Worm algorithm for loop model on the square lattice

Wenan Guo

Beijing Normal University

July, 2010, Melbourne

in collaboration with

Youjin Deng, USTC, China

Chengxiang Ding, BNU, China

Henk W.J. Blöte, Lorentz Institute, The Netherlands

outlines

Motivations

- There are exact results for a number of two-dimensional O(n) loop models. But, these models form only a relatively small subset. It is useful to develop numerical approaches to investigate O(n) loop models in a more general context.

Motivations

- There are exact results for a number of two-dimensional O(n) loop models. But, these models form only a relatively small subset. It is useful to develop numerical approaches to investigate O(n) loop models in a more general context.
- Transfer-matrix calculations are restricted to relatively small sizes, and are able to generate satisfactory results only if the corrections to scaling are not too large.

Motivations

- There are exact results for a number of two-dimensional O(n) loop models. But, these models form only a relatively small subset. It is useful to develop numerical approaches to investigate O(n) loop models in a more general context.
- Transfer-matrix calculations are restricted to relatively small sizes, and are able to generate satisfactory results only if the corrections to scaling are not too large.
- There exists a class of intersecting loop models displaying extremely slow finite-size convergence. (Martins, Nienhuis and Rietman, PRL 1998, Martins and Nienhuis, JPA 1998, de Gier and Nienhuis, JSTAT, 2005.)

Motivations

- There are exact results for a number of two-dimensional O(n) loop models. But, these models form only a relatively small subset. It is useful to develop numerical approaches to investigate O(n) loop models in a more general context.
- Transfer-matrix calculations are restricted to relatively small sizes, and are able to generate satisfactory results only if the corrections to scaling are not too large.
- There exists a class of intersecting loop models displaying extremely slow finite-size convergence. (Martins, Nienhuis and Rietman, PRL 1998, Martins and Nienhuis, JPA 1998, de Gier and Nienhuis, JSTAT, 2005.)
  When crossings of loops are allowed, the low-temperature phase is distinct from nonintersecting loop models. (Jacobsen *et al*, PRL 2003). And the LT branch of the nonintersecting loop model can be mapped onto a tricritical loop model with a different loop weight. (Nienhuis, WG and Blöte, PRE, 2009).

Motivations

- so that Monte Carlo simulation seems a good realistic option to obtain satisfactory numerical results for the intersecting loop models.

Motivations

- so that Monte Carlo simulation seems a good realistic option to obtain satisfactory numerical results for the intersecting loop models.
- An efficient Monte Carlo algorithm of the cluster type is available for 2D nonintersecting loop models (Deng *et al*, PRL 2007). Thus far, no efficient Monte Carlo algorithm to simulate intersecting loop models.

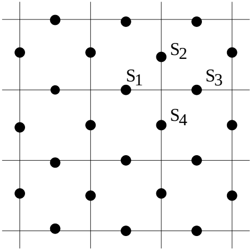
O(n) spin model on the square lattice

Put spins on the middle of the edges of the square lattice

$$\mathcal{Z} = \int \left[\prod_i d\vec{s_i}\right] \prod_V \{1 + u\,(\vec{s_1}\cdot\vec{s_2} + \vec{s_2}\cdot\vec{s_3} + \vec{s_3}\cdot\vec{s_4} + \vec{s_4}\cdot\vec{s_1}) +$$

$$v\,(\vec{s_1}\cdot\vec{s_3} + \vec{s_2}\cdot\vec{s_4}) + w\,[(\vec{s_1}\cdot\vec{s_2})(\vec{s_3}\cdot\vec{s_4}) + (\vec{s_2}\cdot\vec{s_3})(\vec{s_4}\cdot\vec{s_1})] + c(\vec{s_1}\cdot\vec{s_3})(\vec{s_2}\cdot\vec{s_4})\}$$

$\vec{s_i}$: n-component vector spin, the weight is O(n) symmetric



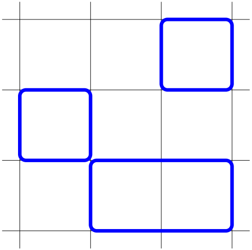
O(n) spin model on the square lattice

Put spins on the middle of the edges of the square lattice

$$\mathcal{Z} = \int \left[\prod_i d\vec{s_i}\right] \prod_V \{1 + u\,(\vec{s_1}\cdot\vec{s_2} + \vec{s_2}\cdot\vec{s_3} + \vec{s_3}\cdot\vec{s_4} + \vec{s_4}\cdot\vec{s_1}) +$$

$$v\,(\vec{s_1}\cdot\vec{s_3} + \vec{s_2}\cdot\vec{s_4}) + w\,[(\vec{s_1}\cdot\vec{s_2})(\vec{s_3}\cdot\vec{s_4}) + (\vec{s_2}\cdot\vec{s_3})(\vec{s_4}\cdot\vec{s_1})] + c(\vec{s_1}\cdot\vec{s_3})(\vec{s_2}\cdot\vec{s_4})\}$$

