$\hbar$

Second Lecture:
Quantum Monte Carlo Techniques

Andreas Läuchli,
"New states of quantum matter"
MPI für Physik komplexer Systeme - Dresden

http://www.pks.mpg.de/~aml
aml@pks.mpg.de

Lecture Notes at http:www.pks.mpg.de/~aml/LesHouches

"Modern theories of correlated electron systems" - Les Houches - 19/5/2009

Outline of today's lecture

- Quantum Monte Carlo (World line)
- Loop Algorithm
- Stochastic Series Expansion
- Worm Algorithm
- Green's function Monte Carlo
- Sign Problem

- What is not treated here:
  - Auxiliary field Monte Carlo for fermions
  - Diagrammatic Monte Carlo
  - CT-QMC for Quantum impurity problems
  - Real Time Quantum Monte Carlo

1. Quantum Monte Carlo (World lines)

Quantum Monte Carlo

- In quantum statistical mechanics the canonical partition function now reads:

$$Z(T) = \mathrm{Tr}\ e^{-H/T}$$

- Expectation values:

$$\langle \mathcal{O} \rangle = \frac{1}{Z(T)} \mathrm{Tr}[\mathcal{O} e^{-H/T}] \equiv \mathrm{Tr}[\mathcal{O}\rho(T)]$$

- We need to find a mapping onto a “classical” problem in order to perform MC
  - World-line methods (Suzuki-Trotter, Loop algorithm, Worm algorithm, ...)
  - Stochastic Series Expansions

- Potential Problem: The mapping can give “probabilities” which are negative ⇒ infamous sign problem. Common cause is frustration or fermionic statistics.

Quantum Monte Carlo

- Feynman (1953) lays foundation for quantum Monte Carlo
- Map d dimensional quantum system to classical world lines in d+1 dimensions

THE

PHYSICAL REVIEW

A journal of experimental and theoretical physics established by E. L. Nichols in 1893

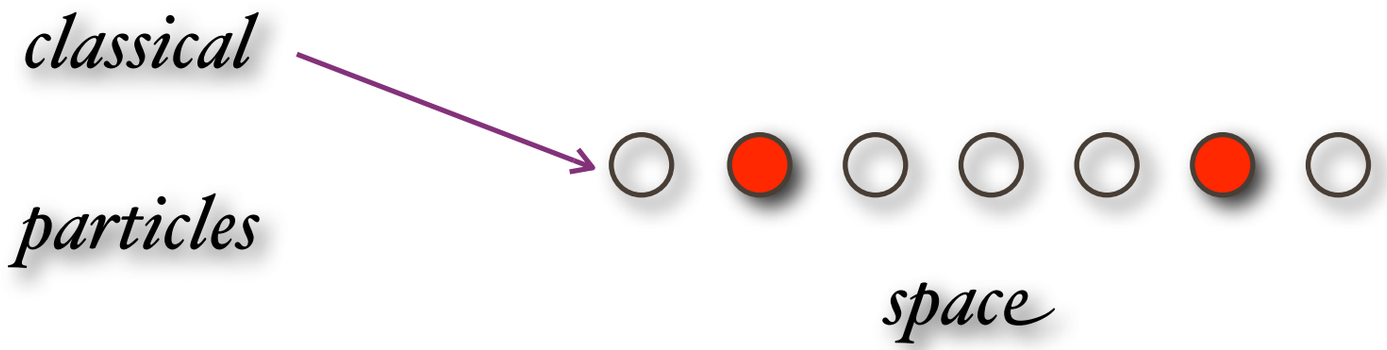
SECOND SERIES, VOL. 91, NO. 6 SEPTEMBER 15, 1953

Atomic Theory of the λ Transition in Helium

R. P. FEYNMAN
California Institute of Technology, Pasadena, California
(Received May 15, 1953)

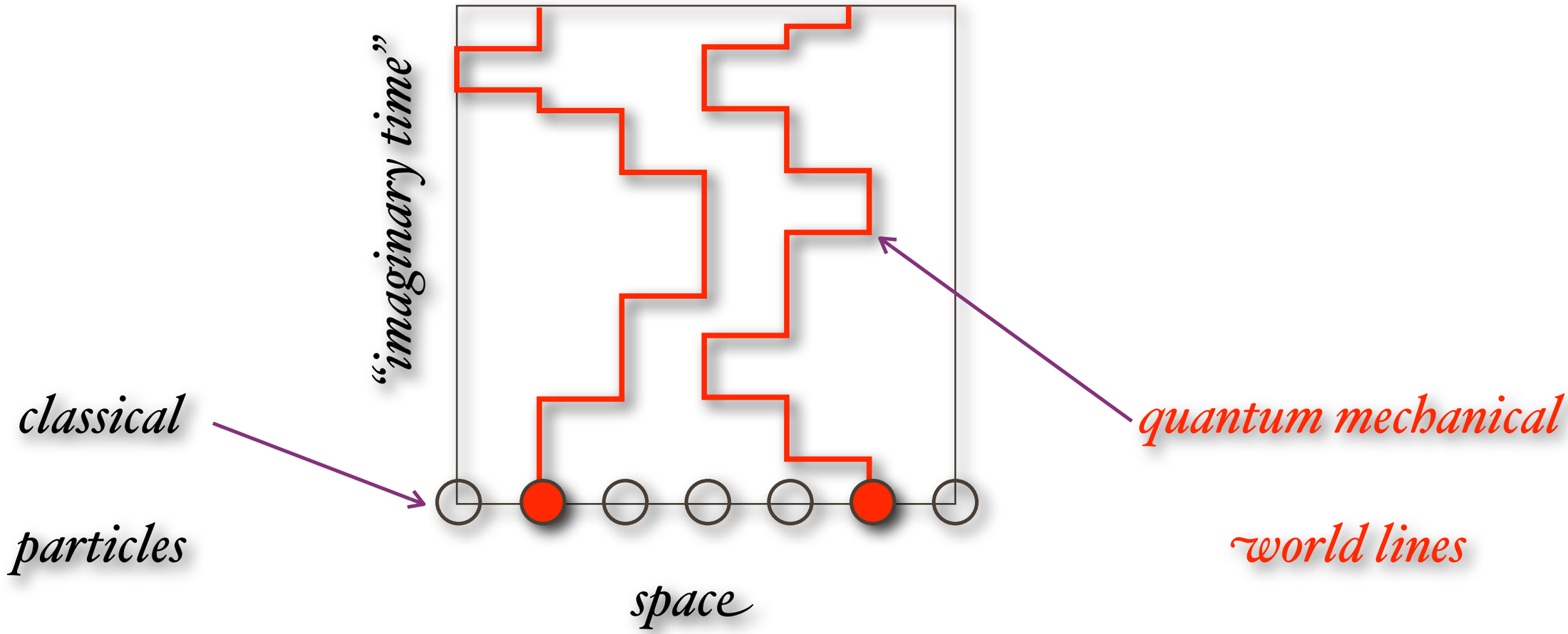
Quantum Monte Carlo

- Feynman (1953) lays foundation for quantum Monte Carlo
- Map d dimensional quantum system to classical world lines in d+1 dimensions



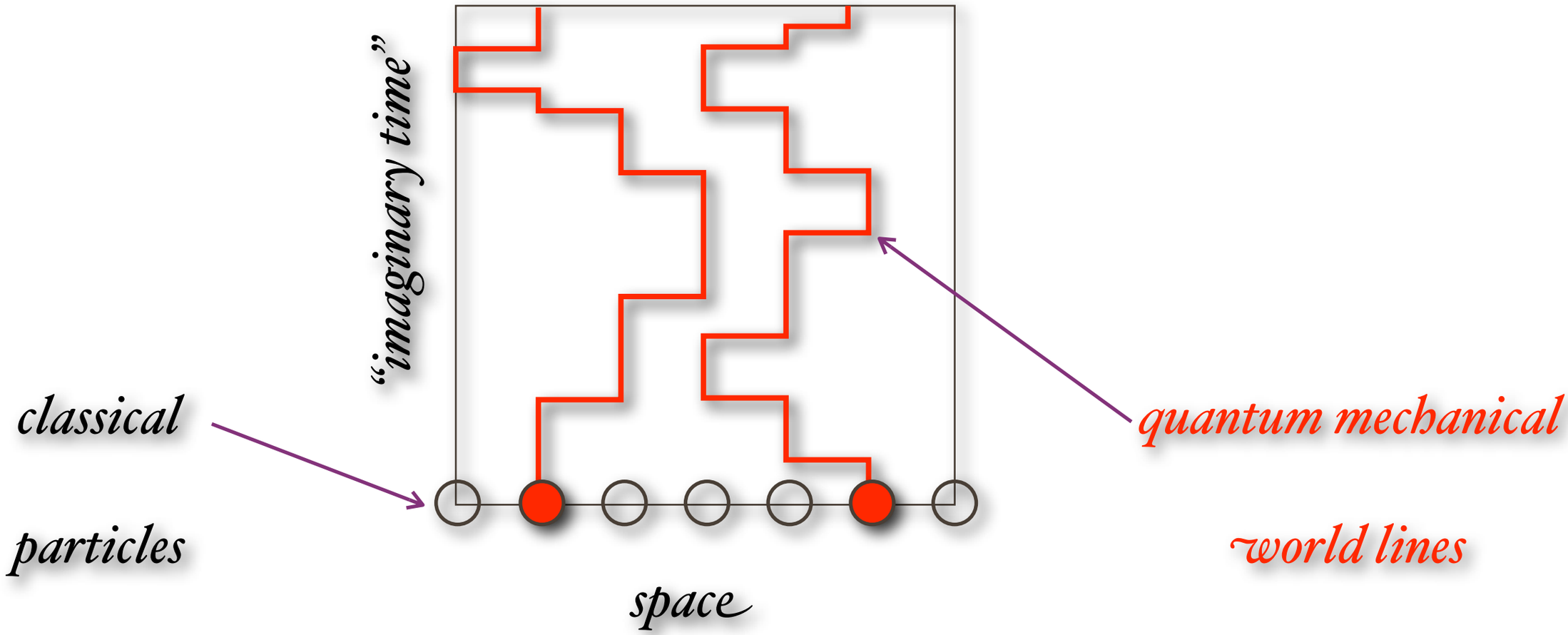
Quantum Monte Carlo

- Feynman (1953) lays foundation for quantum Monte Carlo
- Map d dimensional quantum system to classical world lines in d+1 dimensions



Quantum Monte Carlo

- Feynman (1953) lays foundation for quantum Monte Carlo
- Map d dimensional quantum system to classical world lines in d+1 dimensions



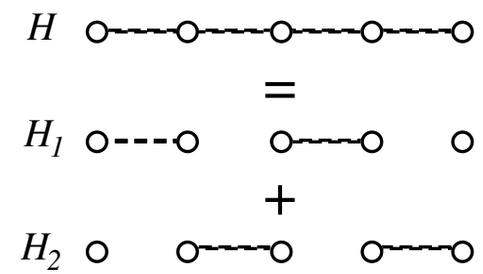
Use Metropolis algorithm to update world lines

The Suzuki-Trotter Decomposition

- Generic mapping of a quantum spin system to “Ising” model (vertex model)
  - basis of most discrete time QMC algorithms
  - not limited to special models
- Split Hamiltonian into two easily diagonalized pieces

$$H = H_1 + H_2$$

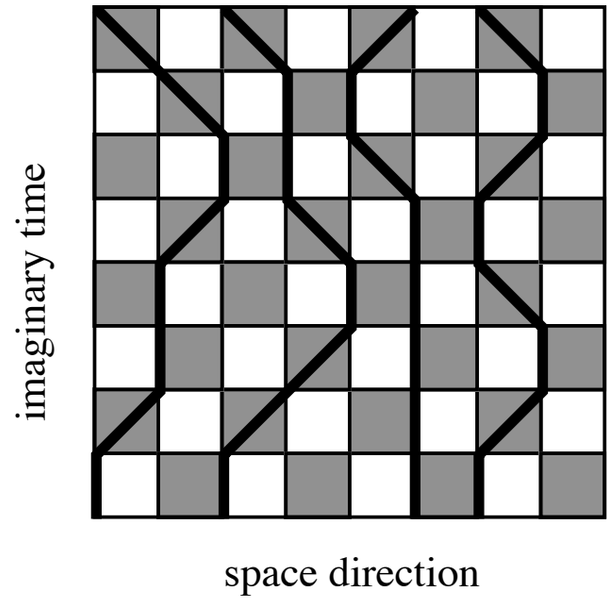
$$e^{-\varepsilon H} = e^{-\varepsilon (H_1 + H_2)} = e^{-\varepsilon H_1} e^{-\varepsilon H_2} + O(\varepsilon^2)$$



- Obtain the checkerboard decomposition

$$Z = \mathrm{Tr}\left[\exp(-\beta H)\right] = \mathrm{Tr}\left[e^{-\beta (H_1 + H_2)}\right]$$

$$= \mathrm{Tr}\left[e^{-(\beta / M) H_1} e^{-(\beta / M) H_2}\right]^M + O(\beta^3 / M^2)$$



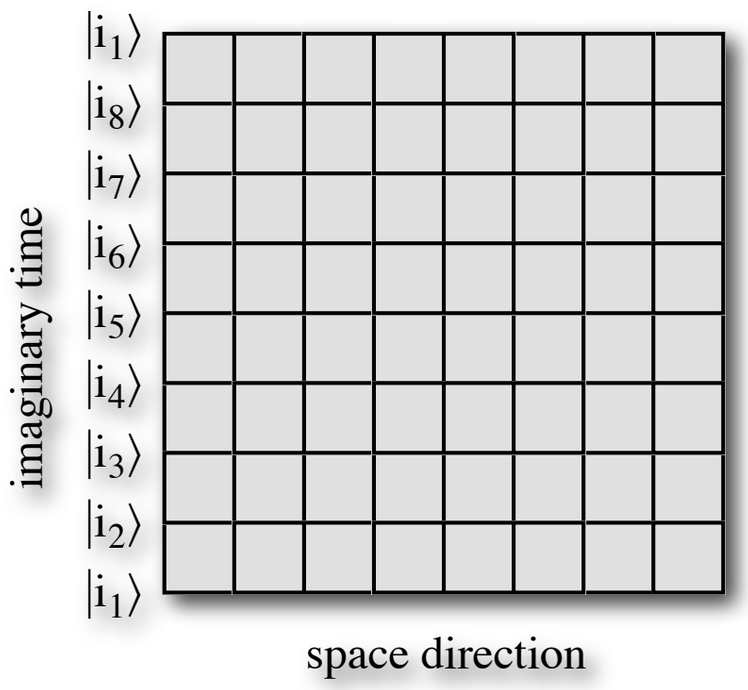
Path integral QMC

- Use Trotter-Suzuki or a simple low-order formula

$$Z = \mathrm{Tr}\, e^{-\beta H} = \mathrm{Tr}\, e^{-M\Delta\tau H} = \mathrm{Tr}\left(e^{-\Delta\tau H}\right)^M = \mathrm{Tr}\left(1-\Delta\tau H\right)^M + O(\beta\Delta\tau)$$

$$= \sum_{\{(i_1 \ldots i_M)\}} \langle i_1 | 1-\Delta\tau H | i_2 \rangle \langle i_2 | 1-\Delta\tau H | i_3 \rangle \cdots \langle i_M | 1-\Delta\tau H | i_1 \rangle$$

- gives a mapping to a (d+1)-dimensional classical model



- partition function of quantum system is sum over classical world lines

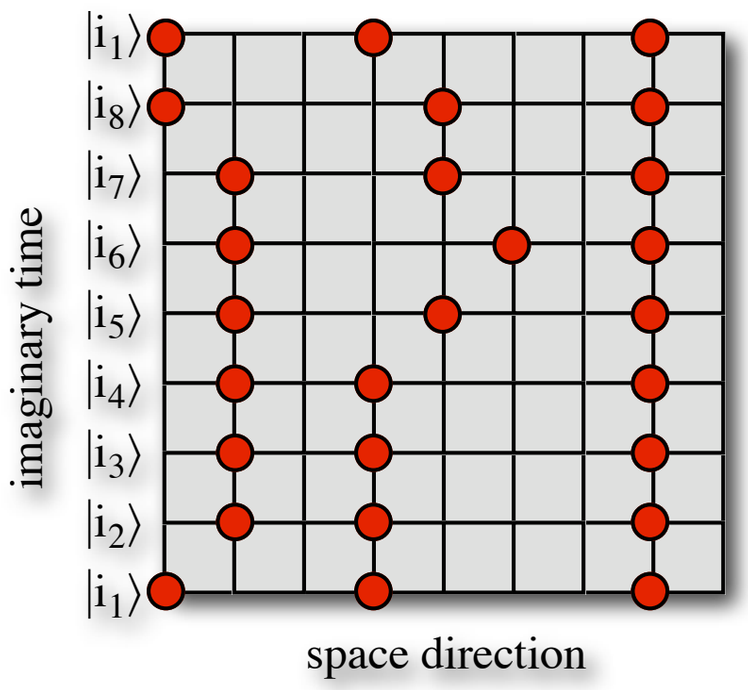
Path integral QMC

- Use Trotter-Suzuki or a simple low-order formula

$$Z = \mathrm{Tr}\, e^{-\beta H} = \mathrm{Tr}\, e^{-M\Delta\tau H} = \mathrm{Tr}\left(e^{-\Delta\tau H}\right)^M = \mathrm{Tr}\left(1-\Delta\tau H\right)^M + O(\beta\Delta\tau)$$

$$= \sum_{\{(i_1 \ldots i_M)\}} \langle i_1 | 1-\Delta\tau H | i_2 \rangle \langle i_2 | 1-\Delta\tau H | i_3 \rangle \cdots \langle i_M | 1-\Delta\tau H | i_1 \rangle$$

- gives a mapping to a (d+1)-dimensional classical model



place particles (spins)

- partition function of quantum system is sum over classical world lines

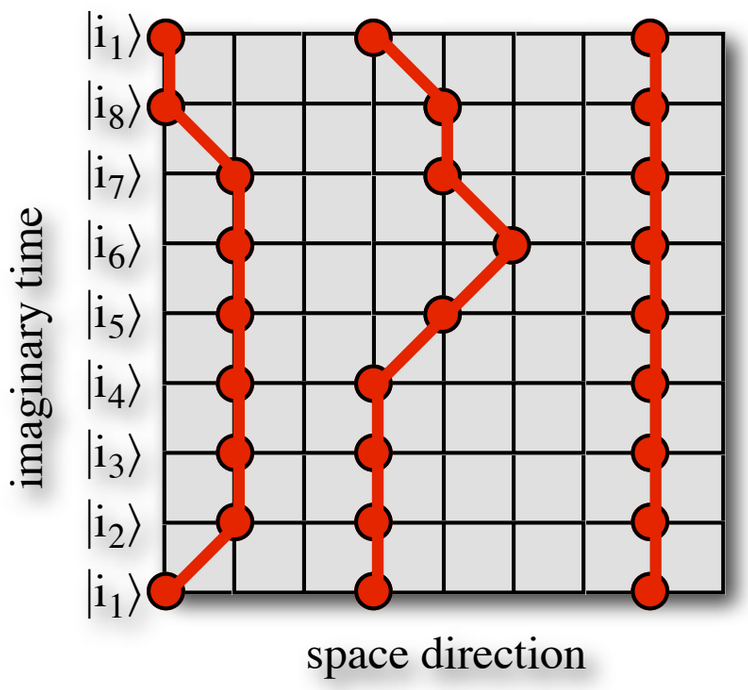
Path integral QMC

- Use Trotter-Suzuki or a simple low-order formula

$$Z = \mathrm{Tr}\, e^{-\beta H} = \mathrm{Tr}\, e^{-M\Delta\tau H} = \mathrm{Tr}\left(e^{-\Delta\tau H}\right)^M = \mathrm{Tr}\left(1-\Delta\tau H\right)^M + O(\beta\Delta\tau)$$

$$= \sum_{\{(i_1 \ldots i_M)\}} \langle i_1 | 1-\Delta\tau H | i_2 \rangle \langle i_2 | 1-\Delta\tau H | i_3 \rangle \cdots \langle i_M | 1-\Delta\tau H | i_1 \rangle$$

