutations (insert $n+1$ into a permutation of $\{1, 2, \ldots, n\}$)

What about objects that can be grown in different, not necessarily unique, ways?

# Extensions

PERM (and its flat histogram version) can be applied objects that are grown in a unique way

- prime example: linear polymers
- but also: permutations (insert $n+1$ into a permutation of $\{1, 2, \ldots, n\}$)

What about objects that can be grown in different, not necessarily unique, ways?

- examples: ring polymers, branched polymers (lattice trees)

# Outline

# Positive and Negative Atmospheres

A. Rechnitzer and E. J. Janse van Rensburg, J. Phys. A **41** 442002 (2008)

## Key idea

Introduce an additional negative atmosphere $a^-$ indicating in how many ways a configuration can be reduced in size

# Positive and Negative Atmospheres

A. Rechnitzer and E. J. Janse van Rensburg, J. Phys. A **41** 442002 (2008)

**Key idea**

Introduce an additional negative atmosphere $a^{-}$ indicating in how many ways a configuration can be reduced in size

- For linear polymers the negative atmosphere is always unity, as there is only one way to remove a step

# Positive and Negative Atmospheres

A. Rechnitzer and E. J. Janse van Rensburg, J. Phys. A **41** 442002 (2008)

**Key idea**

Introduce an additional negative atmosphere $a^{-}$ indicating in how many ways a configuration can be reduced in size

- For linear polymers the negative atmosphere is always unity, as there is only one way to remove a step
- For lattice trees the negative atmosphere is equal to the number of its leaves

# Generalized Atmospheric Rosenbluth Sampling

A new algorithm: Generalized Atmospheric Rosenbluth Method (GARM)

# Generalized Atmospheric Rosenbluth Sampling

A new algorithm: Generalized Atmospheric Rosenbluth Method (GARM)

- An $n$-step configuration grown has weight

$$W_n = \prod_{i=0}^{n-1} \frac{a_i}{a^-_{i+1}} , \qquad (1)$$

where $a_i$ are the (positive) atmospheres of the configuration after $i$ growth steps, and $a^-_i$ are the negative atmospheres of the configuration after $i$ growth steps.

# Generalized Atmospheric Rosenbluth Sampling

A new algorithm: Generalized Atmospheric Rosenbluth Method (GARM)

- An $n$-step configuration grown has weight

$$W_n = \prod_{i=0}^{n-1} \frac{a_i}{a^-_{i+1}} \,, \qquad (1)$$

  where $a_i$ are the (positive) atmospheres of the configuration after $i$ growth steps, and $a^-_i$ are the negative atmospheres of the configuration after $i$ growth steps.
- The probability of growing this configuration is $P_n = 1/W_n$, so again $P_n W_n = 1$ holds as required.

# Generalized Atmospheric Rosenbluth Sampling

**while** $Samples < MaxSamples$ **do**
  $Samples \leftarrow Samples + 1$
  $n \leftarrow 0$, $Weight \leftarrow 1$, Start with seed configuration
  $s_0 \leftarrow s_0 + 1$, $w_0 \leftarrow w_0 + Weight$
  **while** $n < MaxSize$ **do**
    Create list of growth possibilities, determine the atmosphere $a$
    **if** $a = 0$ (no growth possible) **then**
      Reject entire configuration and exit loop
    **else**
      Draw one of the growth possibilities uniformly at random
      Grow configuration
      $n \leftarrow n + 1$, $Weight \leftarrow Weight \times a$
      Compute negative atmosphere $a^-$
      $Weight \leftarrow Weight / a^-$
      $s_n \leftarrow s_n + 1$, $w_n \leftarrow w_n + Weight$
    **end if**
  **end while**
**end while**

# Generalized Atmospheric Rosenbluth Sampling

- Implementing GARM is straightforward
- Computation of atmospheres might be expensive

# Generalized Atmospheric Rosenbluth Sampling

- Implementing GARM is straightforward
- Computation of atmospheres might be expensive
- Can add pruning/enrichment
- Can extend to flat histogram sampling

# Generalized Atmospheric Rosenbluth Sampling

- Implementing GARM is straightforward
- Computation of atmospheres might be expensive
- Can add pruning/enrichment
- Can extend to flat histogram sampling

Other developments

- add moves that don't change the system size ("neutral" atmosphere)

# Generalized Atmospheric Rosenbluth Sampling

- Implementing GARM is straightforward
- Computation of atmospheres might be expensive
- Can add pruning/enrichment
- Can extend to flat histogram sampling

Other developments

- add moves that don't change the system size ("neutral" atmosphere)
- grow and shrink independently (Generalized Atmospheric Sampling, GAS)

# Generalized Atmospheric Rosenbluth Sampling

- Implementing GARM is straightforward
- Computation of atmospheres might be expensive
- Can add pruning/enrichment
- Can extend to flat histogram sampling

Other developments

- add moves that don't change the system size ("neutral" atmosphere)
- grow and shrink independently (Generalized Atmospheric Sampling, GAS)

For further extensions to Rosenbluth sampling, and indeed many more algorithms for simulating self-avoiding walks, as well as applications, see the review "Monte Carlo methods for the self-avoiding walk," E. J. Janse van Rensburg, J. Phys. A **42** 323001 (2009)