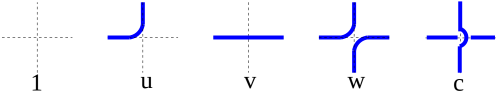
$\vec{s_i}$: n-component vector spin, the weight is O(n) symmetric



Expansion of $\mathcal{Z}$ in powers of u, v, w, c yields an O(n) (intersecting) loop model

$$\mathcal{Z} = \sum_{\mathcal{A}} n^l \prod_{i\in V} \omega_i = \sum_{\mathcal{A}} n^l u^{N_u} v^{N_v} w^{N_w} c^{N_c}$$

$u^{10}v^2w^1n^3$



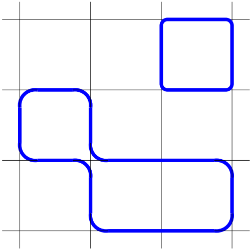
O(n) spin model on the square lattice

Put spins on the middle of the edges of the square lattice

$$\mathcal{Z} = \int \left[\prod_i d\vec{s_i}\right] \prod_V \{1 + u\,(\vec{s_1}\cdot\vec{s_2} + \vec{s_2}\cdot\vec{s_3} + \vec{s_3}\cdot\vec{s_4} + \vec{s_4}\cdot\vec{s_1}) +$$

$$v\,(\vec{s_1}\cdot\vec{s_3} + \vec{s_2}\cdot\vec{s_4}) + w\,[(\vec{s_1}\cdot\vec{s_2})(\vec{s_3}\cdot\vec{s_4}) + (\vec{s_2}\cdot\vec{s_3})(\vec{s_4}\cdot\vec{s_1})] + c(\vec{s_1}\cdot\vec{s_3})(\vec{s_2}\cdot\vec{s_4})\}$$

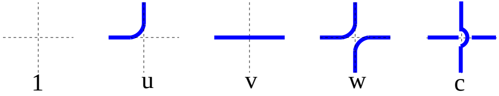
$\vec{s_i}$: n-component vector spin, the weight is O(n) symmetric



Expansion of $\mathcal{Z}$ in powers of u, v, w, c yields an O(n) (intersecting) loop model

$$\mathcal{Z} = \sum_{\mathcal{A}} n^l \prod_{i\in V} \omega_i = \sum_{\mathcal{A}} n^l u^{N_u} v^{N_v} w^{N_w} c^{N_c}$$

$u^{10} v^2 w^1 n^2$



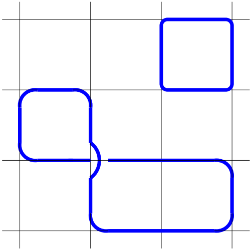
O(n) spin model on the square lattice

Put spins on the middle of the edges of the square lattice

$$\mathcal{Z} = \int \left[\prod_i d\vec{s_i}\right] \prod_V \{1 + u\,(\vec{s_1}\cdot\vec{s_2} + \vec{s_2}\cdot\vec{s_3} + \vec{s_3}\cdot\vec{s_4} + \vec{s_4}\cdot\vec{s_1}) +$$

$$v\,(\vec{s_1}\cdot\vec{s_3} + \vec{s_2}\cdot\vec{s_4}) + w\,[(\vec{s_1}\cdot\vec{s_2})(\vec{s_3}\cdot\vec{s_4}) + (\vec{s_2}\cdot\vec{s_3})(\vec{s_4}\cdot\vec{s_1})] + c(\vec{s_1}\cdot\vec{s_3})(\vec{s_2}\cdot\vec{s_4})\}$$

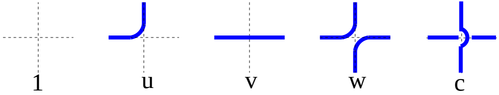
$\vec{s_i}$: n-component vector spin, the weight is O(n) symmetric



Expansion of $\mathcal{Z}$ in powers of u, v, w, c yields an O(n) (intersecting) loop model

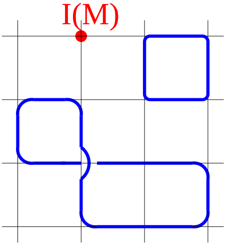
$$\mathcal{Z} = \sum_{\mathcal{A}} n^l \prod_{i\in V} \omega_i = \sum_{\mathcal{A}} n^l u^{N_u} v^{N_v} w^{N_w} c^{N_c}$$

$u^{10}v^2c^1n^2$



Worm algorithm

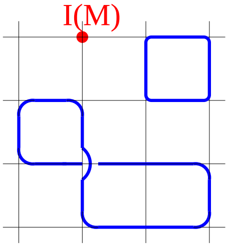
Enlarge the set of configurations of the loop model by including two "special" vertices I and M.