- gives a mapping to a (d+1)-dimensional classical model



place particles (spins)

for Hamiltonians conserving particle number (magnetization) we get world lines

- partition function of quantum system is sum over classical world lines

Calculating configuration weights

$$Z = \mathrm{Tr}\, e^{-\beta H} = \mathrm{Tr}\, e^{-M\Delta\tau H} = \mathrm{Tr}\left(e^{-\Delta\tau H}\right)^M = \mathrm{Tr}\left(1-\Delta\tau H\right)^M + O(\beta\Delta\tau)$$

$$= \sum_{\{(i_1 \ldots i_M)\}} \langle i_1 | 1-\Delta\tau H | i_2 \rangle \langle i_2 | 1-\Delta\tau H | i_3 \rangle \cdots \langle i_M | 1-\Delta\tau H | i_1 \rangle$$

- Examples: particles with nearest neighbor repulsion

$$H = -t \sum_{\langle i,j \rangle} (a_i^\dagger a_j + a_j^\dagger a_i) + V \sum_{\langle i,j \rangle} n_i n_j$$

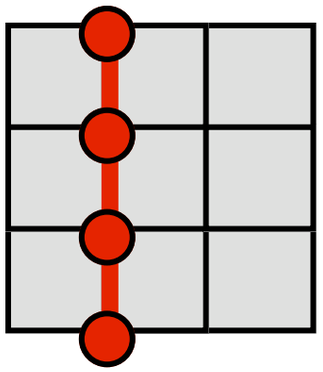
Calculating configuration weights

$$Z = \mathrm{Tr}\, e^{-\beta H} = \mathrm{Tr}\, e^{-M\Delta\tau H} = \mathrm{Tr}\left(e^{-\Delta\tau H}\right)^M = \mathrm{Tr}\left(1-\Delta\tau H\right)^M + O(\beta\Delta\tau)$$

$$= \sum_{\{(i_1 \ldots i_M)\}} \langle i_1 | 1-\Delta\tau H | i_2 \rangle \langle i_2 | 1-\Delta\tau H | i_3 \rangle \cdots \langle i_M | 1-\Delta\tau H | i_1 \rangle$$

- Examples: particles with nearest neighbor repulsion

$$H = -t \sum_{\langle i,j \rangle} (a_i^\dagger a_j + a_j^\dagger a_i) + V \sum_{\langle i,j \rangle} n_i n_j$$



1

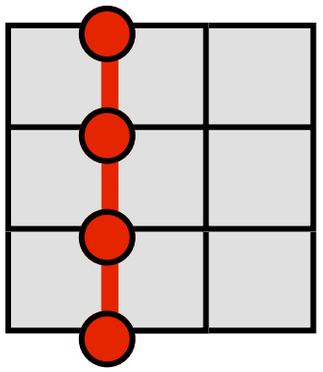
Calculating configuration weights

$$Z = \mathrm{Tr}\, e^{-\beta H} = \mathrm{Tr}\, e^{-M\Delta\tau H} = \mathrm{Tr}\left(e^{-\Delta\tau H}\right)^M = \mathrm{Tr}\left(1-\Delta\tau H\right)^M + O(\beta\Delta\tau)$$

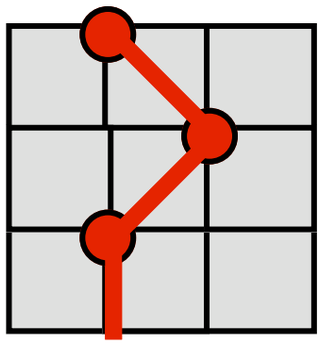
$$= \sum_{\{(i_1 \ldots i_M)\}} \langle i_1 | 1-\Delta\tau H | i_2 \rangle \langle i_2 | 1-\Delta\tau H | i_3 \rangle \cdots \langle i_M | 1-\Delta\tau H | i_1 \rangle$$

- Examples: particles with nearest neighbor repulsion

$$H = -t \sum_{\langle i,j \rangle} (a_i^\dagger a_j + a_j^\dagger a_i) + V \sum_{\langle i,j \rangle} n_i n_j$$



1



$(\Delta\tau t)^2$

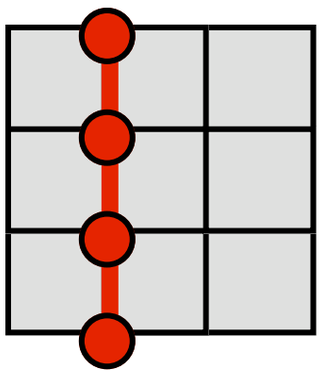
Calculating configuration weights

$$Z = \mathrm{Tr}\, e^{-\beta H} = \mathrm{Tr}\, e^{-M\Delta\tau H} = \mathrm{Tr}\left(e^{-\Delta\tau H}\right)^M = \mathrm{Tr}\left(1-\Delta\tau H\right)^M + O(\beta\Delta\tau)$$

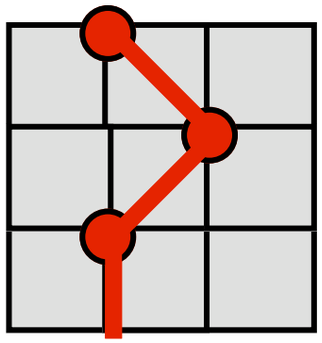
$$= \sum_{\{(i_1 \ldots i_M)\}} \langle i_1 | 1-\Delta\tau H | i_2 \rangle \langle i_2 | 1-\Delta\tau H | i_3 \rangle \cdots \langle i_M | 1-\Delta\tau H | i_1 \rangle$$

- Examples: particles with nearest neighbor repulsion

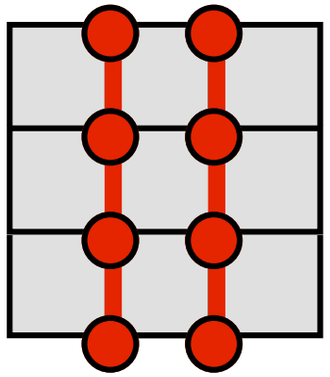
$$H = -t \sum_{\langle i,j \rangle} (a_i^\dagger a_j + a_j^\dagger a_i) + V \sum_{\langle i,j \rangle} n_i n_j$$



1



$(\Delta\tau t)^2$



$(1-\Delta\tau V)^3$

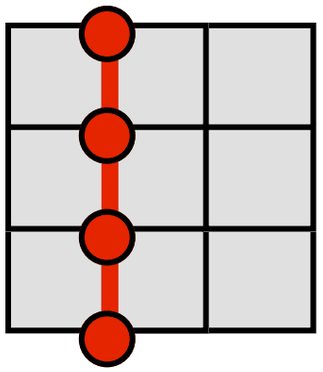
Calculating configuration weights

$$Z = \mathrm{Tr}\, e^{-\beta H} = \mathrm{Tr}\, e^{-M\Delta\tau H} = \mathrm{Tr}\left(e^{-\Delta\tau H}\right)^M = \mathrm{Tr}\left(1-\Delta\tau H\right)^M + O(\beta\Delta\tau)$$

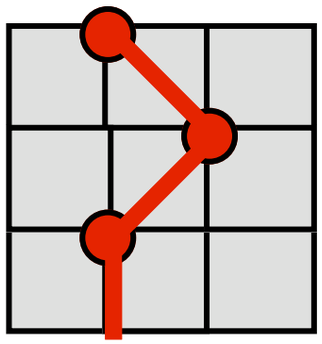
$$= \sum_{\{(i_1 \ldots i_M)\}} \langle i_1 | 1-\Delta\tau H | i_2 \rangle \langle i_2 | 1-\Delta\tau H | i_3 \rangle \cdots \langle i_M | 1-\Delta\tau H | i_1 \rangle$$

- Examples: particles with nearest neighbor repulsion

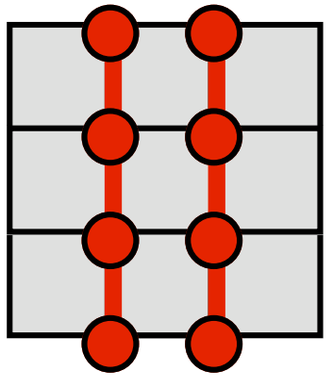
$$H = -t \sum_{\langle i,j \rangle} (a_i^\dagger a_j + a_j^\dagger a_i) + V \sum_{\langle i,j \rangle} n_i n_j$$



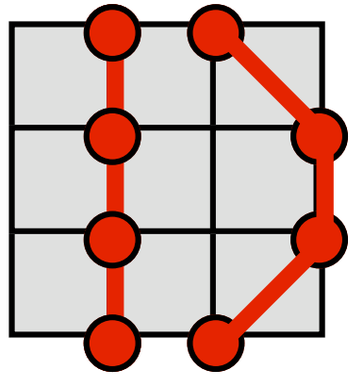
1



$(\Delta\tau t)^2$



$(1-\Delta\tau V)^3$



$(\Delta\tau t)^2$

Monte Carlo updates

- just move the world lines locally
  - probabilities given by matrix element of Hamiltonian
  - example: tight binding model

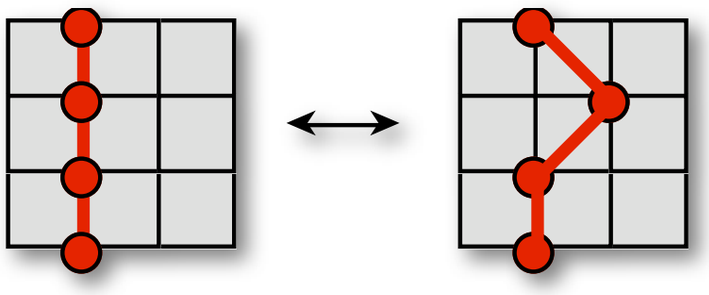
$$H = -t \sum_{\langle i,j \rangle} \left(c_i^\dagger c_{i+1} + c_{i+1}^\dagger c_i \right)$$

Monte Carlo updates

- just move the world lines locally
  - probabilities given by matrix element of Hamiltonian
  - example: tight binding model

$$H = -t \sum_{\langle i,j \rangle} \left(c_i^\dagger c_{i+1} + c_{i+1}^\dagger c_i \right)$$

introduce or remove two kinks:

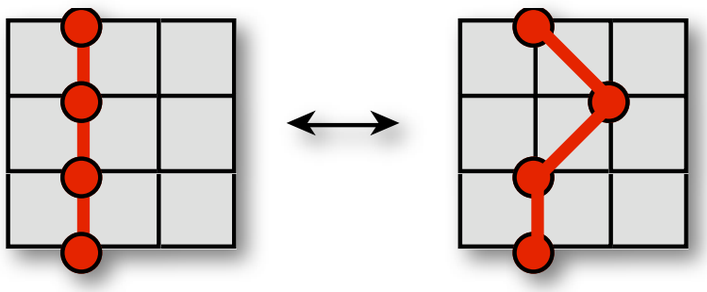


Monte Carlo updates

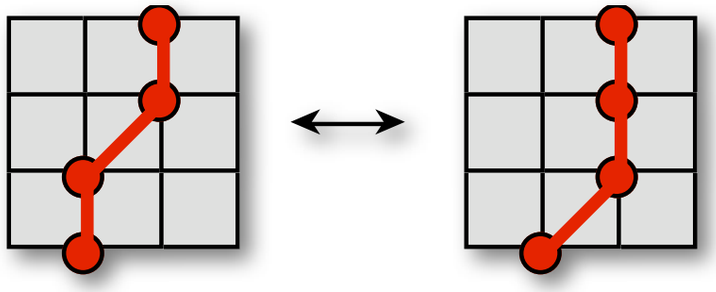
- just move the world lines locally
  - probabilities given by matrix element of Hamiltonian
  - example: tight binding model

$$H = -t \sum_{\langle i,j \rangle} \left(c_i^\dagger c_{i+1} + c_{i+1}^\dagger c_i \right)$$

introduce or remove two kinks:



shift a kink:

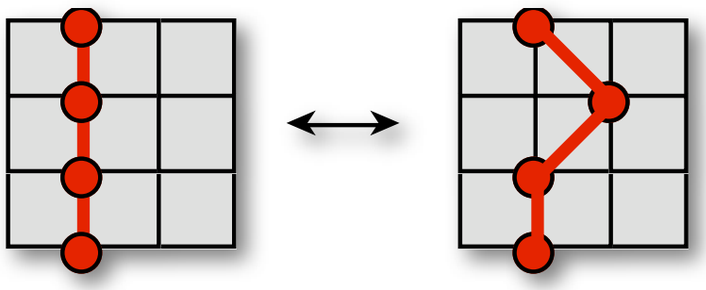


Monte Carlo updates

- just move the world lines locally
  - probabilities given by matrix element of Hamiltonian
  - example: tight binding model

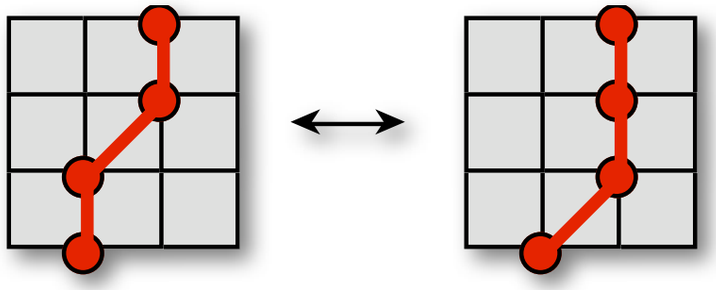
$$H = -t \sum_{\langle i,j \rangle} \left(c_i^\dagger c_{i+1} + c_{i+1}^\dagger c_i \right)$$

introduce or remove two kinks:



$P = 1$ $\qquad$ $P = (\Delta\tau t)^2$

shift a kink:



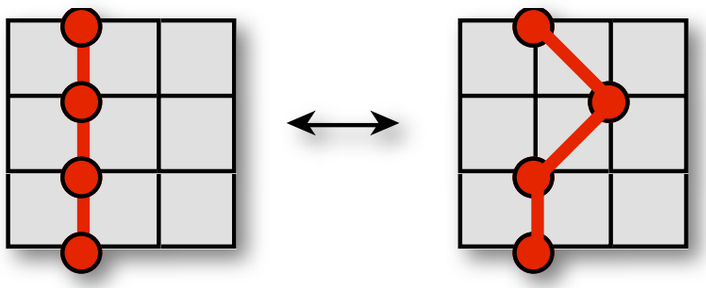
$P = \Delta\tau t$

Monte Carlo updates

- just move the world lines locally
  - probabilities given by matrix element of Hamiltonian
  - example: tight binding model

$$H = -t \sum_{\langle i,j \rangle} \left(c_i^\dagger c_{i+1} + c_{i+1}^\dagger c_i \right)$$

introduce or remove two kinks:

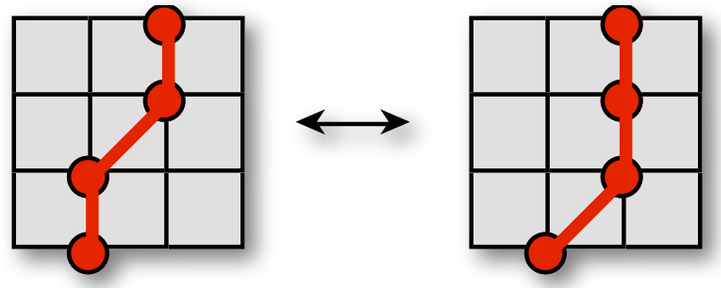


$$P = 1 \qquad P = (\Delta\tau t)^2$$

$$P_{\rightarrow} = \min\left[1,(\Delta\tau t)^2\right]$$

$$P_{\leftarrow} = \min\left[1,1/(\Delta\tau t)^2\right]$$

shift a kink:

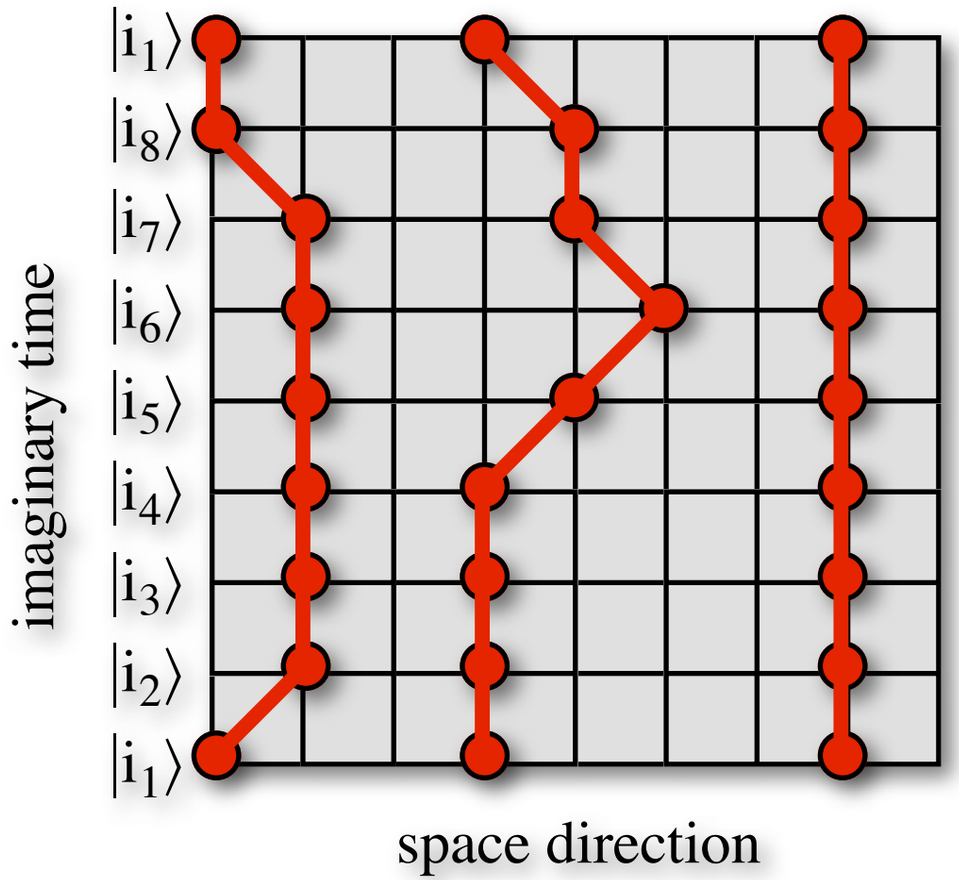


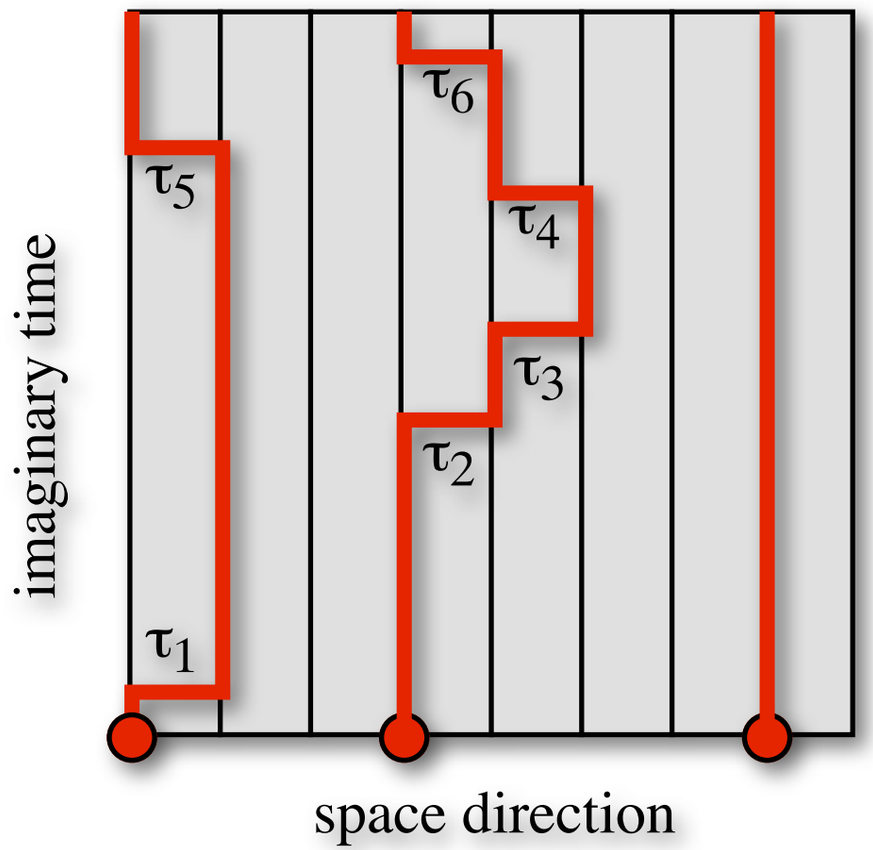
$$P = \Delta\tau t$$

$$P_{\rightarrow} = P_{\leftarrow} = 1$$

The continuous time limit

- the limit $\Delta\tau \to 0$ can be taken in the algorithm
  [Prokof'ev *et al.*, Pis'ma v Zh.Eks. Teor. Fiz. **64**, 853 (1996)]