Worm algorithm

Enlarge the set of configurations of the loop model by including two "special" vertices I and M.

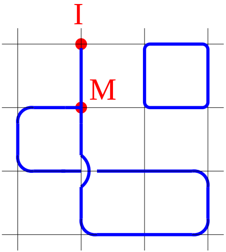
- If $I = M$, all vertices have an even number of incident occupied bonds.



Worm algorithm

Enlarge the set of configurations of the loop model by including two "special" vertices I and M.

- If $I = M$, all vertices have an even number of incident occupied bonds.
- if $I \neq M$ the vertices I and M have an *odd* number of incident occupied bonds.



Worm algorithm

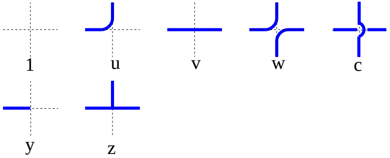
Enlarge the set of configurations of the loop model by including two "special" vertices I and M.

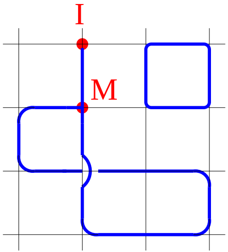
- If $I = M$, all vertices have an even number of incident occupied bonds.
- if $I \neq M$ the vertices I and M have an *odd* number of incident occupied bonds.

Let $\mathcal{S}$ be a state in the enlarged space

$$\phi_{\mathcal{S}} = n^{l} \cdot \prod_{i \in V} \omega_i \ ,$$

ω_i can be 1, u, v, w, c, and y, z





$$\mathcal{Z}_{\mathrm{w}} = \sum_{\mathcal{S}} \phi_{\mathcal{S}} \ .$$

The Algorithm

1. Randomly choose a vertex $k \in V$ and move $I = M$ to k. The configuration of occupied bonds remains unchanged.

The Algorithm

1. Randomly choose a vertex $k \in V$ and move $I = M$ to k. The configuration of occupied bonds remains unchanged.
2. Let the current state be $\mathcal{S}$. Randomly choose I or M– say M; randomly choose one M' of the 4 nearest-neighbor vertices of M. Propose a move $M \to M'$ while inverting the edge e between M and M' as $e = 0 \leftrightarrow e = 1$.

The Algorithm

1. Randomly choose a vertex $k \in V$ and move $I = M$ to k. The configuration of occupied bonds remains unchanged.
2. Let the current state be $\mathcal{S}$. Randomly choose I or M– say M; randomly choose one M' of the 4 nearest-neighbor vertices of M. Propose a move $M \to M'$ while inverting the edge e between M and M' as $e = 0 \leftrightarrow e = 1$.
3. Randomly select one of the possible states $\mathcal{S}'$ by taking into account all possible ways to connect the incoming occupied bonds of M and M' after the proposed bond update.

The Algorithm

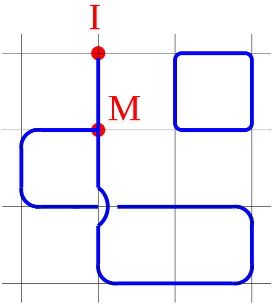
1. Randomly choose a vertex $k \in V$ and move $I = M$ to k. The configuration of occupied bonds remains unchanged.
2. Let the current state be $\mathcal{S}$. Randomly choose I or M– say M; randomly choose one M' of the 4 nearest-neighbor vertices of M. Propose a move $M \to M'$ while inverting the edge e between M and M' as $e = 0 \leftrightarrow e = 1$.
3. Randomly select one of the possible states $\mathcal{S}'$ by taking into account all possible ways to connect the incoming occupied bonds of M and M' after the proposed bond update.
4. Accept the proposed update $\mathcal{S} \to \mathcal{S}'$ with appropriate acceptance probability $P_a(\mathcal{S} \to \mathcal{S}')$.

The Algorithm

1. Randomly choose a vertex $k \in V$ and move $I = M$ to k. The configuration of occupied bonds remains unchanged.
2. Let the current state be $\mathcal{S}$. Randomly choose I or M– say M; randomly choose one M' of the 4 nearest-neighbor vertices of M. Propose a move $M \to M'$ while inverting the edge e between M and M' as $e = 0 \leftrightarrow e = 1$.
3. Randomly select one of the possible states $\mathcal{S}'$ by taking into account all possible ways to connect the incoming occupied bonds of M and M' after the proposed bond update.
4. Accept the proposed update $\mathcal{S} \to \mathcal{S}'$ with appropriate acceptance probability $P_a(\mathcal{S} \to \mathcal{S}')$.
5. Relabel M' as M. If $M = I$, then goto 1; else go to 2.

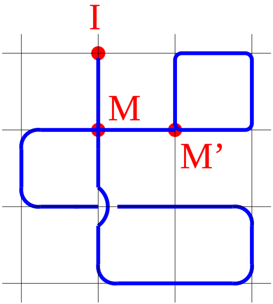
Acceptance probability

step 2, current state $\mathcal{S}$



Acceptance probability

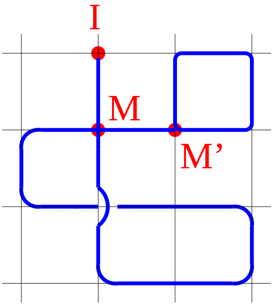
step 2, a test move



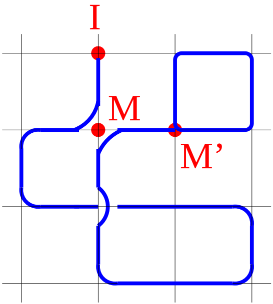
Acceptance probability

step 2, a test move

A given bond configuration may correspond with different loop configurations.



Acceptance probability

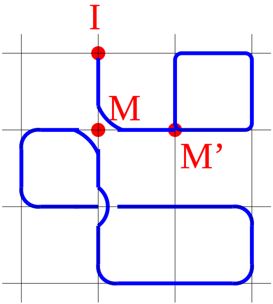


step 2, a test move

A given bond configuration may correspond with different loop configurations.

step 3, select a state $\mathcal{S}'$

Acceptance probability



step 2, a test move

A given bond configuration may correspond with different loop configurations.

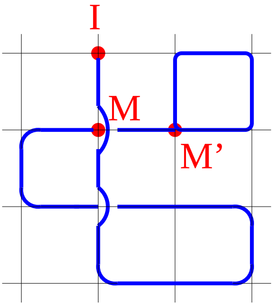
step 3, or a state $\mathcal{S}'$

Acceptance probability

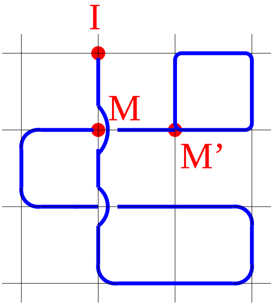
step 2, a test move

A given bond configuration may correspond with different loop configurations.

step 3, or another $\mathcal{S}'$



Acceptance probability



step 2, a test move

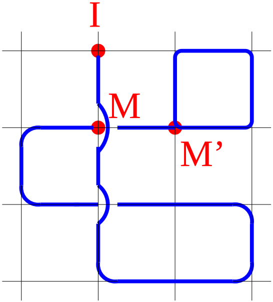
A given bond configuration may correspond with different loop configurations.

step 3, or another $\mathcal{S}'$

The acceptance probability

$$P_a(\mathcal{S} \to \mathcal{S}') = \min\left(1, \frac{p_p(\mathcal{S}|\mathcal{S}')}{p_p(\mathcal{S}'|\mathcal{S})} \cdot \frac{\phi_{\mathcal{S}'}}{\phi_{\mathcal{S}}}\right) ,$$

Acceptance probability



step 2, a test move
A given bond configuration may
correspond with different loop configurations.
step 3, or another $\mathcal{S}'$

The acceptance probability

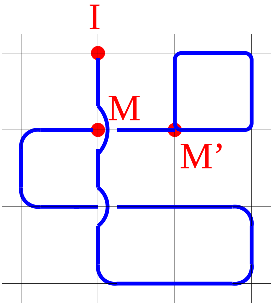
$$P_a(\mathcal{S} \to \mathcal{S}') = \min\left(1, \frac{p_p(\mathcal{S}|\mathcal{S}')}{p_p(\mathcal{S}'|\mathcal{S})} \cdot \frac{\phi_{\mathcal{S}'}}{\phi_{\mathcal{S}}}\right) ,$$

$p_p(\mathcal{S}'|\mathcal{S})$ the proposal probability from $\mathcal{S}$ to $\mathcal{S}'$:

$$p_p(\mathcal{S}'|\mathcal{S}) = \frac{1}{8} \cdot \frac{1}{d_M(\mathcal{S}')} \cdot \frac{1}{d_{M'}(\mathcal{S}')}, \quad d_M(\mathcal{S}') = 3, or, 1$$

$1/8$ accounts for the random choices of I (M) and of one out of four neighbors.