- discrete time: store configuration at all time steps

- continuous time: store times at which configuration changes

Advantages of continuous time

- No need to extrapolate in time step
  - a single simulation is sufficient
  - no additional errors from extrapolation
- Less memory and CPU time required
  - Instead of a time step $\Delta\tau << t$ we only have to store changes in the configuration happening at mean distances $\approx t$
  - Speedup of $1 / \Delta\tau \approx 10$
- Conceptual advantage
  - we directly sample a diagrammatic perturbation expansion

2. Cluster updates: The loop algorithm

Problems with local updates

Problems with local updates

- Local updates cannot change global topological properties
  - number of world lines (particles, magnetization) conserved
  - winding conserved
  - braiding conserved
  - cannot sample grand-canonical ensemble

Problems with local updates

- Local updates cannot change global topological properties
  - number of world lines (particles, magnetization) conserved
  - winding conserved
  - braiding conserved
    - cannot sample grand-canonical ensemble
- Critical slowing down at second order phase transitions
  - solved by cluster updates

Problems with local updates

- Local updates cannot change global topological properties
  - number of world lines (particles, magnetization) conserved
  - winding conserved
  - braiding conserved
    - cannot sample grand-canonical ensemble
- Critical slowing down at second order phase transitions
  - solved by cluster updates
- Tunneling problem at first order phase transitions
  - solved by extended sampling techniques

Cluster algorithms: the formal explanation

$$Z = \sum_C W(C) = \sum_C \sum_G W(C,G) \text{ with } W(C) = \sum_G W(C,G)$$

$$W(C,G) = \Delta(C,G)V(G) \text{ where } \Delta(C,G) = \begin{cases} 1 & \text{graph } G \text{ allowed for } C \\ 0 & \text{otherwise} \end{cases}$$

$$C_i \quad \rightarrow \quad (C_i,G) \quad \rightarrow \quad G \quad \rightarrow \quad (C_{i+1},G) \quad \rightarrow \quad C_{i+1}$$

Cluster algorithms: the formal explanation

- Extend the phase space to configurations + graphs (*C,G*)

$$Z = \sum_C W(C) = \sum_C \sum_G W(C,G) \text{ with } W(C) = \sum_G W(C,G)$$

- Choose graph weights independent of configuration (C)

$$W(C,G) = \Delta(C,G)V(G) \text{ where } \Delta(C,G) = \begin{cases} 1 & \text{graph } G \text{ allowed for } C \\ 0 & \text{otherwise} \end{cases}$$

- Perform updates

$$C_i \quad \rightarrow \quad (C_i,G) \quad \rightarrow \quad G \quad \rightarrow \quad (C_{i+1},G) \quad \rightarrow \quad C_{i+1}$$

Cluster algorithms: the formal explanation

- Extend the phase space to configurations + graphs (*C,G*)

$$Z = \sum_C W(C) = \sum_C \sum_G W(C,G) \text{ with } W(C) = \sum_G W(C,G)$$

- Choose graph weights independent of configuration (C)

$$W(C,G) = \Delta(C,G)V(G) \text{ where } \Delta(C,G) = \begin{cases} 1 & \text{graph } G \text{ allowed for } C \\ 0 & \text{otherwise} \end{cases}$$

- Perform updates

$$C_i \rightarrow (C_i,G) \rightarrow G \rightarrow (C_{i+1},G) \rightarrow C_{i+1}$$

1. Pick a graph G $\quad P[G] = \dfrac{V(G)}{W(C)}$

Cluster algorithms: the formal explanation

- Extend the phase space to configurations + graphs (*C,G*)

$$Z = \sum_C W(C) = \sum_C \sum_G W(C,G) \text{ with } W(C) = \sum_G W(C,G)$$

- Choose graph weights independent of configuration (C)

$$W(C,G) = \Delta(C,G)V(G) \text{ where } \Delta(C,G) = \begin{cases} 1 & \text{graph } G \text{ allowed for } C \\ 0 & \text{otherwise} \end{cases}$$

- Perform updates

2. Discard configuration

$$C_i \rightarrow (C_i,G) \rightarrow G \rightarrow (C_{i+1},G) \rightarrow C_{i+1}$$

1. Pick a graph G $\quad P[G] = \dfrac{V(G)}{W(C)}$

Cluster algorithms: the formal explanation

- Extend the phase space to configurations + graphs (*C,G*)

$$Z = \sum_C W(C) = \sum_C \sum_G W(C,G) \text{ with } W(C) = \sum_G W(C,G)$$

- Choose graph weights independent of configuration (C)

$$W(C,G) = \Delta(C,G)V(G) \text{ where } \Delta(C,G) = \begin{cases} 1 & \text{graph } G \text{ allowed for } C \\ 0 & \text{otherwise} \end{cases}$$

- Perform updates

2. Discard configuration

$$C_i \rightarrow (C_i,G) \rightarrow G \rightarrow (C_{i+1},G) \rightarrow C_{i+1}$$

1. Pick a graph G $P[G] = \dfrac{V(G)}{W(C)}$

3. Pick any allowed new configuration

Cluster algorithms: the formal explanation

- Extend the phase space to configurations + graphs (*C,G*)

$$Z = \sum_C W(C) = \sum_C \sum_G W(C,G) \text{ with } W(C) = \sum_G W(C,G)$$

- Choose graph weights independent of configuration (C)

$$W(C,G) = \Delta(C,G)V(G) \text{ where } \Delta(C,G) = \begin{cases} 1 & \text{graph } G \text{ allowed for } C \\ 0 & \text{otherwise} \end{cases}$$

- Perform updates

2. Discard configuration

4. Discard graph

$$C_i \rightarrow (C_i,G) \rightarrow G \rightarrow (C_{i+1},G) \rightarrow C_{i+1}$$

1. Pick a graph G $P[G] = \dfrac{V(G)}{W(C)}$

3. Pick any allowed new configuration

Cluster algorithms: the formal explanation

- Extend the phase space to configurations + graphs (*C,G*)

$$Z = \sum_C W(C) = \sum_C \sum_G W(C,G) \text{ with } W(C) = \sum_G W(C,G)$$

- Choose graph weights independent of configuration (C)

$$W(C,G) = \Delta(C,G)V(G) \text{ where } \Delta(C,G) = \begin{cases} 1 & \text{graph } G \text{ allowed for } C \\ 0 & \text{otherwise} \end{cases}$$

- Perform updates

2. Discard configuration

4. Discard graph

$$C_i \rightarrow (C_i,G) \rightarrow G \rightarrow (C_{i+1},G) \rightarrow C_{i+1}$$

1. Pick a graph G $P[G] = \dfrac{V(G)}{W(C)}$

3. Pick any allowed new configuration

- Detailed balance is satisfied

$$\frac{P[(C_i,G) \rightarrow (C_{i+1},G)]}{P[(C_{i+1},G) \rightarrow (C_i,G)]} = \frac{1/\mathrm{N_C}}{1/\mathrm{N_C}} = 1 = \frac{\Delta(C_{i+1},G)V(G)}{\Delta(C_i,G)V(G)} = \frac{P[(C_{i+1},G)]}{P[(C_i,G)]}$$

Cluster algorithms: Ising model

- We need to find $\Delta(C,G)$ and $V(G)$ to fulfill $W(C) = \sum_G W(C,G) = \sum_G \Delta(C,G)V(G)$

$\Delta(C,G)$	o-o	o o	$W(C)$
↑↑, ↓↓	1	1	$e^{+\beta J}$
↑↓, ↓↑	0	1	$e^{-\beta J}$
$W(G)$	$e^{+\beta J} - e^{-\beta J}$	$e^{-\beta J}$	

- This means for: $C_i \rightarrow (C_i,G) \rightarrow G$
  - Parallel spins: pick connected graph **o-o** with $P(\text{ o-o }) = \dfrac{e^{+\beta J} + e^{-\beta J}}{e^{+\beta J}} = 1 - e^{-2\beta J}$
  - Antiparallel spins: always pick open graph **o o**
- And for: $G \rightarrow (C_{i+1},G) \rightarrow C_{i+1}$
  - Configuration must be allowed ⇒ connected spins must be parallel
    ⇒ connected spins flipped as one cluster

PHYSICAL REVIEW
LETTERS

VOLUME 70 | 15 FEBRUARY 1993 | NUMBER 7

Cluster Algorithm for Vertex Models

Hans Gerd Evertz,[1],[a] Gideon Lana,[2],[b] and Mihai Marcu[2],[c]
[1] *Supercomputer Computations Research Institute, Florida State University, Tallahassee, Florida 32306*
[2] *School of Physics and Astronomy, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, 69978 Tel Aviv, Israel*

(Received 17 November 1992)

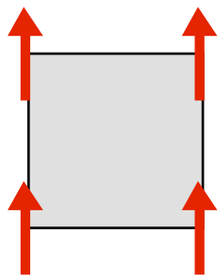
We present a new type of cluster algorithm that strongly reduces critical slowing down in simulations of vertex models. Since the clusters are closed paths of bonds, we call it the *loop algorithm*. The basic steps in constructing a cluster are the breakup and the freezing of vertices. We concentrate on the case of the F model, which is a subset of the six-vertex model exhibiting a Kosterlitz-Thouless transition. The loop algorithm is also applicable to simulations of other vertex models and of one- and two-dimensional quantum spin systems.

PACS numbers: 02.70.–c, 05.50.+q, 68.35.Rh, 75.10.Jm

The loop algorithm (Evertz *et al)*

- Swendsen-Wang cluster algorithm for the Ising model
  - two choices on each bond: connected (flip both spins) or disconnected

  - all connected spins are flipped together
- Loop algorithm is a generalization to quantum systems
  - world lines may not be broken
  - always 2 or 4 spins must be flipped together

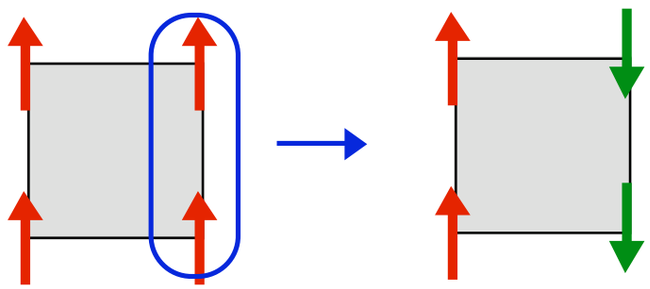


  - four different connection types

The loop algorithm (Evertz *et al)*

- Swendsen-Wang cluster algorithm for the Ising model
  - two choices on each bond: connected (flip both spins) or disconnected

  - all connected spins are flipped together
- Loop algorithm is a generalization to quantum systems
  - world lines may not be broken
  - always 2 or 4 spins must be flipped together



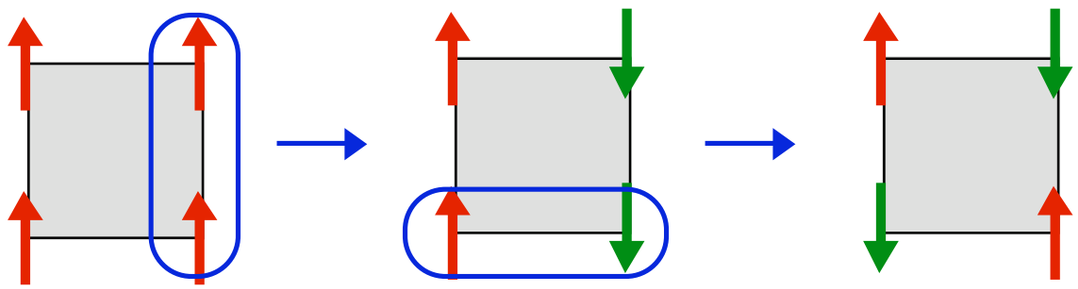
  - four different connection types

The loop algorithm (Evertz *et al)*

- Swendsen-Wang cluster algorithm for the Ising model
  - two choices on each bond: connected (flip both spins) or disconnected

  - all connected spins are flipped together

- Loop algorithm is a generalization to quantum systems
  - world lines may not be broken
  - always 2 or 4 spins must be flipped together



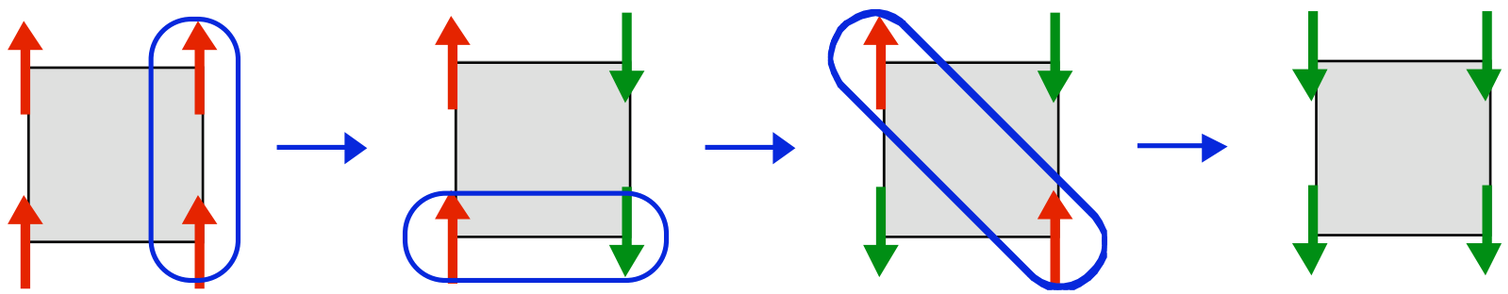
  - four different connection types

The loop algorithm (Evertz *et al)*

- Swendsen-Wang cluster algorithm for the Ising model
  - two choices on each bond: connected (flip both spins) or disconnected

  - all connected spins are flipped together

- Loop algorithm is a generalization to quantum systems
  - world lines may not be broken
  - always 2 or 4 spins must be flipped together



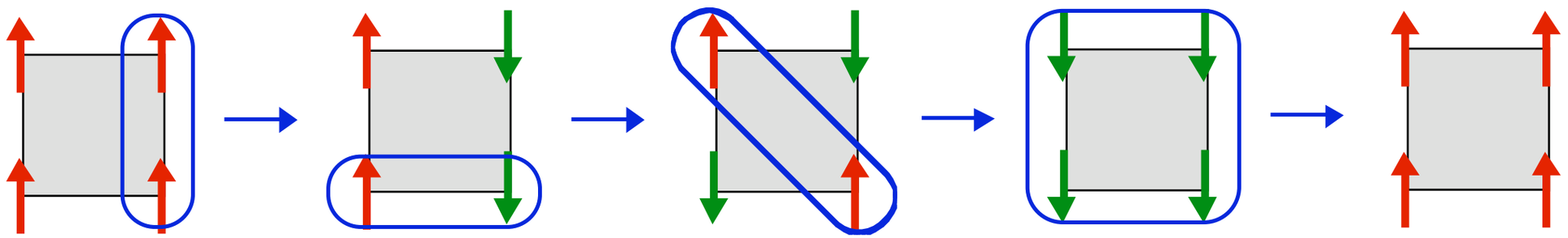
  - four different connection types

The loop algorithm (Evertz *et al)*

- Swendsen-Wang cluster algorithm for the Ising model
  - two choices on each bond: connected (flip both spins) or disconnected

  - all connected spins are flipped together

- Loop algorithm is a generalization to quantum systems
  - world lines may not be broken
  - always 2 or 4 spins must be flipped together



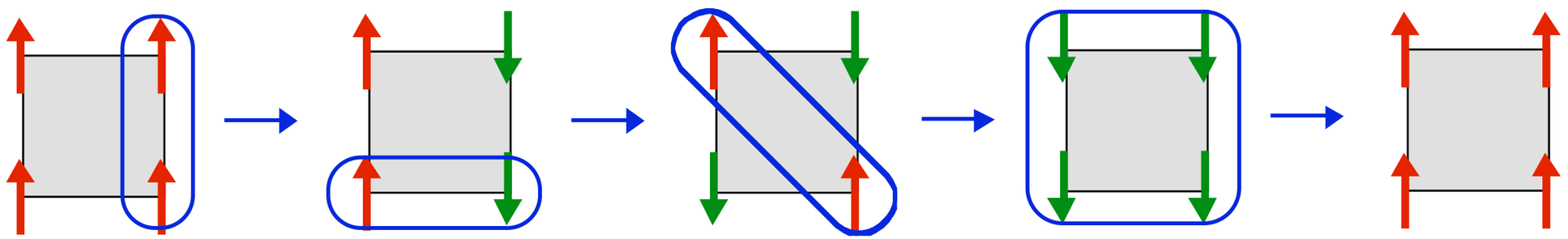
  - four different connection types

The loop algorithm (Evertz *et al)*

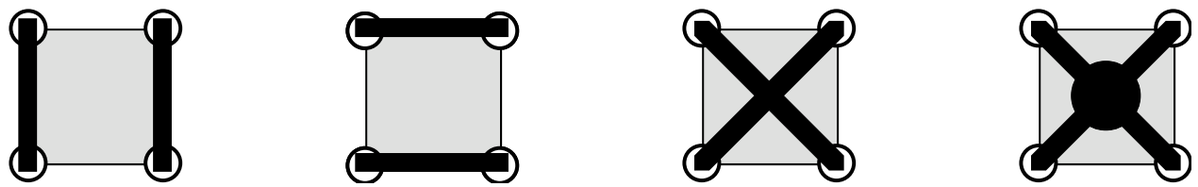
- Swendsen-Wang cluster algorithm for the Ising model
  - two choices on each bond: connected (flip both spins) or disconnected

  - all connected spins are flipped together

- Loop algorithm is a generalization to quantum systems
  - world lines may not be broken
  - always 2 or 4 spins must be flipped together