Acceptance probability



step 2, a test move

A given bond configuration may correspond with different loop configurations.

step 3, or another $\mathcal{S}'$

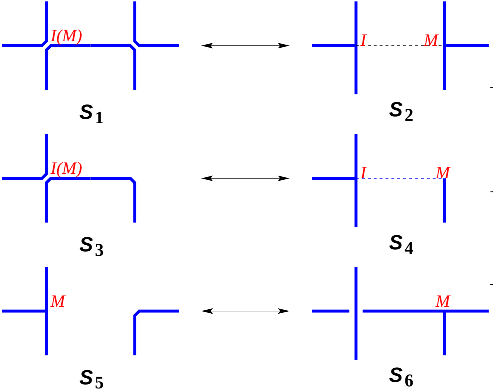
The acceptance probability

$$P_a(\mathcal{S} \to \mathcal{S}') = \min\left(1, \frac{p_p(\mathcal{S}|\mathcal{S}')}{p_p(\mathcal{S}'|\mathcal{S})} \cdot \frac{\phi_{\mathcal{S}'}}{\phi_{\mathcal{S}}}\right) \ ,$$

$$= \min\left(1, \frac{d_M(\mathcal{S}')d_{M'}(\mathcal{S}')\omega_M(\mathcal{S}')\omega_{M'}(\mathcal{S}')n^{\Delta l}}{d_M(\mathcal{S})d_{M'}(\mathcal{S})\omega_M(\mathcal{S})\omega_{M'}(\mathcal{S})}\right) \ .$$

where $\Delta l = l(\mathcal{S}') - l(\mathcal{S})$
is the change of the number of loops

Examples



$$P_a(\mathcal{S}_1 \to \mathcal{S}_2) = \min\left(1, \frac{z^2 n^{l_2}}{9w^2 n^{l_1}}\right)$$

$$P_a(\mathcal{S}_3 \to \mathcal{S}_4) = \min\left(1, \frac{zyn^{l_4}}{3wun^{l_3}}\right)$$

$$P_a(\mathcal{S}_5 \to \mathcal{S}_6) = \min\left(1, \frac{3czn^{l_6}}{uzn^{l_5}}\right)$$

$n \neq 1$ case

In the calculation of the acceptance probability P_a, one has to count the change Δl of the loop number. This is a nonlocal procedure.

- Sweeny algorithm
- Coloring technique

Worm algorithm with the coloring technique ($n \geq 1$)

1. Randomly choose a vertex $k \in V$, move $I = M$ to k, and do the following: independently for each loop, color all its occupied *bonds* to be "active" (green) with probability $1/n$ and to be "inactive" (red) with probability $1 - 1/n$; all empty edges are assigned "active" (green).

Worm algorithm with the coloring technique ($n \geq 1$)

1. Randomly choose a vertex $k \in V$, move $I = M$ to k, and do the following: independently for each loop, color all its occupied *bonds* to be "active" (green) with probability $1/n$ and to be "inactive" (red) with probability $1 - 1/n$; all empty edges are assigned "active" (green).
2. Let the current state be $\mathcal{S}$. Randomly choose I or M–say I; randomly choose one I' of the four nearest-neighbor vertices of I. If the edge e between I and I' is "red", the present Monte Carlo step is *done*. Otherwise, propose moving $I \rightarrow I'$ and inverting the edge e between I and I' as: $e = 0 \leftrightarrow e = 1$.

Worm algorithm with the coloring technique ($n \geq 1$)

1. Randomly choose a vertex $k \in V$, move $I = M$ to k, and do the following: independently for each loop, color all its occupied *bonds* to be "active" (green) with probability $1/n$ and to be "inactive" (red) with probability $1 - 1/n$; all empty edges are assigned "active" (green).
2. Let the current state be $\mathcal{S}$. Randomly choose I or M–say I; randomly choose one I' of the four nearest-neighbor vertices of I. If the edge e between I and I' is "red", the present Monte Carlo step is *done*. Otherwise, propose moving $I \to I'$ and inverting the edge e between I and I' as: $e = 0 \leftrightarrow e = 1$.
3. Randomly select one of the possible states $\mathcal{S}'$ by taking into account all possible pairings for vertices I and I' after the proposed bond update. Note that different pairings can only occur for "green" occupied bonds; no repairing should occur for edges in red, and no pairing exists between "red" and "green" edges.