  - four different connection types



Hamiltonian of spin-1/2 models

- Consider a 2-site quantum spin-1/2 model

$$H_{XXZ} = J_{xz}(S_1^x S_2^x + S_1^y S_2^y) + J_z S_1^z S_2^z - h\left(S_1^z + S_2^z\right)$$

$$= \frac{J_{xz}}{2}(S_1^+ S_2^- + S_1^- S_2^+) + J_z S_1^z S_2^z - h\left(S_1^z + S_2^z\right)$$

  - Heisenberg model if $J_{xy} = J_z = J$

$$H = J\vec{S}_1\vec{S}_2 - h\left(S_1^z + S_2^z\right)$$

- Hamiltonian matrix in 2-site basis

$$\{\ |\uparrow\uparrow\rangle,\ |\uparrow\downarrow\rangle,\ |\downarrow\uparrow\rangle,\ |\downarrow\downarrow\rangle\ \}$$

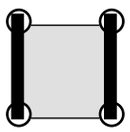
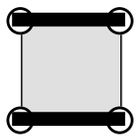
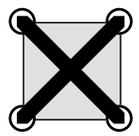
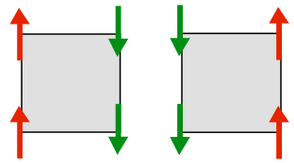
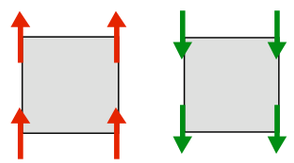
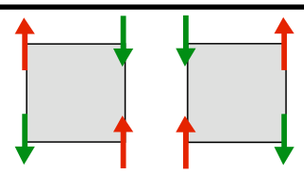
$$H_{XXZ} = \begin{pmatrix} \frac{J_z}{4} + h & 0 & 0 & 0 \\ 0 & -\frac{J_z}{4} & \frac{J_{xy}}{2} & 0 \\ 0 & \frac{J_{xy}}{2} & -\frac{J_z}{4} & 0 \\ 0 & 0 & 0 & \frac{J_z}{4} - h \end{pmatrix}$$

Cluster building rules: XY-like antiferromagnet

$$H_{XXZ} = \frac{J_{xz}}{2} \sum_{\langle i,j \rangle} (S_i^+ S_j^- + S_i^- S_j^+) + J_z \sum_{\langle i,j \rangle} S_i^z S_j^z$$

$$W(C) = \sum_G W(C,G) = \sum_G \Delta(C,G) V(G)$$

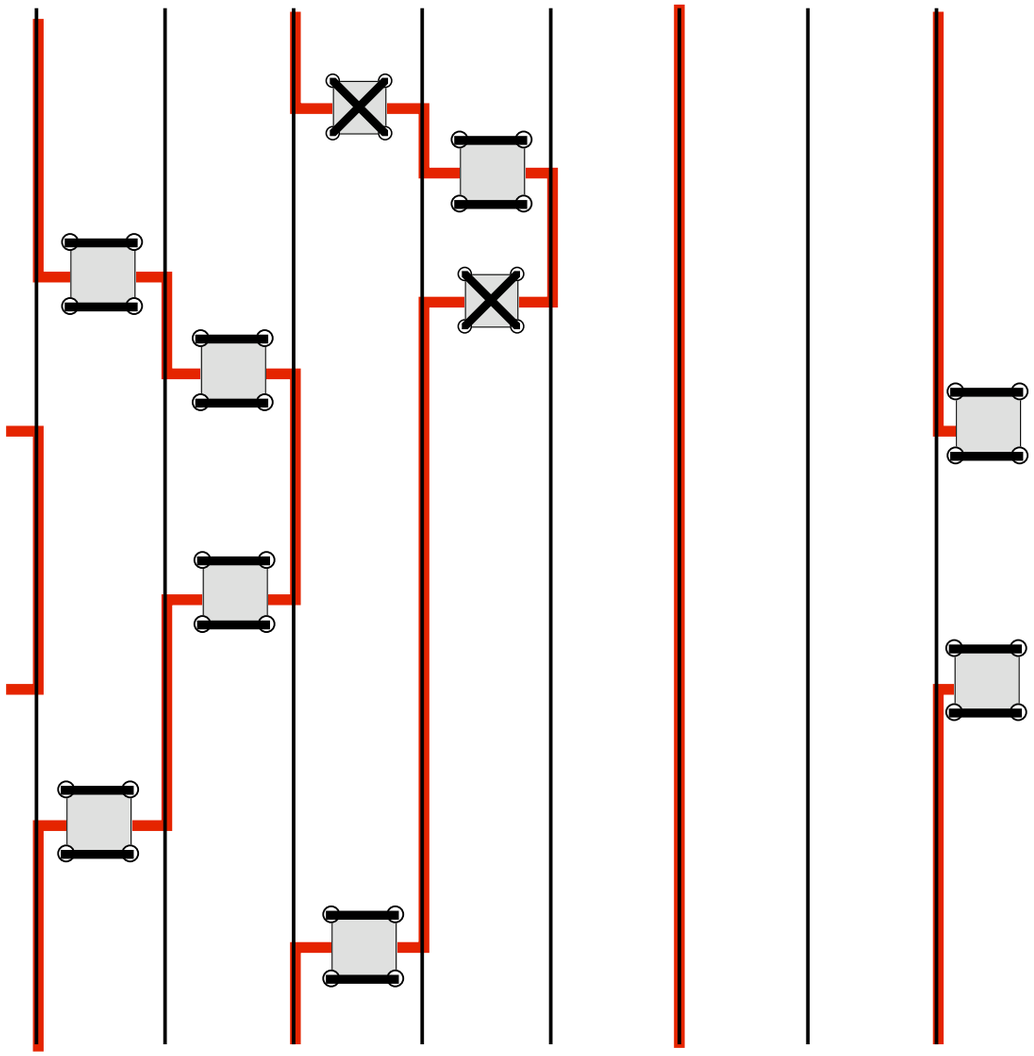
with $0 \le J_z \le J_{xy}$

$\Delta(C,G)$				$W(C)$
	1	1	–	$1+ (J_z/4)\ d\tau$
	1	–	0	$1-(J_z/4)\ d\tau$
	–	1	1	$(J_{xy}/2)\ d\tau$
$V(G)$	$1-(J_z/4)\ d\tau$	$(J_z/2)\ d\tau$	$(J_{xy}- J_z)/2\ d\tau$	

- Connected spins form a cluster and have to be flipped together

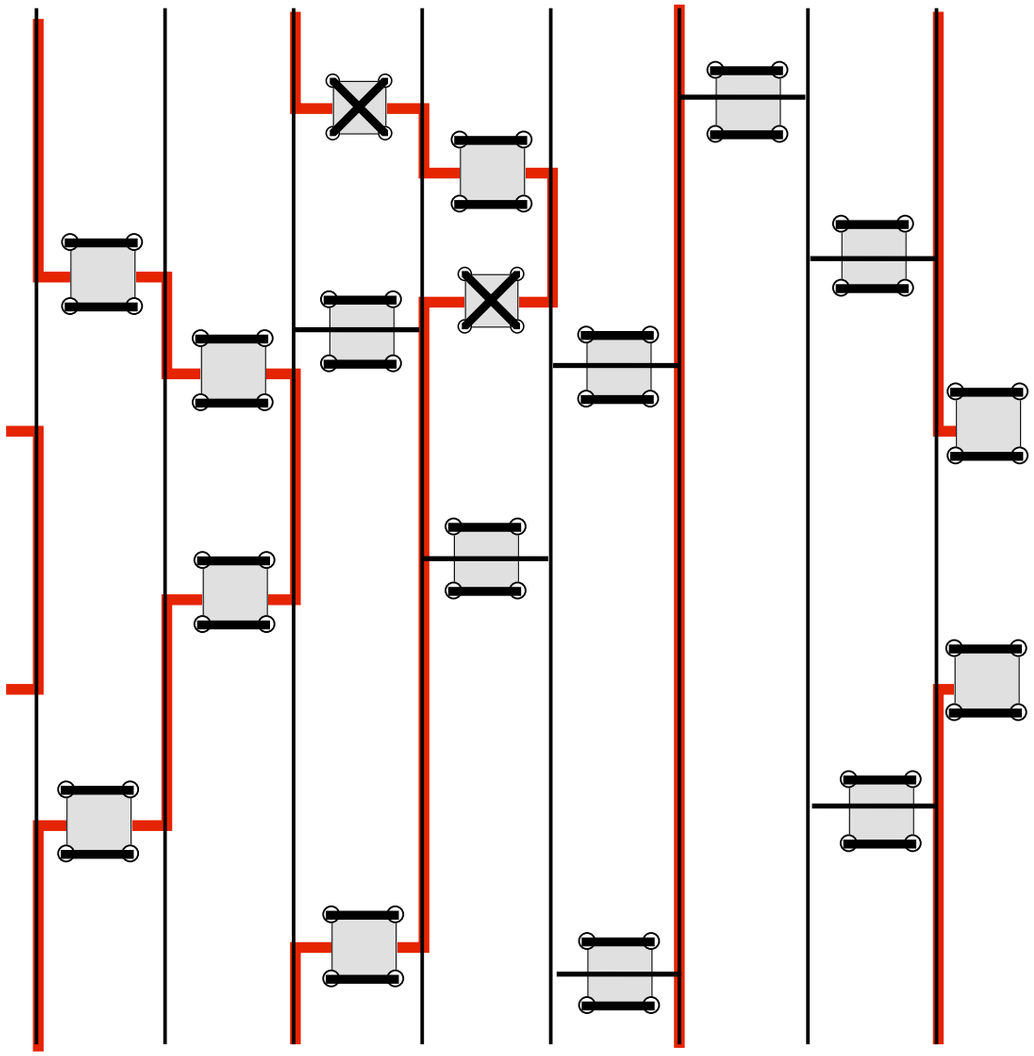
Loop-cluster updates

1. Connect spins according to loop-custer building rules



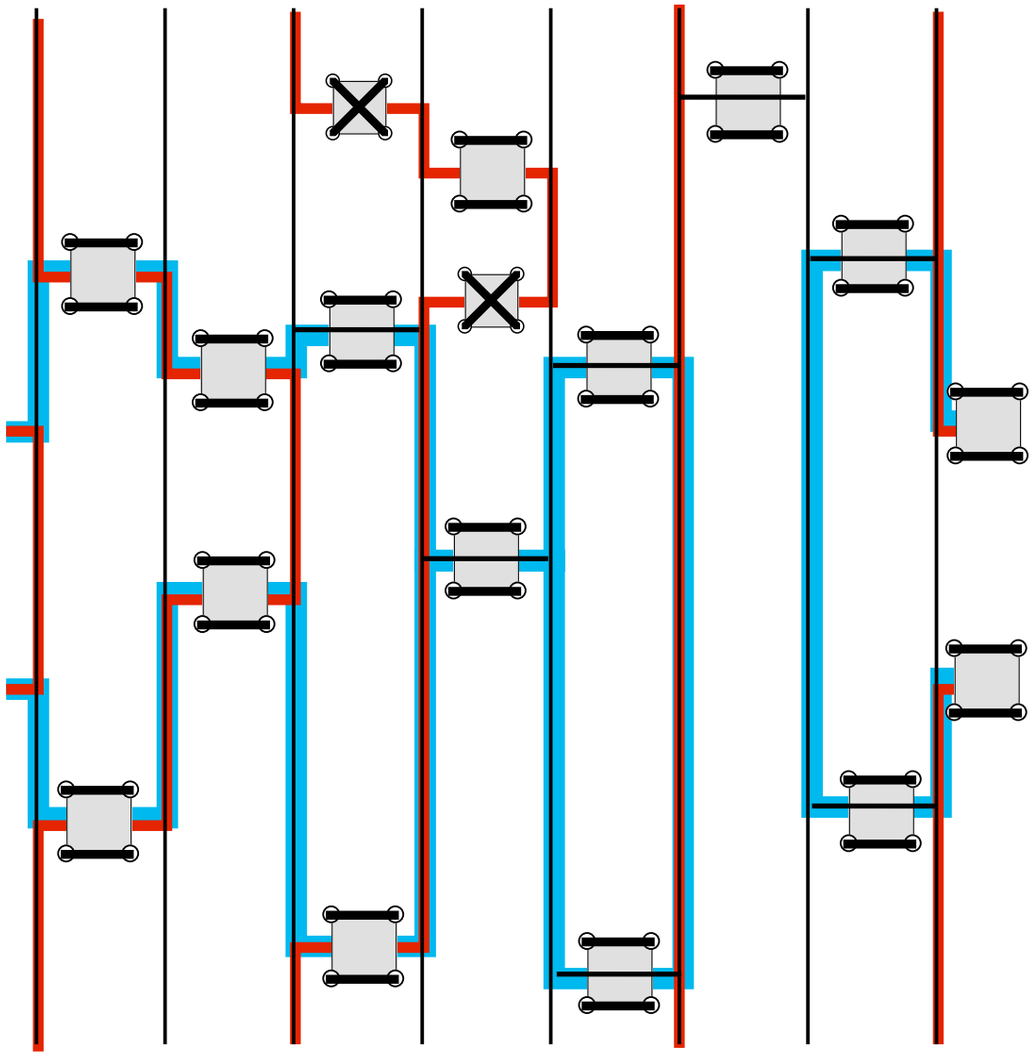
Loop-cluster updates

1. Connect spins according to loop-custer building rules



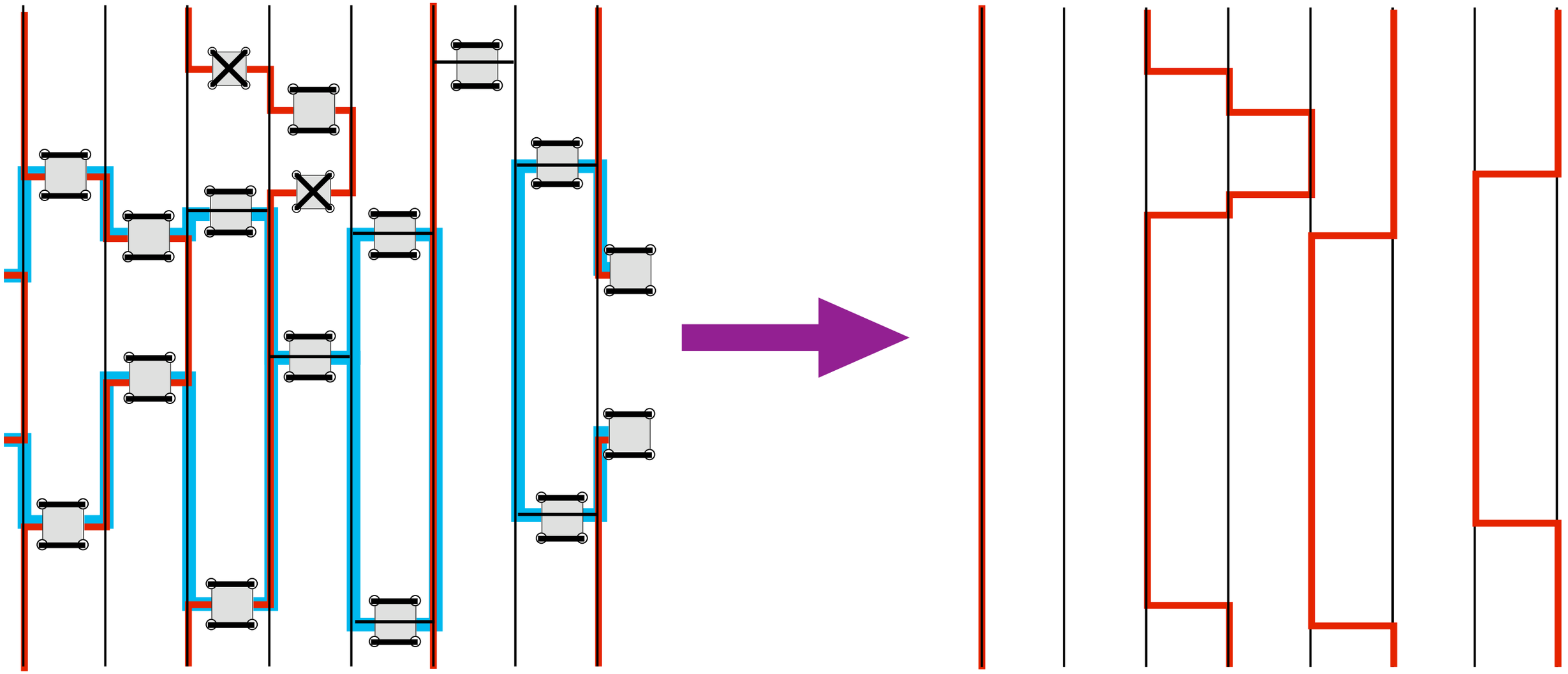
Loop-cluster updates

1. Connect spins according to loop-custer building rules
2. Build and flip loop-cluster



Loop-cluster updates

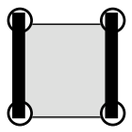
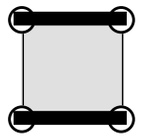
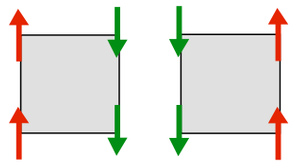
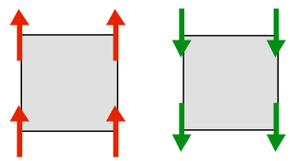
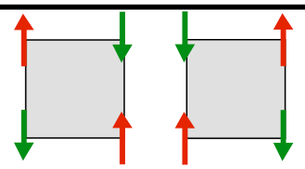
1. Connect spins according to loop-custer building rules
2. Build and flip loop-cluster



Heisenberg antiferromagnet

$$H_{\text{Heisenberg}} = J \sum_{\langle i,j \rangle} \vec{S}_i \vec{S}_j$$

$$W(C) = \sum_G W(C,G) = \sum_G \Delta(C,G) V(G)$$

$\Delta(C,G)$			$W(C)$
	1	1	$1+ (J/4)\ d\tau$
	1	0	$1-(J/4)\ d\tau$
	0	1	$(J/2)\ d\tau$
$V(G)$	$1-(J/4)\ d\tau$	$(J/2)\ d\tau$	

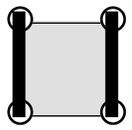
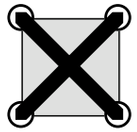
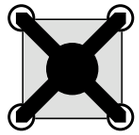
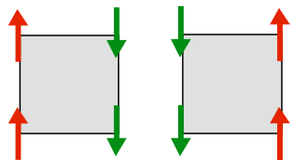
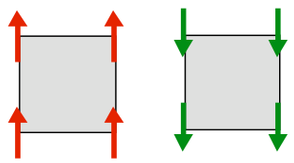
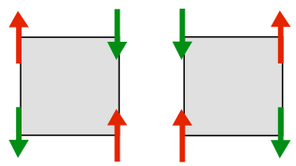
- Connected spins form a cluster and have to be flipped together
- Very simple and deterministic for Heisenberg model

Ising-like ferromagnet

$$H_{XXZ} = -\frac{J_{xz}}{2}\sum_{\langle i,j\rangle}(S_i^+ S_j^- + S_i^- S_j^+) - J_z \sum_{\langle i,j\rangle} S_i^z S_j^z$$

with $0 \le J_{xy} \le J_z$

$$W(C) = \sum_G W(C,G) = \sum_G \Delta(C,G) V(G)$$

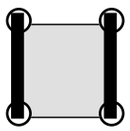
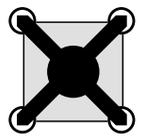
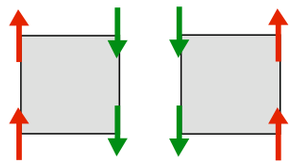
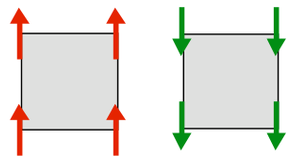
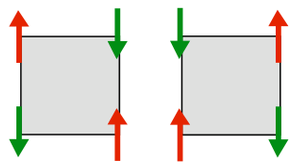
$\Delta(C,G)$				$W(C)$
	1	0	0	$1 - (J_z/4)\ d\tau$
	1	1	1	$1 + (J_z/4)\ d\tau$
	0	1	0	$(J_{xy}/2)\ d\tau$
$V(G)$	$1-(J_z/4)\ d\tau$	$(J_{xy}/2)\ d\tau$	$(J_z - J_{xy})/2\ d\tau$	

- Now 4-spin freezing graph is needed: connects (freezes) loops

Ising ferromagnet

$$H_{\text{Ising}} = -J\sum_{\langle i,j\rangle} S_i^z S_j^z = -\frac{J}{4}\sum_{\langle i,j\rangle}\sigma_i\sigma_j$$

$$W(C) = \sum_G W(C,G) = \sum_G \Delta(C,G)V(G)$$

$\Delta(C,G)$			$W(C)$
	1	0	1- $(J/4)\ d\tau$
	1	1	1+ $(J/4)\ d\tau$
	0	0	0
$V(G)$	$1-(J/4)\ d\tau$	$(J/2)\ d\tau$	

- Two spins are frozen if there is any freezing graph along the world line

$$P_{\text{no freezing}} = \lim_{M\to\infty}(1-(\beta/M)J/2)^M = \exp(-\beta J/2) = \exp(-2\beta J_{classical})$$

- We recover the Swendsen Wang algorithm: probability for no freezing

Extensions of the loop algorithm

- The loop algorithm can also extended to other models
  - Hubbard model (Kawashima Gubernatis, Evertz, PRB 1994)
  - higher spin S > 1/2 (Kawashima, J. Stat. Phys. 1996)
  - t-J model (Ammon, Evertz, Kawashima, Troyer, PRB 1998)
  - SU(N) models (Harada and Kawashima, JPSJ, 2001)
  - dissipative quantum models (Werner and Troyer, PRL 2005)
  - Josepshon junction arrays (Werner and Troyer, PRL 2005)
  - ...