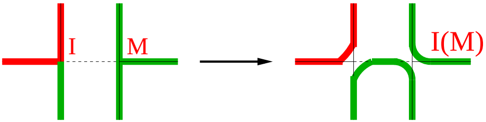
Worm algorithm with the coloring technique ($n \geq 1$)

1. Randomly choose a vertex $k \in V$, move $I = M$ to k, and do the following: independently for each loop, color all its occupied *bonds* to be "active" (green) with probability $1/n$ and to be "inactive" (red) with probability $1 - 1/n$; all empty edges are assigned "active" (green).
2. Let the current state be $\mathcal{S}$. Randomly choose I or M–say I; randomly choose one I' of the four nearest-neighbor vertices of I. If the edge e between I and I' is "red", the present Monte Carlo step is *done*. Otherwise, propose moving $I \to I'$ and inverting the edge e between I and I' as: $e = 0 \leftrightarrow e = 1$.
3. Randomly select one of the possible states $\mathcal{S}'$ by taking into account all possible pairings for vertices I and I' after the proposed bond update. Note that different pairings can only occur for "green" occupied bonds; no repairing should occur for edges in red, and no pairing exists between "red" and "green" edges.
4. Accept the proposal update $\mathcal{S} \to \mathcal{S}'$ with the appropriate acceptance probability $P_a(\mathcal{S} \to \mathcal{S}')$.

Worm algorithm with the coloring technique ($n \geq 1$)

1. Randomly choose a vertex $k \in V$, move $I = M$ to k, and do the following: independently for each loop, color all its occupied *bonds* to be "active" (green) with probability $1/n$ and to be "inactive" (red) with probability $1 - 1/n$; all empty edges are assigned "active" (green).
2. Let the current state be $\mathcal{S}$. Randomly choose I or M–say I; randomly choose one I' of the four nearest-neighbor vertices of I. If the edge e between I and I' is "red", the present Monte Carlo step is *done*. Otherwise, propose moving $I \to I'$ and inverting the edge e between I and I' as: $e = 0 \leftrightarrow e = 1$.
3. Randomly select one of the possible states $\mathcal{S}'$ by taking into account all possible pairings for vertices I and I' after the proposed bond update. Note that different pairings can only occur for "green" occupied bonds; no repairing should occur for edges in red, and no pairing exists between "red" and "green" edges.
4. Accept the proposal update $\mathcal{S} \to \mathcal{S}'$ with the appropriate acceptance probability $P_a(\mathcal{S} \to \mathcal{S}')$.
5. Relabel I' as I. If $I = M$, then goto 1; else go to 2.

Example

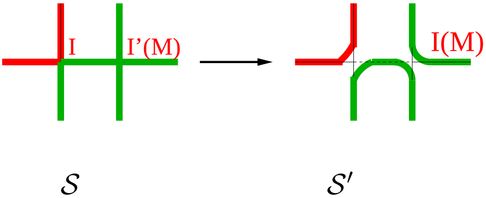


$\mathcal{S}$ $\qquad\qquad$ $\mathcal{S}'$

$$d_I(\mathcal{S}) = 1, \qquad d_{I'}(\mathcal{S}) = 1$$

$$d_I(\mathcal{S}') = 1, \qquad d_{I'}(\mathcal{S}') = 3$$

Example



$$d_I(\mathcal{S}) = 1, \qquad d_{I'}(\mathcal{S}) = 1$$

$$d_I(\mathcal{S}') = 1, \qquad d_{I'}(\mathcal{S}') = 3$$

- Test the worm algorithm by studying the critical properties of the model
- Check the efficiency of the algorithm

Exactly known critical exponents of the O(n) loop model

At the critical branch of the model

- thermal exponent: $y_t = \frac{4g-4}{g}$
- Magnetic exponent: $y_h = 1 + \frac{1}{2g} + \frac{3g}{8}$
- Hull exponent: $y_H = 1 + \frac{1}{2g}$
  which describes the decay of the probability that two bonds are sitting at the same loop, is also the fractal dimension d_l of the loops.

g is the Coulomb-gas coupling: $n = -2\cos(\pi g), \quad 1 \leq g \leq 3/2$

Determination of the critical points and y_t

We determine the critical point in two subspace:

- without crossing bonds, $u = v = x, w = x^2, c = 0$
- with crossing bonds, $u = v = x, w = c = x^2$

Determination of the critical points and y_t

We determine the critical point in two subspace:

- without crossing bonds, $u = v = x, w = x^2, c = 0$
- with crossing bonds, $u = v = x, w = c = x^2$