3. Stochastic Series Expansion (SSE)

Path integral representation

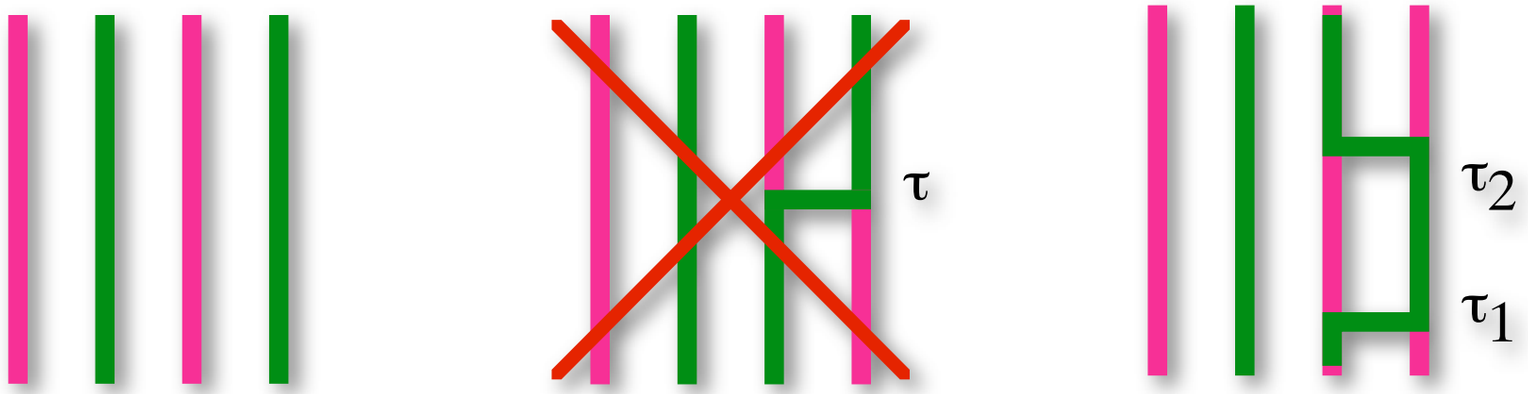
- Split the Hamiltonian into diagonal term H_0 and perturbation V
- Then perform time-dependent perturbation theory

$$H = H_0 + V, \quad H_0 = \sum_{<i,j>} J_{ij}^z S_i^z S_j^z - \sum_i h S_i^z, \quad V = \sum_{<i,j>} J_{ij}^{xy} (S_i^x S_j^x + S_i^y S_j^y)$$

$$Z = \mathrm{Tr}(e^{-\beta H}) = \mathrm{Tr}(e^{-\beta H_0} \mathrm{T} e^{-\int_0^\beta d\tau V(\tau)})$$

$$Z = \mathrm{Tr}(e^{-\beta H_0} (1 - \int_0^\beta d\tau V(\tau) + \int_0^\beta d\tau_1 \int_{\tau_1}^\beta d\tau_2 V(\tau_1) V(\tau_2) + \ldots))$$

- Each term is represented by a diagram (world line configuration)



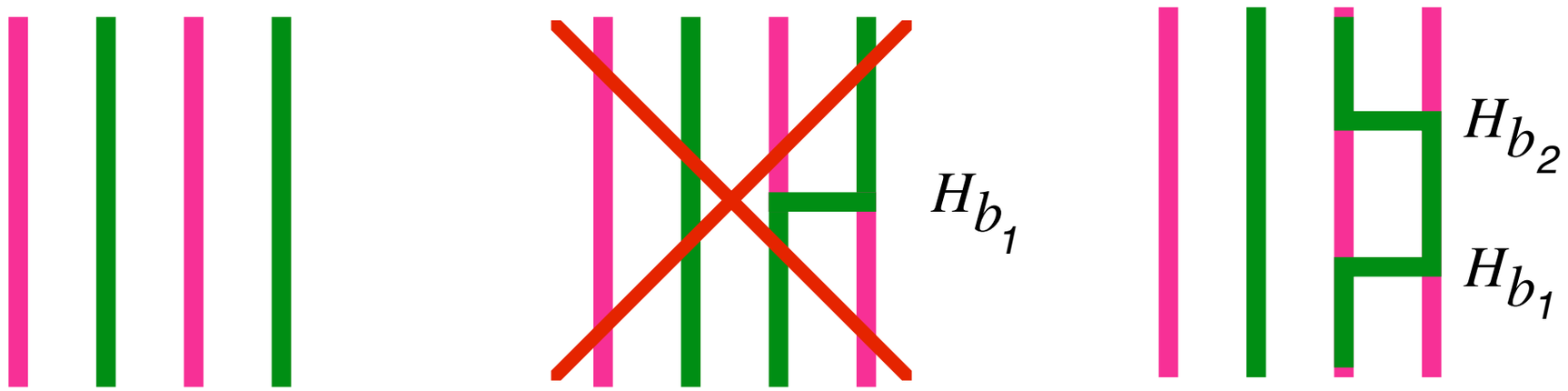
Stochastic Series Expansion

- based on high temperature expansion, developed by Sandvik

$$Z = \mathrm{Tr}(e^{-\beta H}) = \sum_{n=0}^{\infty} (-\beta)^n \, \mathrm{Tr}(H^n)$$

$$= \sum_{n=0}^{\infty} \frac{\beta^n}{n!} \sum_{|\alpha\rangle} \sum_{(b_1,\ldots,b_n)} \langle \alpha | \prod_{i=1}^{n} (-H_{b_i}) | \alpha \rangle$$

$$\text{with } H = \sum_i H_i$$



- Similar to world line representation but without times assigned

Splitting the Hamiltonian for SSE

- Break up the Hamiltonian into offdiagonal and diagonal bond terms

$$H = \sum_{\langle i,j \rangle} H^o_{(i,j)} + H^d_{(i,j)}$$

- Example: Heisenberg antiferromagnet

$$H_{XXZ} = \frac{J_{xz}}{2} \sum_{\langle i,j \rangle} (S_i^+ S_j^- + S_i^- S_j^+) + J_z \sum_{\langle i,j \rangle} S_i^z S_j^z - h \sum_i S_1^z$$

convert site terms into bond terms

$$= \frac{J_{xz}}{2} \sum_{\langle i,j \rangle} (S_i^+ S_j^- + S_i^- S_j^+) + J_z \sum_{\langle i,j \rangle} S_i^z S_j^z - \frac{h}{z} \sum_{\langle i,j \rangle} \left(S_i^z + S_j^z \right)$$

split into diagonal and offdiagonal bond terms

$$= \sum_{\langle i,j \rangle} H^o_{(i,j)} + \sum_{\langle i,j \rangle} H^d_{(i,j)}$$

with

$$H^o_{(i,j)} = \frac{J_{xz}}{2} (S_i^+ S_j^- + S_i^- S_j^+)$$

$$H^d_{(i,j)} = J_z S_i^z S_j^z - \frac{h}{z} \left(S_i^z + S_j^z \right)$$

Ensuring positivity of diagonal bond weights

- Recall the SSE expansion:

$$Z = \sum_{n=0}^{\infty} \frac{\beta^n}{n!} \sum_{|\alpha\rangle} \sum_{(b_1,\ldots,b_n)} \langle \alpha | \prod_{i=1}^{n} (-H_{b_i}) | \alpha \rangle$$

- Negative matrix elements of H are the weights
  - Need to make all matrix elements non-positive
  - Diagonal matrix elements: subtract an energy shift

$$H^d_{(i,j)} = J_z S^z_i S^z_j - \frac{h}{z}\left(S^z_i + S^z_j\right) - C \qquad C \geq \left|\frac{J_z}{4}\right| + |h|$$

$$H^d_{(i,j)} = \begin{pmatrix} \frac{J_z}{4} + h - C & 0 & 0 & 0 \\ 0 & -\frac{J_z}{4} - C & 0 & 0 \\ 0 & 0 & -\frac{J_z}{4} - C & 0 \\ 0 & 0 & 0 & \frac{J_z}{4} - h - C \end{pmatrix}$$

Positivity of off-diagonal bond weights

- Energy shift will not help with off-diagonal matrix elements

$$H^o_{(i,j)} = \frac{J_{xz}}{2}(S^+_i S^-_j + S^-_i S^+_j)$$

$$H^o_{(i,j)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{J_{xy}}{2} & 0 \\ 0 & \frac{J_{xy}}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

- Ferromagnet ($J_{xy} < 0$) is no problem
- Antiferromagnet on bipartite lattice
  - perform a gauge transformation on one sublattice

$$\frac{J_{xy}}{2}\left(S^+_i S^-_j + S^-_i S^+_j\right) \xrightarrow{S^\pm_i \to (-1)^{|i|} S^\pm_i} -\frac{J_{xy}}{2}\left(S^+_i S^-_j + S^-_i S^+_j\right)$$

- Frustrated antiferromagnet: we have a sign problem

Fixed length operator strings

- SSE sampling requires variable length n operator strings

$$Z = \sum_{n=0}^{\infty} \frac{\beta^n}{n!} \sum_{|\alpha\rangle} \sum_{(b_1,\ldots,b_n)} \langle \alpha | \prod_{i=1}^{n} (-H_{b_i}) | \alpha \rangle \qquad H_{b_i} \in \bigcup_{\langle i,j \rangle} \left\{ H^d_{(i,j)}, H^o_{(i,j)} \right\}$$

- Extend operator string to fixed length by adding extra identity operators:

n … number of non-unit operators

$$Z = \sum_{n=0}^{\Lambda} \sum_{|\alpha\rangle} \sum_{(b_1,\ldots,b_\Lambda)} \frac{(\Lambda - n)! \beta^n}{\Lambda!} \langle \alpha | \prod_{i=1}^{\Lambda} (-H_{b_i}) | \alpha \rangle \qquad H_{id} = -1$$

$$H_{b_i} \in \{ H_{id} \} \cup \bigcup_{\langle i,j \rangle} \left\{ H^d_{(i,j)}, H^o_{(i,j)} \right\}$$

  - pick Λ large enough during thermalization

- And now just perform updates

Step 1: Local diagonal updates

- Recall the weight of a configuration:

$$Z = \sum_{n=0}^{\Lambda} \sum_{|\alpha\rangle} \sum_{(b_1,\ldots,b_\Lambda)} \frac{(\Lambda - n)!\beta^n}{\Lambda!} \langle \alpha | \prod_{i=1}^{\Lambda} (-H_{b_i}) | \alpha \rangle$$

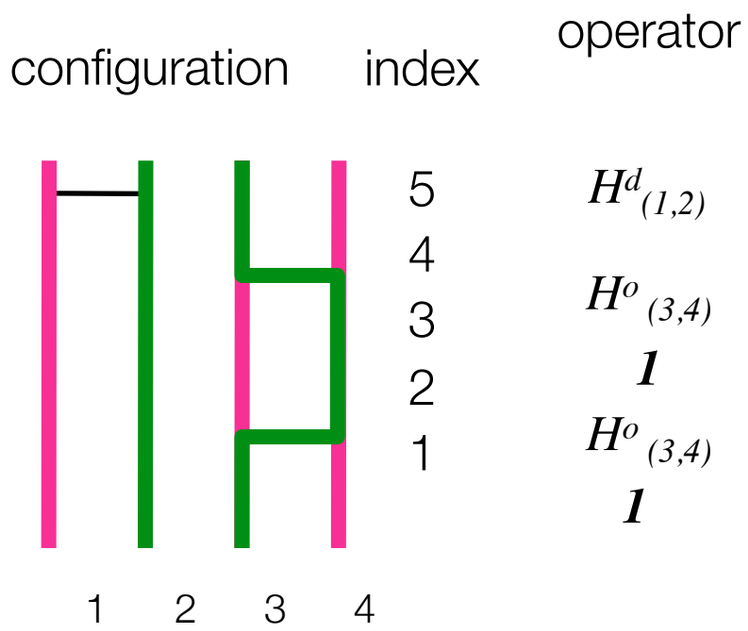
- Walk through operator string

  - Propose to insert diagonal operators instead of unit operators

$$P[1 \to H^d_{(i,j)}] = \min\left(1, \frac{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}{\Lambda - n}\right)$$

  - Propose to remove diagonal operators

$$P[H^d_{(i,j)} \to 1] = \min\left(1, \frac{\Lambda - n + 1}{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}\right)$$



Step 1: Local diagonal updates

- Recall the weight of a configuration:

$$Z = \sum_{n=0}^{\Lambda} \sum_{|\alpha\rangle} \sum_{(b_1,\ldots,b_\Lambda)} \frac{(\Lambda - n)!\beta^n}{\Lambda!} \langle \alpha | \prod_{i=1}^{\Lambda} (-H_{b_i}) | \alpha \rangle$$

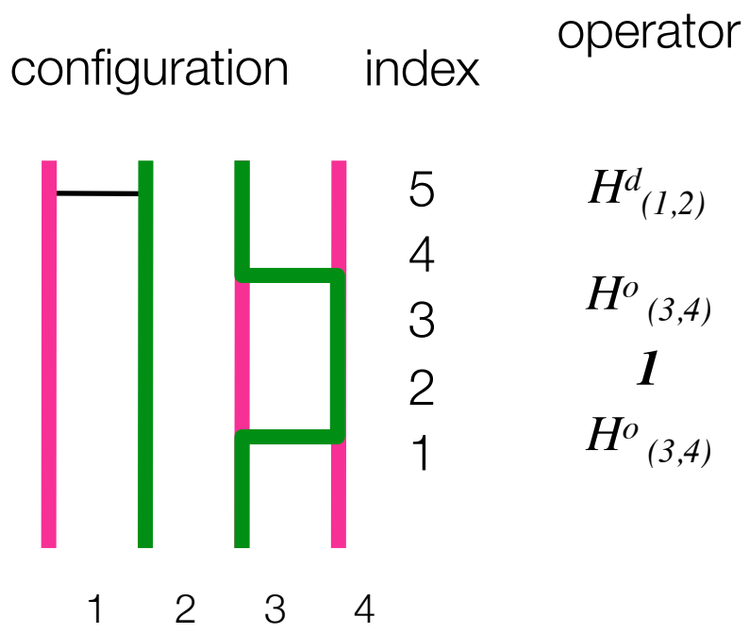
- Walk through operator string

  - Propose to insert diagonal operators instead of unit operators

$$P[1 \to H^d_{(i,j)}] = \min\left(1, \frac{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}{\Lambda - n}\right)$$

  - Propose to remove diagonal operators

$$P[H^d_{(i,j)} \to 1] = \min\left(1, \frac{\Lambda - n + 1}{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}\right)$$



Step 1: Local diagonal updates

- Recall the weight of a configuration:

$$Z = \sum_{n=0}^{\Lambda} \sum_{|\alpha\rangle} \sum_{(b_1,\ldots,b_\Lambda)} \frac{(\Lambda - n)!\beta^n}{\Lambda!} \langle \alpha | \prod_{i=1}^{\Lambda} (-H_{b_i}) | \alpha \rangle$$

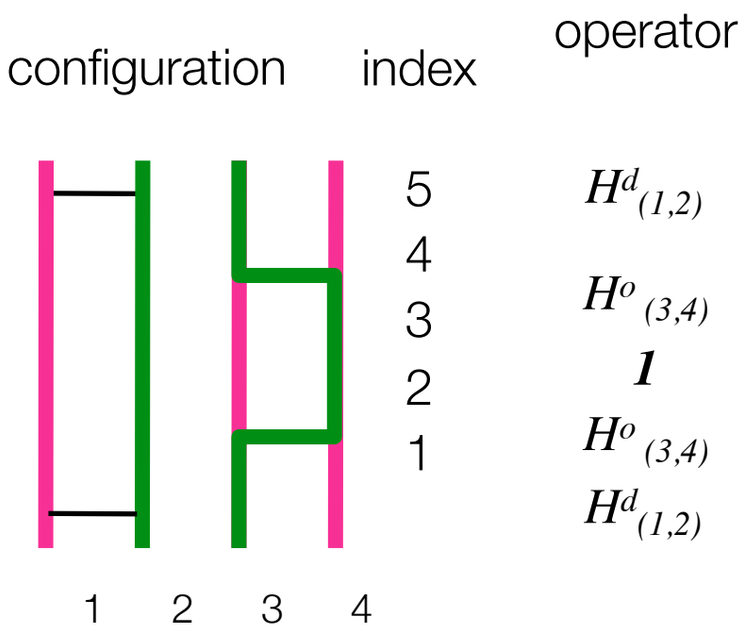
- Walk through operator string

  - Propose to insert diagonal operators instead of unit operators

$$P[1 \to H^d_{(i,j)}] = \min\left(1, \frac{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}{\Lambda - n}\right)$$

  - Propose to remove diagonal operators

$$P[H^d_{(i,j)} \to 1] = \min\left(1, \frac{\Lambda - n + 1}{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}\right)$$



Step 1: Local diagonal updates

- Recall the weight of a configuration:

$$Z = \sum_{n=0}^{\Lambda} \sum_{|\alpha\rangle} \sum_{(b_1,\ldots,b_\Lambda)} \frac{(\Lambda - n)!\beta^n}{\Lambda!} \langle \alpha | \prod_{i=1}^{\Lambda} (-H_{b_i}) | \alpha \rangle$$

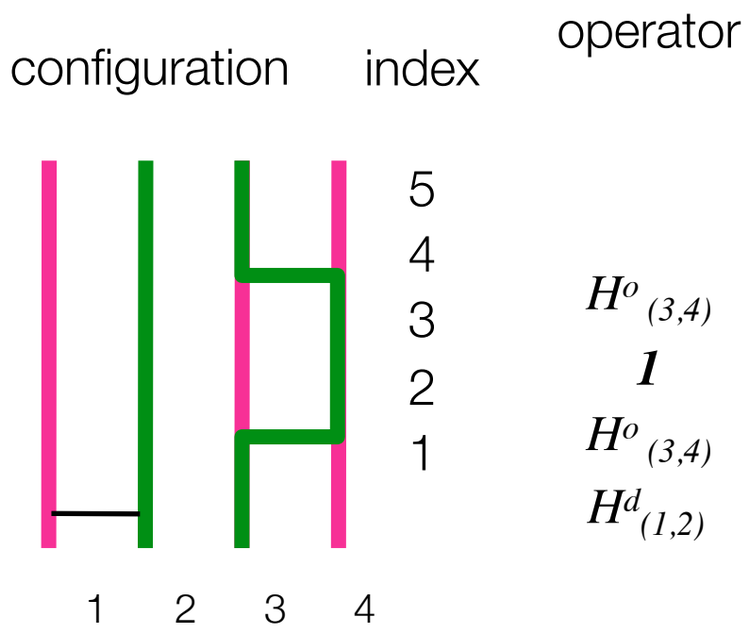
- Walk through operator string

  - Propose to insert diagonal operators instead of unit operators

$$P[1 \to H^d_{(i,j)}] = \min\left(1, \frac{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}{\Lambda - n}\right)$$

  - Propose to remove diagonal operators

$$P[H^d_{(i,j)} \to 1] = \min\left(1, \frac{\Lambda - n + 1}{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}\right)$$



Step 1: Local diagonal updates

- Recall the weight of a configuration:

$$Z = \sum_{n=0}^{\Lambda} \sum_{|\alpha\rangle} \sum_{(b_1,\ldots,b_\Lambda)} \frac{(\Lambda - n)!\beta^n}{\Lambda!} \langle \alpha | \prod_{i=1}^{\Lambda} (-H_{b_i}) | \alpha \rangle$$

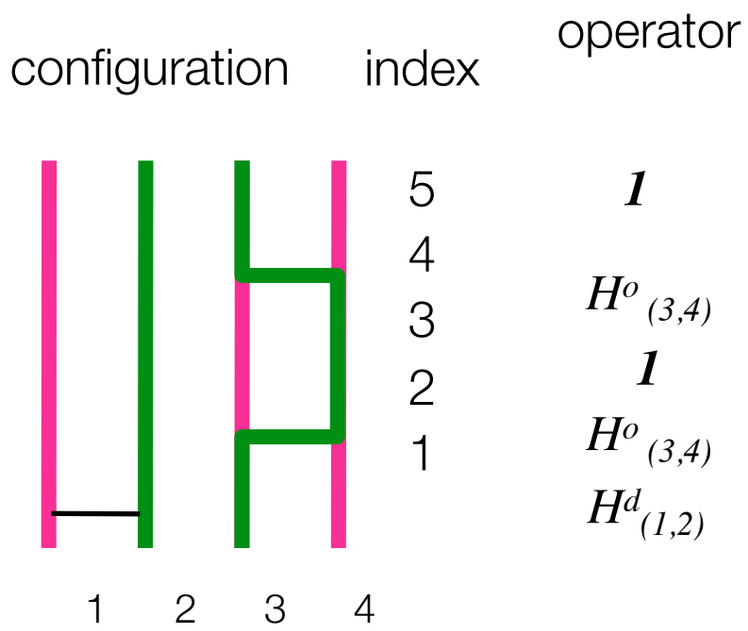
- Walk through operator string

  - Propose to insert diagonal operators instead of unit operators

$$P[1 \to H^d_{(i,j)}] = \min\left(1, \frac{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}{\Lambda - n}\right)$$

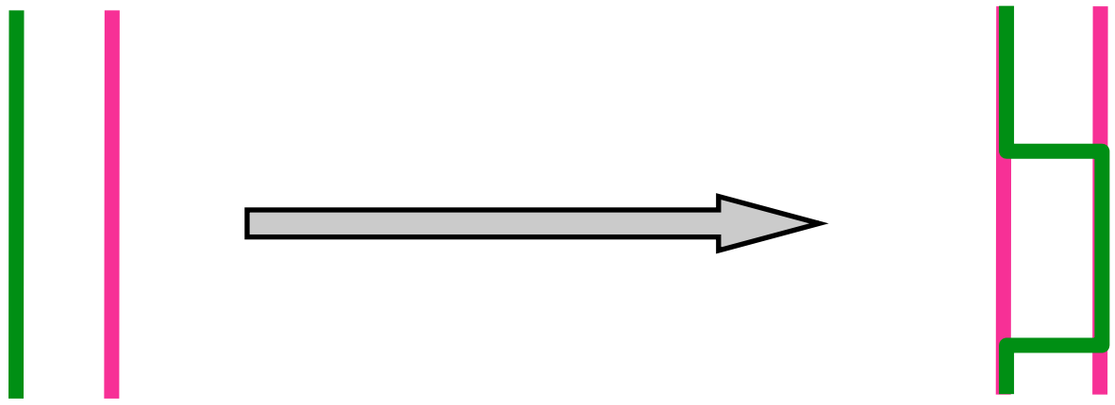
  - Propose to remove diagonal operators

$$P[H^d_{(i,j)} \to 1] = \min\left(1, \frac{\Lambda - n + 1}{\beta N_{bonds} \langle \alpha | H^d_{(i,j)} | \alpha \rangle}\right)$$



Step 2: Offdiagonal updates (local)

- Are very easy, can be done in any representation



- Problem 1: only local changes
  - **Nonergodic**
  - No change of magnetization, particle number, winding number
- Problem 2:
  - **Critical slowing down**
- Solution: loop algorithm

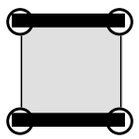
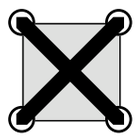
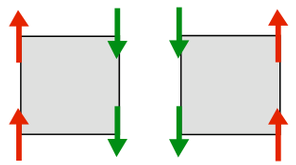
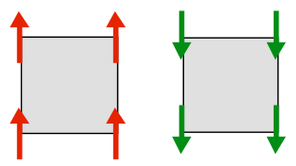
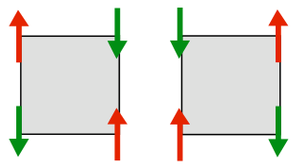
Loop building rules in SSE

- Example: XY-like AFM:

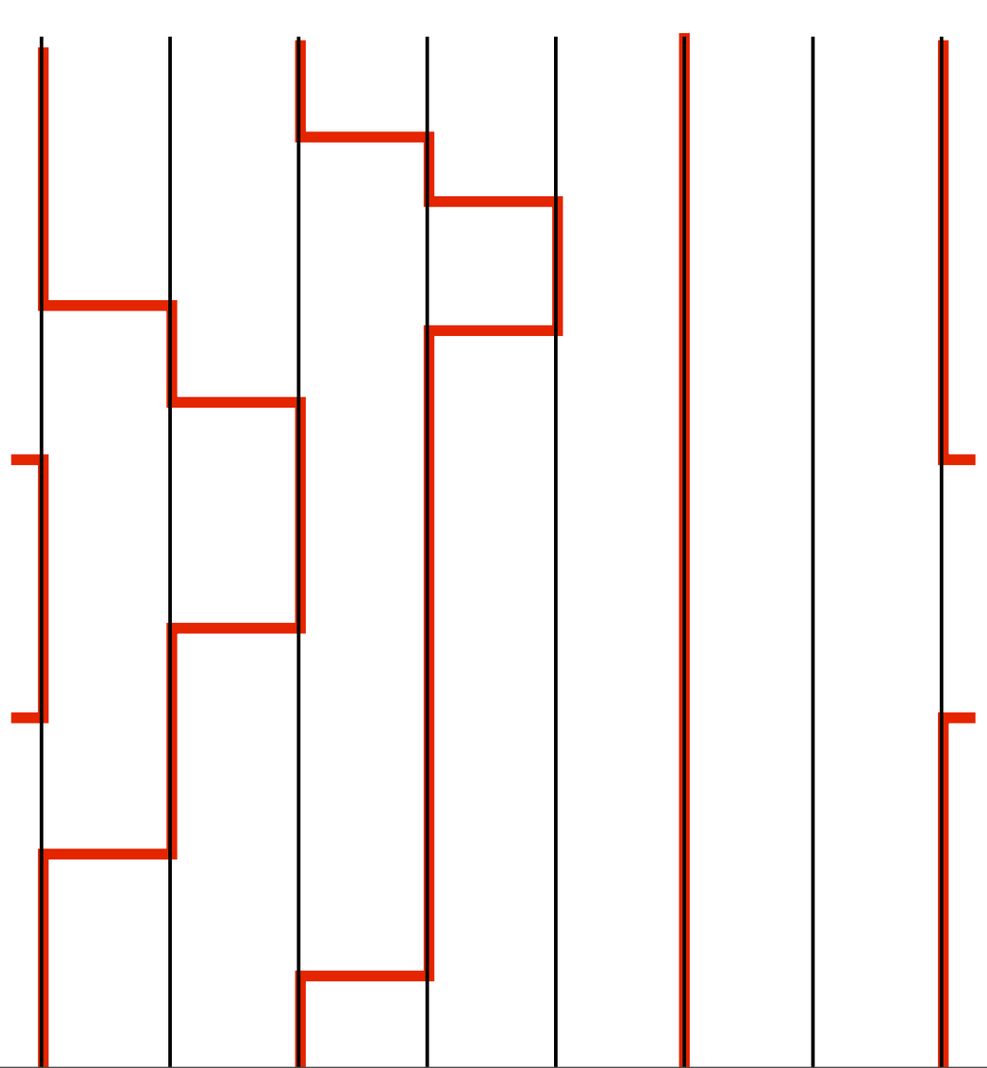
$$H_{XXZ} = \frac{J_{xz}}{2}\sum_{\langle i,j\rangle}(S_i^+S_j^- + S_i^-S_j^+) + J_z\sum_{\langle i,j\rangle}S_i^zS_j^z$$

with $0 \le J_z \le J_{xy}$

$$W(C) = \sum_G W(C,G) = \sum_G \Delta(C,G)V(G)$$

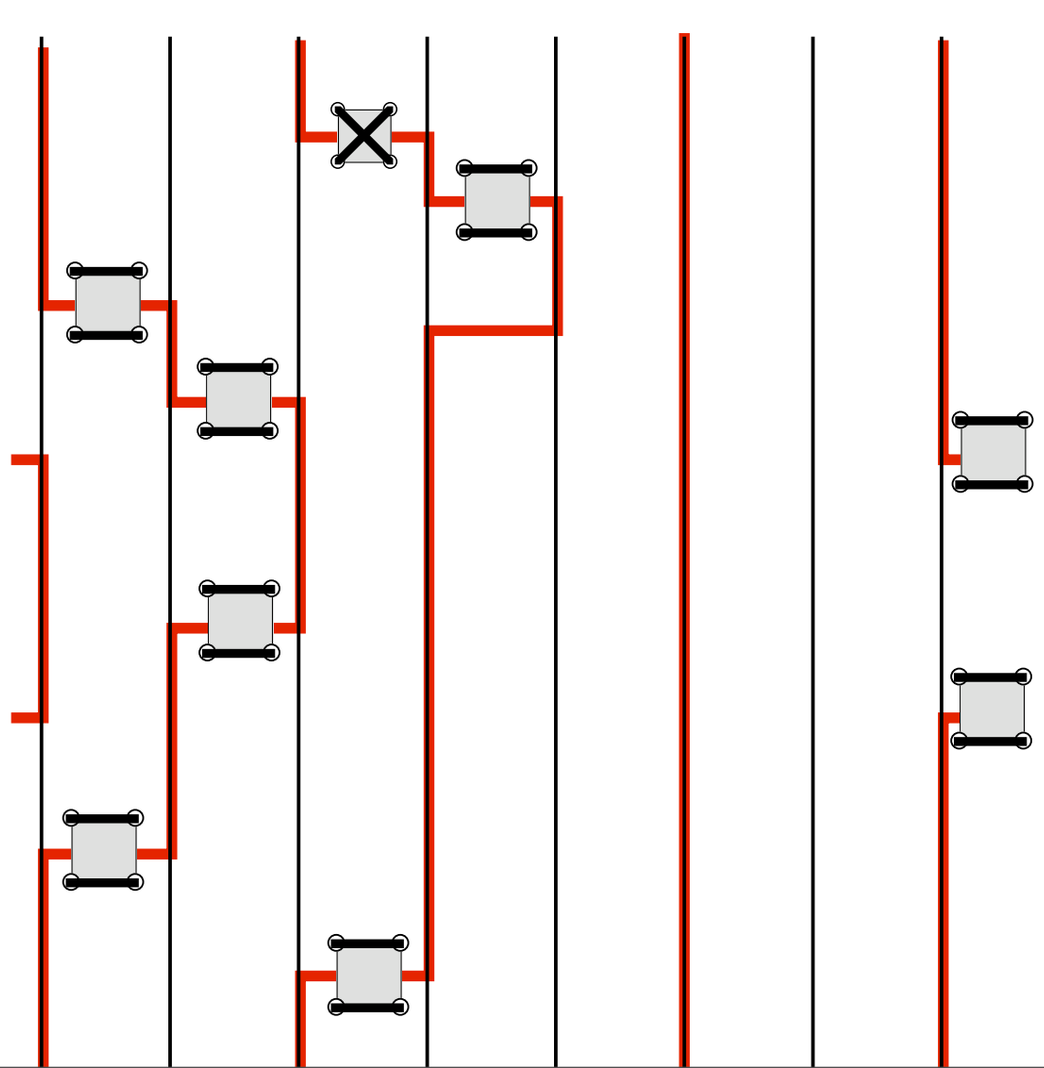
$\Delta(C,G)$			$W(C)$
	1	0	$J_z/2$
	0	0	0
	1	1	$J_{xy}/2$
$W(G)$	$J_z/2$	$(J_{xy}- J_z)/2$	