Consider the wrapping probability P_w

$$P_w(x, L) = P_w^{(0)} + a(x - x_c)L^{y_t} + b_1 L^{y_{u_1}} + \cdots$$

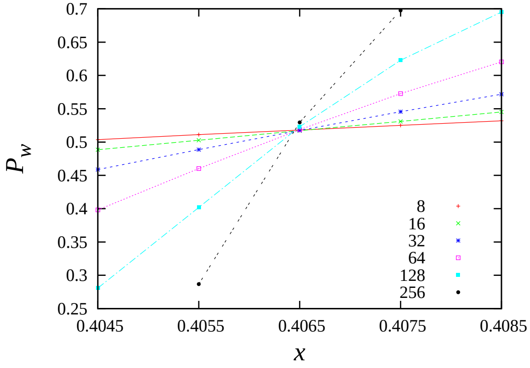
Determination of the critical points and y_t

We determine the critical point in two subspace:

- without crossing bonds, $u = v = x, w = x^2, c = 0$
- with crossing bonds, $u = v = x, w = c = x^2$

Consider the wrapping probability P_w

$$P_w(x, L) = P_w^{(0)} + a(x - x_c)L^{y_t} + b_1 L^{y_{u1}} + \cdots$$



without crossing bonds, $n = 1$

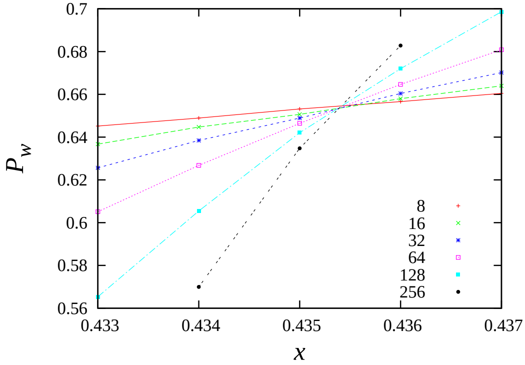
Determination of the critical points and y_t

We determine the critical point in two subspace:

- without crossing bonds, $u = v = x, w = x^2, c = 0$
- with crossing bonds, $u = v = x, w = c = x^2$

Consider the wrapping probability P_w

$$P_w(x, L) = P_w^{(0)} + a(x - x_c)L^{y_t} + b_1 L^{y_{u_1}} + \cdots$$



with crossing bonds, $n = 1.5$

Determination of other exponents

Simulate the model at the estimated critical point.

$n_b =$ the average density of occupied bonds

$n_w =$ the average fraction of edges covered by the wrapping loop

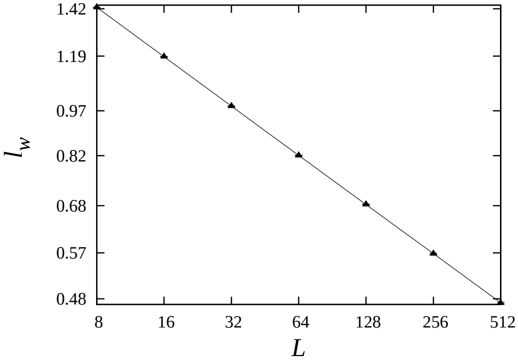
$S_2 =$ the average of the sum of squares of loop lengths per site