Loop updates in path integrals



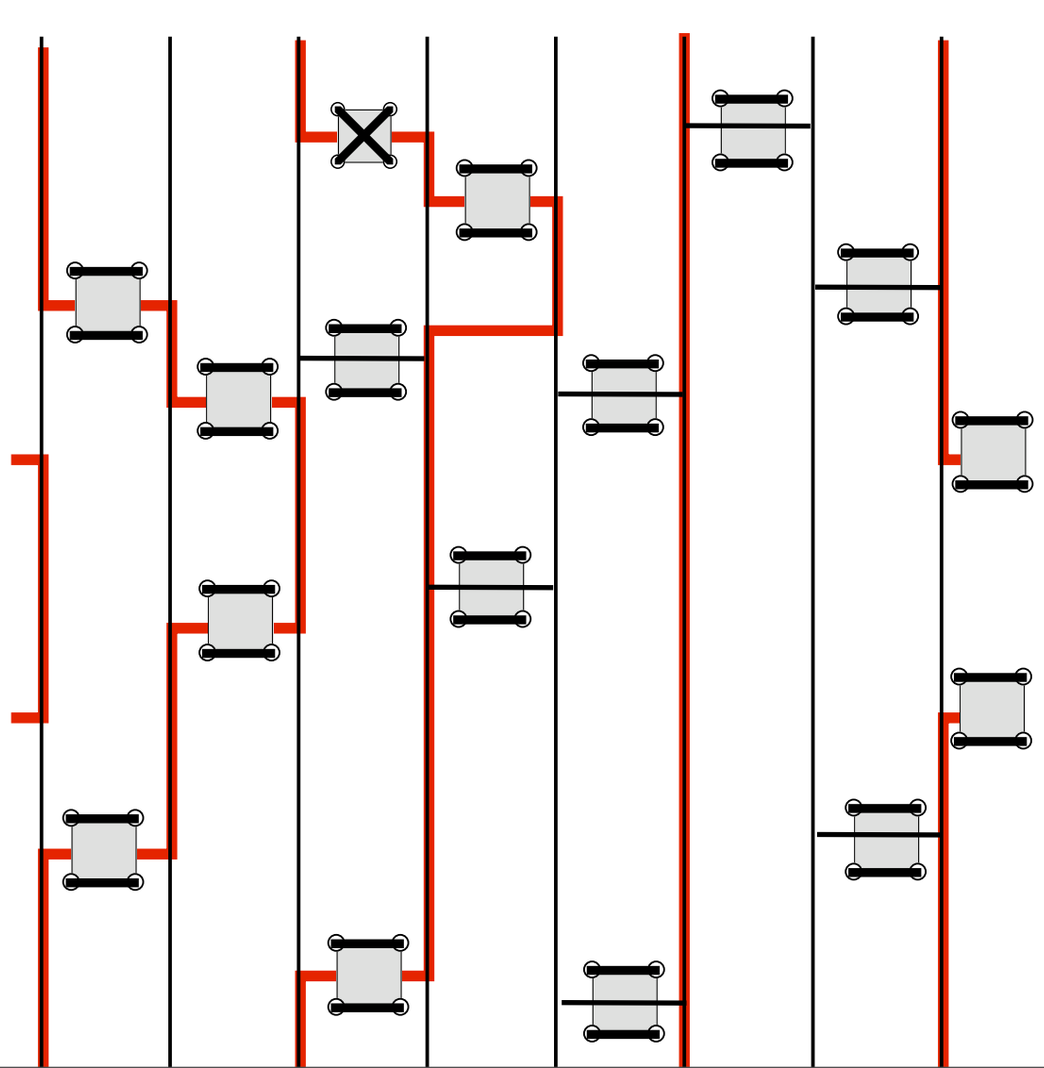
Loop updates in path integrals

1. Define “breakups” (graphs) for exchange processes



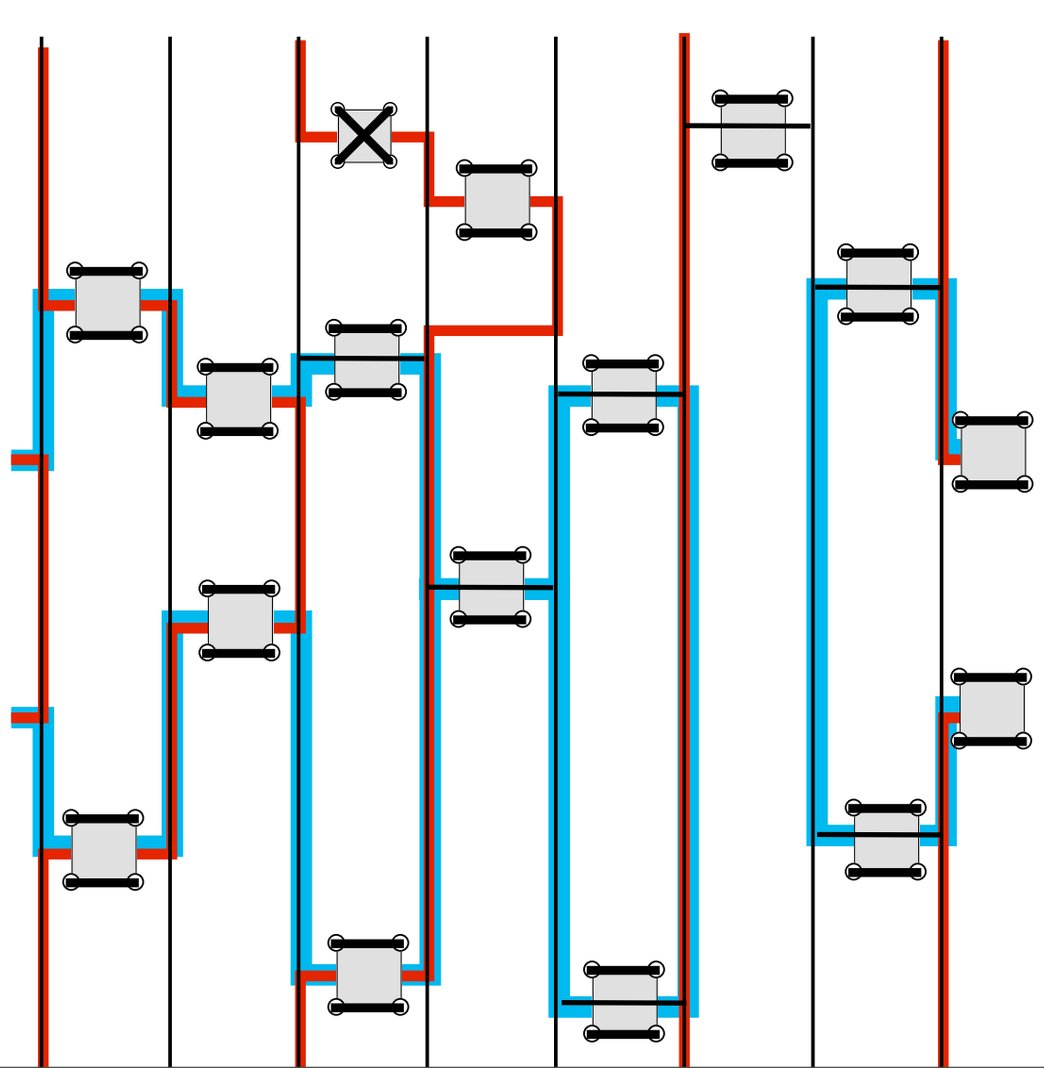
Loop updates in path integrals

1. Define “breakups” (graphs) for exchange processes
2. Insert “decay” graphs



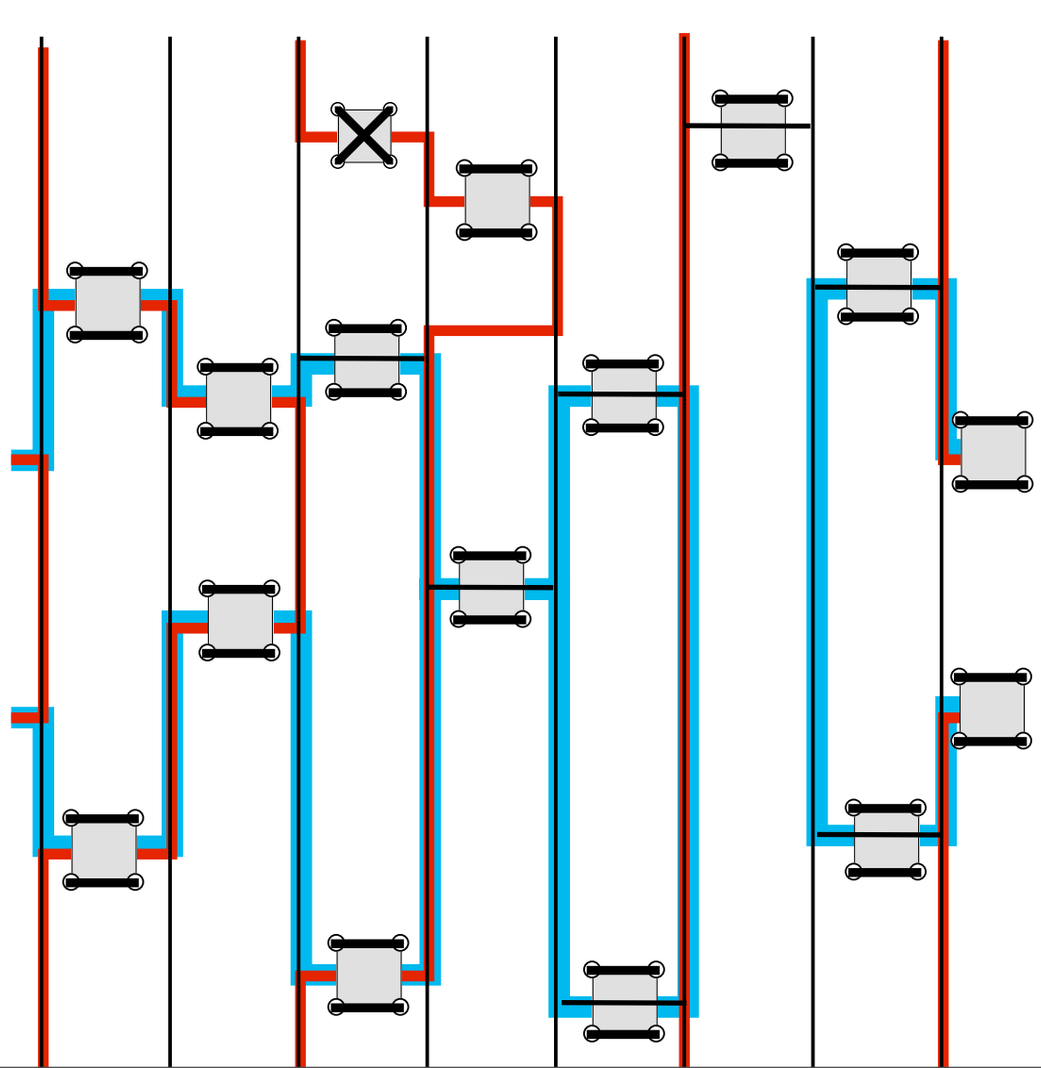
Loop updates in path integrals

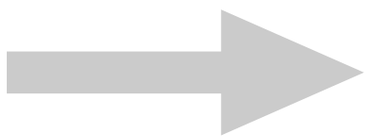
1. Define “breakups” (graphs) for exchange processes
2. Insert “decay” graphs
3. Build and flip one or more loops

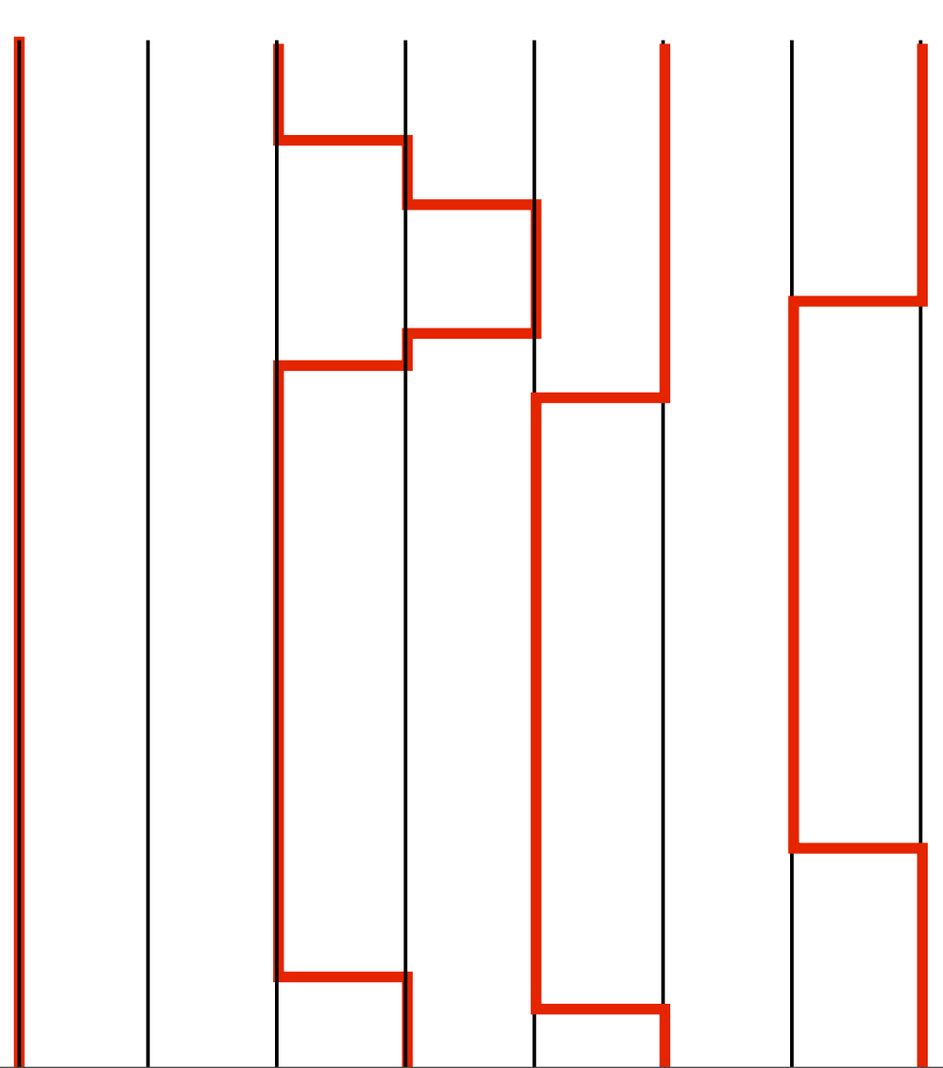


Loop updates in path integrals

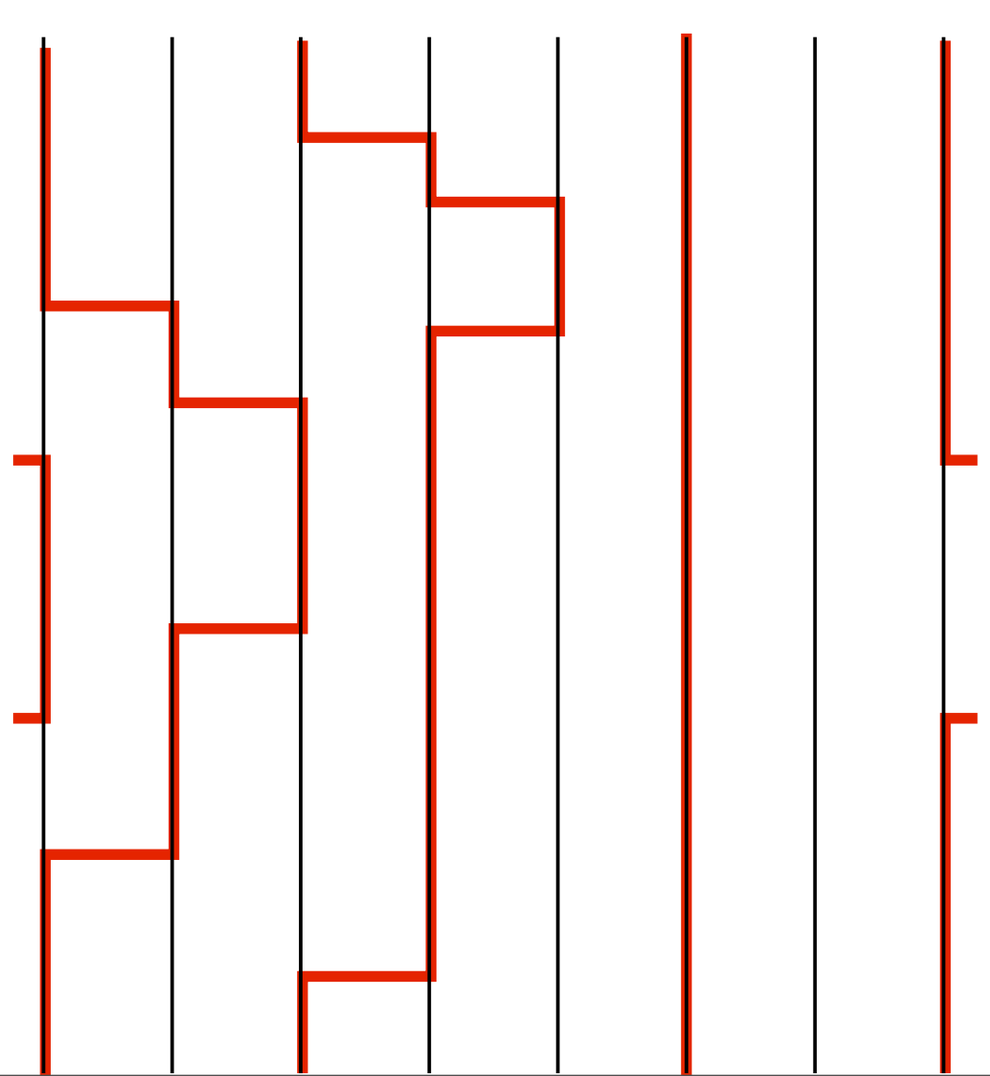
1. Define “breakups” (graphs) for exchange processes
2. Insert “decay” graphs
3. Build and flip one or more loops