$l_w =$ the average length of worm steps per site

Determination of other exponents: y_h

$n = 1.5$, in the subspace with crossing bonds

$$l_w \propto L^{2y_h - 4}$$

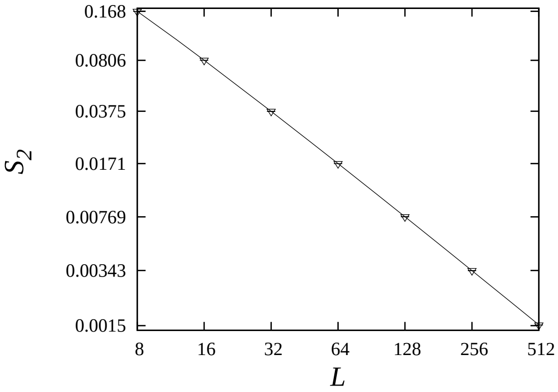


$$y_h = 1.8679(6)$$

Determination of other exponents: y_H

$n = 1.5$, in the subspace with crossing bonds

$$S_2 \propto L^{2y_H - 4}$$

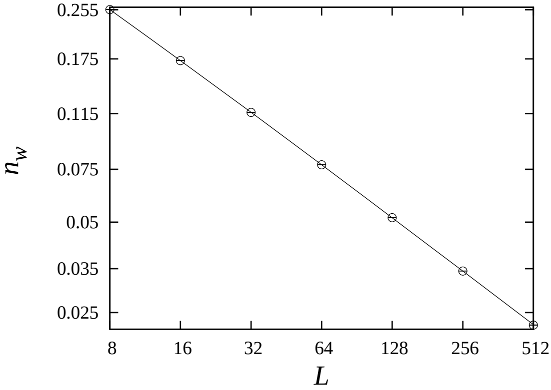


$$y_H = 1.405(2)$$

Determination of other exponents: y_H

$n = 1.5$, in the subspace with crossing bonds

$$n_w \propto L^{y_H - 2}$$



$$y_H = d_l = 1.405(2)$$

Numerical results

Simulation results (S) in the subspace $u = v = x, w = x^2, c = 0$.

n		x_c	y_t	y_h	y_H	$P_w^{(0)}$
1	E		1	1.875	1.375	
	T	0.40644(1)				
	S	0.40644(1)	1.002(3)	1.8749(3)	1.374(1)	0.516(1)
1.5	E		0.748109	1.86776	1.40649	
	T	0.43535(2)				
	S	0.43535(1)	0.747(5)	1.8675(5)	1.4067(6)	0.6530(4)

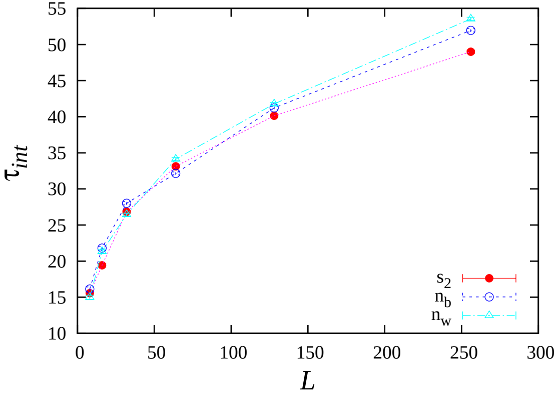
Numerical results

Simulation results (S) in the subspace $u = v = x, w = c = x^2$.

n		x_c	y_t	y_h	y_H	$P_w^{(0)}$
1	E		1	1.875	1.375	
	T	0.398048(2)				
	S	0.398050(5)	1.001(3)	1.8749(3)	1.3755(6)	0.516(1)
1.5	E		0.748109	1.86776	1.40649	
	T	0.423622(2)				
	S	0.42366(5)	0.744(5)	1.8679(6)	1.405 (2)	0.654(1)

Dynamic behavior of the algorithm

Integrated autocorrelation times τ_{int} versus lattice size L ($n = 1.5$) in the subspace $u = v = x, w = c = x^2$. (The unit of time is normalized to 'visit per site').



$z(S_2) \approx 0.2, z(n_b) \approx 0.3, z(n_w) \approx 0.3$

Summary

- We developed a worm algorithm for the O(n) intersecting loop model on the square lattice.

Summary

- We developed a worm algorithm for the O(n) intersecting loop model on the square lattice.
- Our algorithm has little critical slowing-down when $1 \leq n \leq 2$.

Summary

- We developed a worm algorithm for the O(n) intersecting loop model on the square lattice.
- Our algorithm has little critical slowing-down when $1 \le n \le 2$.
- We tested this algorithm by investigating the critical properties of the model, for which we determine the critical points and several critical exponents.

Summary

- We developed a worm algorithm for the O(n) intersecting loop model on the square lattice.
- Our algorithm has little critical slowing-down when $1 \leq n \leq 2$.
- We tested this algorithm by investigating the critical properties of the model, for which we determine the critical points and several critical exponents.
- The present of crossing bonds is found to be irrelevant at the critical branch of the loop.

Summary

- We developed a worm algorithm for the O(n) intersecting loop model on the square lattice.
- Our algorithm has little critical slowing-down when $1 \leq n \leq 2$.
- We tested this algorithm by investigating the critical properties of the model, for which we determine the critical points and several critical exponents.
- The present of crossing bonds is found to be irrelevant at the critical branch of the loop.
- we shall check the low-temperature phase of the intersecting loop model

Summary

- We developed a worm algorithm for the O(n) intersecting loop model on the square lattice.
- Our algorithm has little critical slowing-down when $1 \leq n \leq 2$.
- We tested this algorithm by investigating the critical properties of the model, for which we determine the critical points and several critical exponents.
- The present of crossing bonds is found to be irrelevant at the critical branch of the loop.
- we shall check the low-temperature phase of the intersecting loop model

Thank You

Thank You