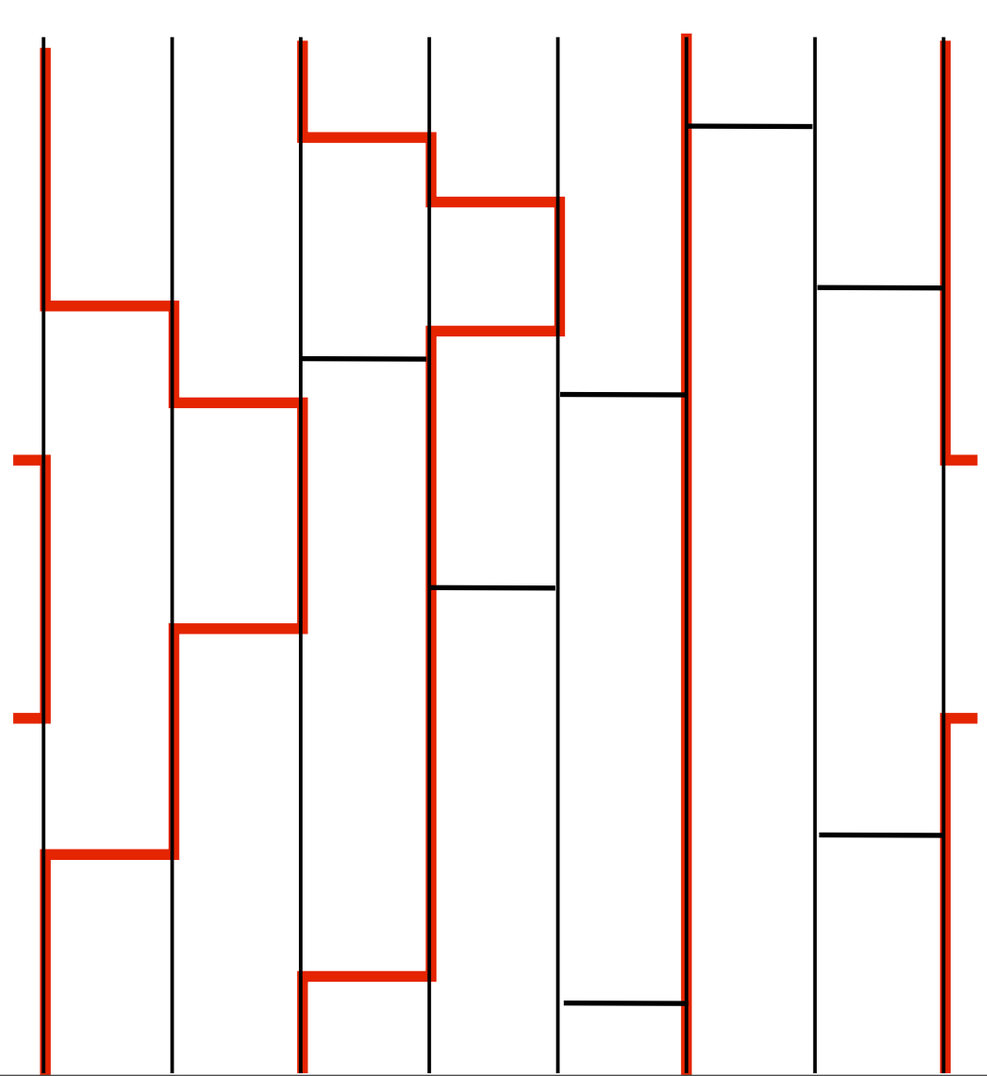


Loop updates in SSE



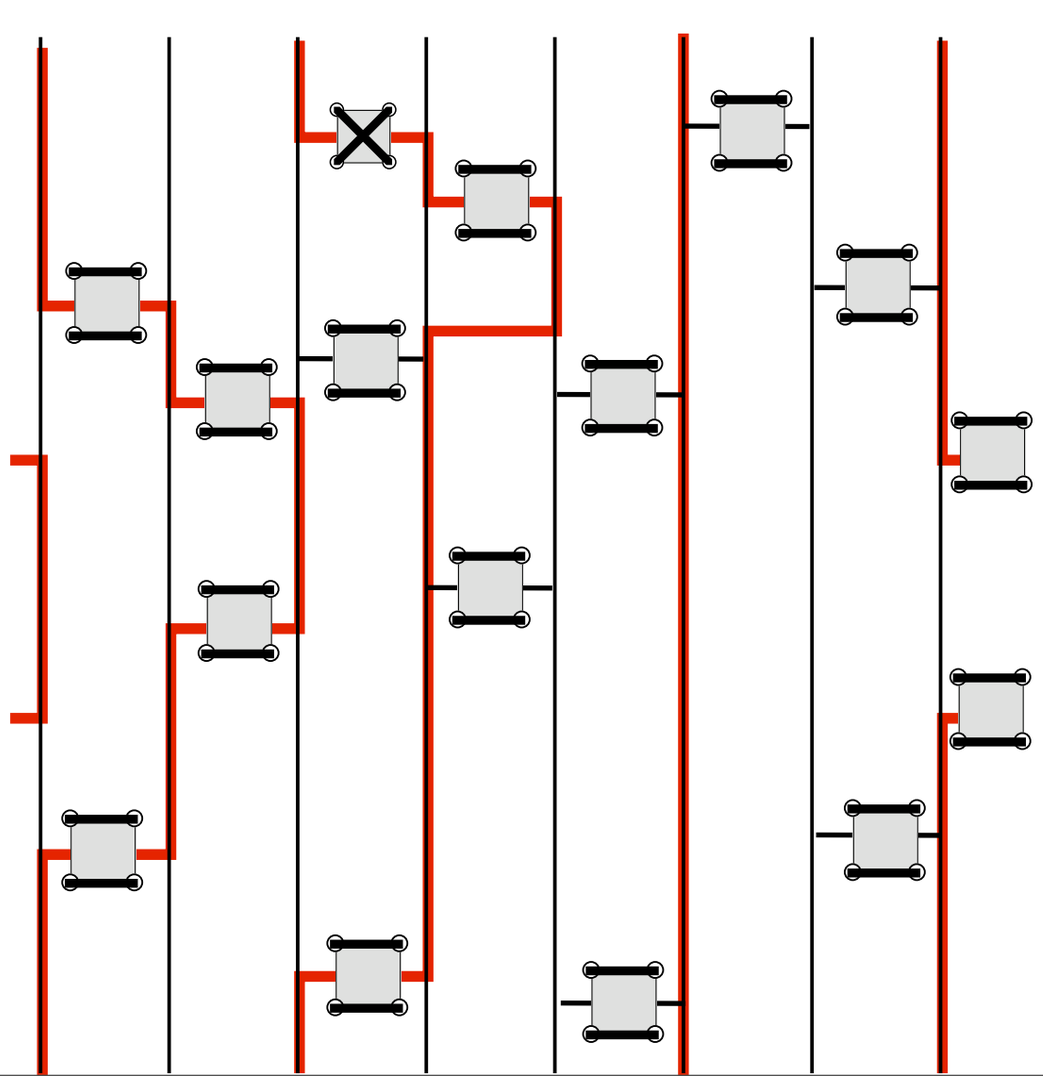
Loop updates in SSE

1. Insert/remove diagonal operators



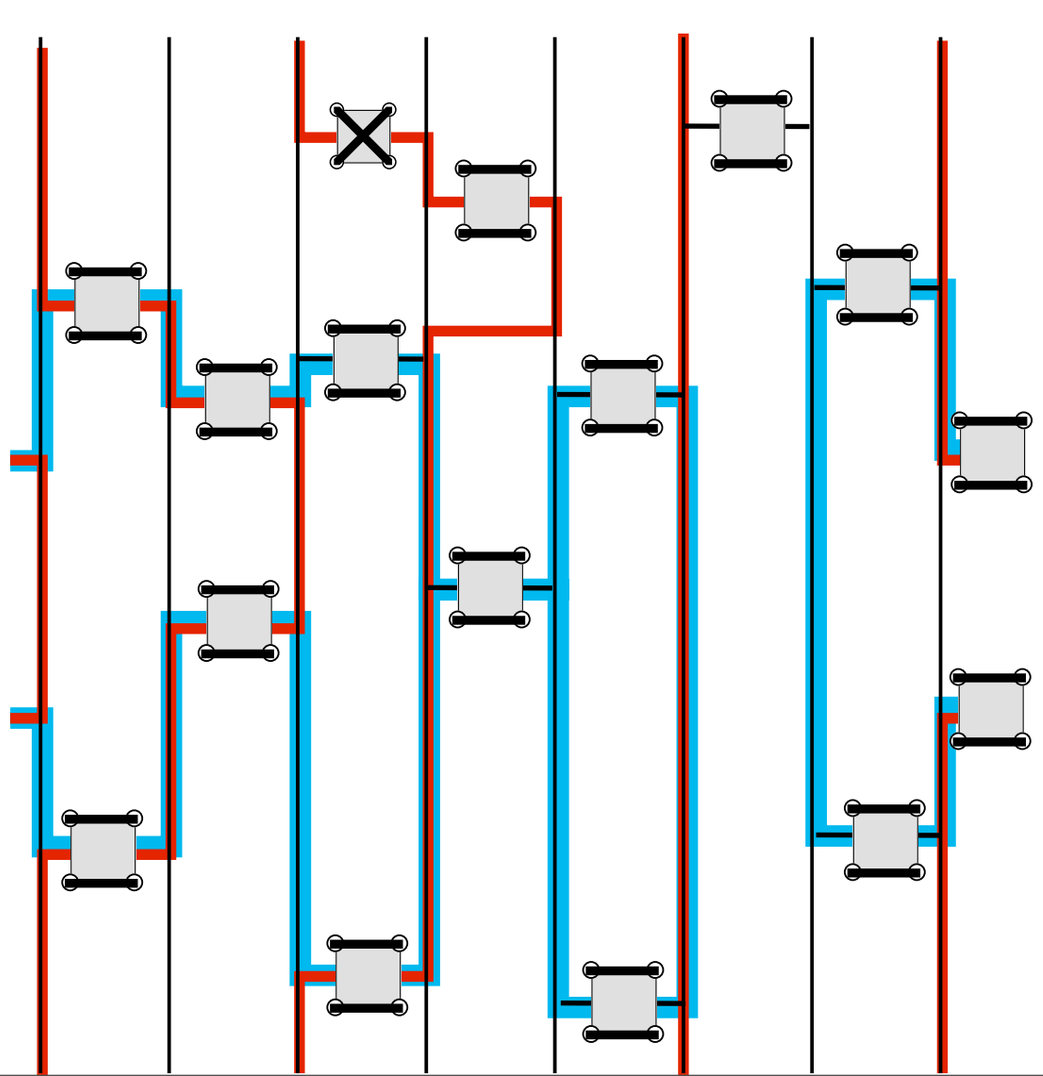
Loop updates in SSE

1. Insert/remove diagonal operators
2. Decide “breakups” (graphs)



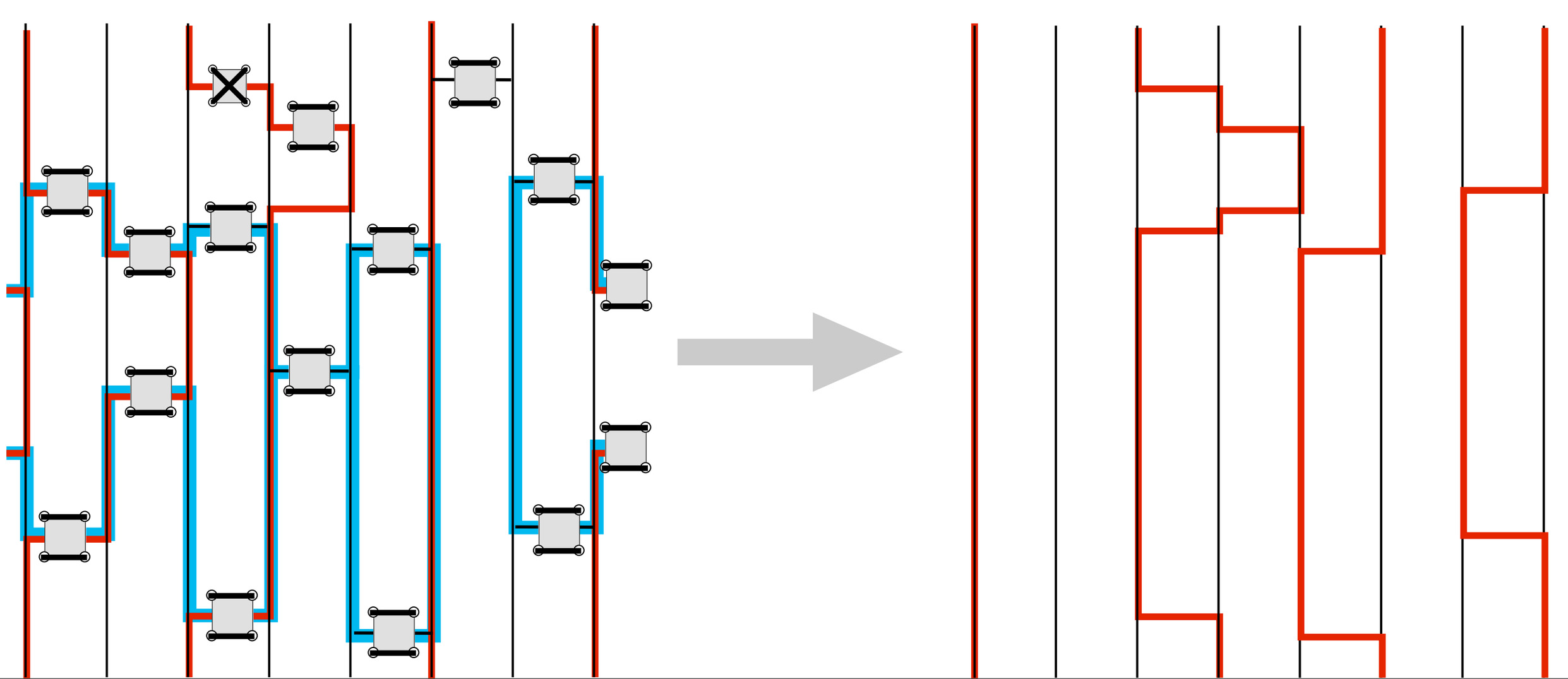
Loop updates in SSE

1. Insert/remove diagonal operators
2. Decide “breakups” (graphs)
3. Build and flip one or more loops



Loop updates in SSE

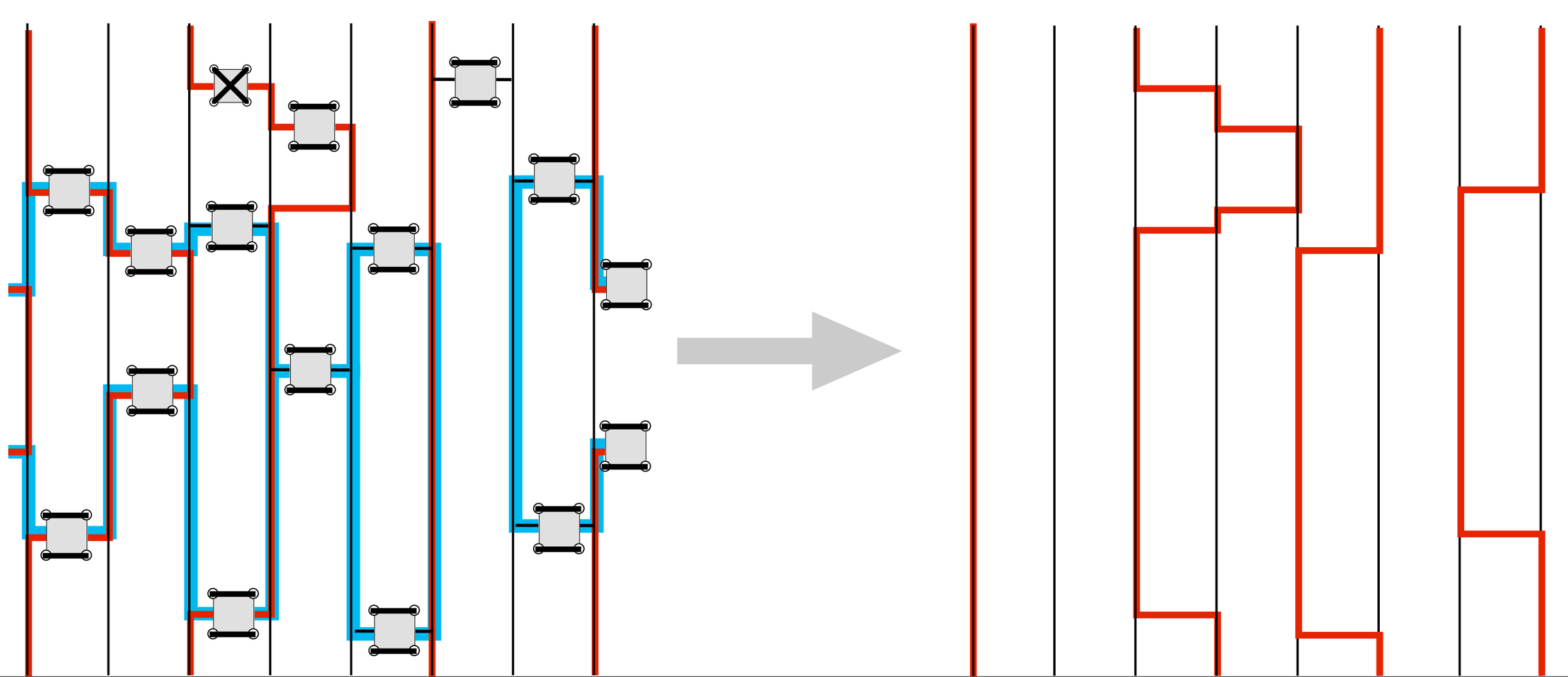
1. Insert/remove diagonal operators
2. Decide “breakups” (graphs)
3. Build and flip one or more loops



Loop updates in SSE

1. Insert/remove diagonal operators
2. Decide “breakups” (graphs)
3. Build and flip one or more loops

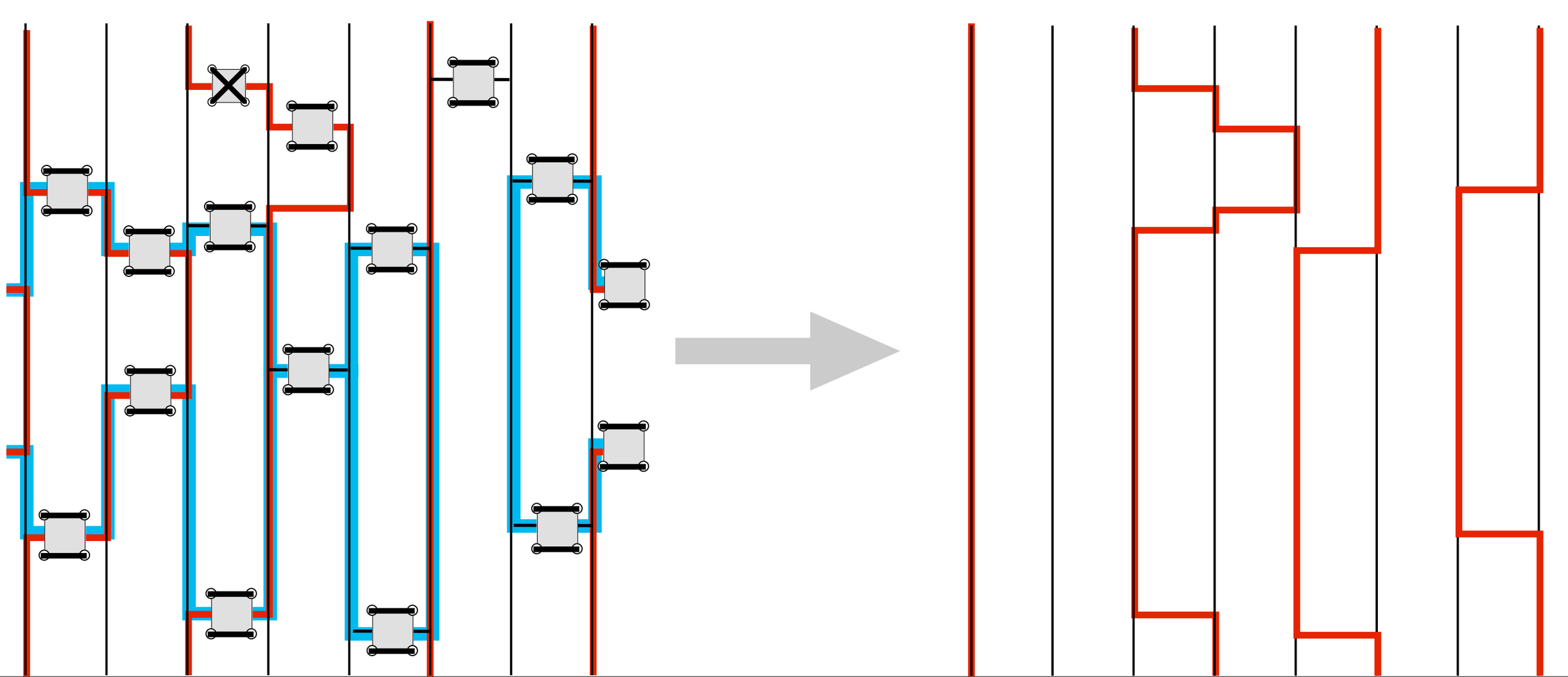
Similarity is no chance:



Loop updates in SSE

1. Insert/remove diagonal operators
2. Decide “breakups” (graphs)
3. Build and flip one or more loops

Similarity is no chance: an exact mapping exists



Measurements

- All diagonal operators can be measured easily
  - diagonal spin correlation functions
  - magnetization
  - diagonal components of the energy
- Green's function:
  - The S^+ - S^- off-diagonal Green's function can be measured during loop construction.
- Imaginary time displaced correlation functions can also be calculated and give access to dynamical correlation functions after an analytical continuation.

4. The worm algorithm

Loop algorithm in a magnetic field

- Loop cluster algorithm requires spin inversion symmetry
  - Magnetic field implemented by a-posteriori acceptance rate
- Example: spin dimer at $J = h = 1$

$$H = J\vec{S}_1\vec{S}_2 - h\left(S_1^z + S_2^z\right)$$

Loop algorithm in a magnetic field

- Loop cluster algorithm requires spin inversion symmetry
  - Magnetic field implemented by a-posteriori acceptance rate
- Example: spin dimer at $J = h = 1$

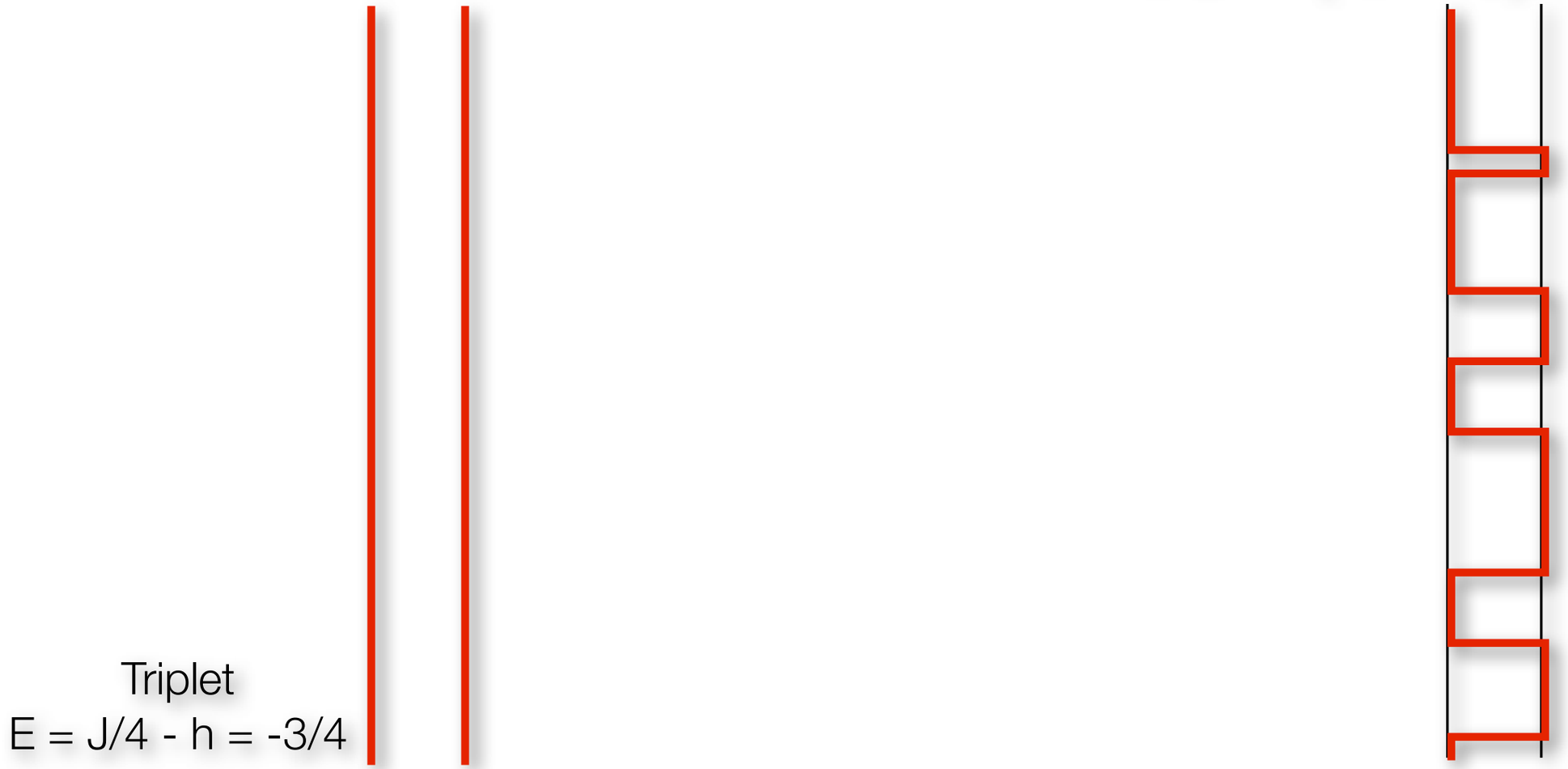
$$H = J\vec{S}_1\vec{S}_2 - h\left(S_1^z + S_2^z\right)$$

Triplet
E = J/4 - h = -3/4

Loop algorithm in a magnetic field

- Loop cluster algorithm requires spin inversion symmetry
  - Magnetic field implemented by a-posteriori acceptance rate
- Example: spin dimer at $J = h = 1$

$$H = J\vec{S}_1\vec{S}_2 - h\left(S_1^z + S_2^z\right)$$



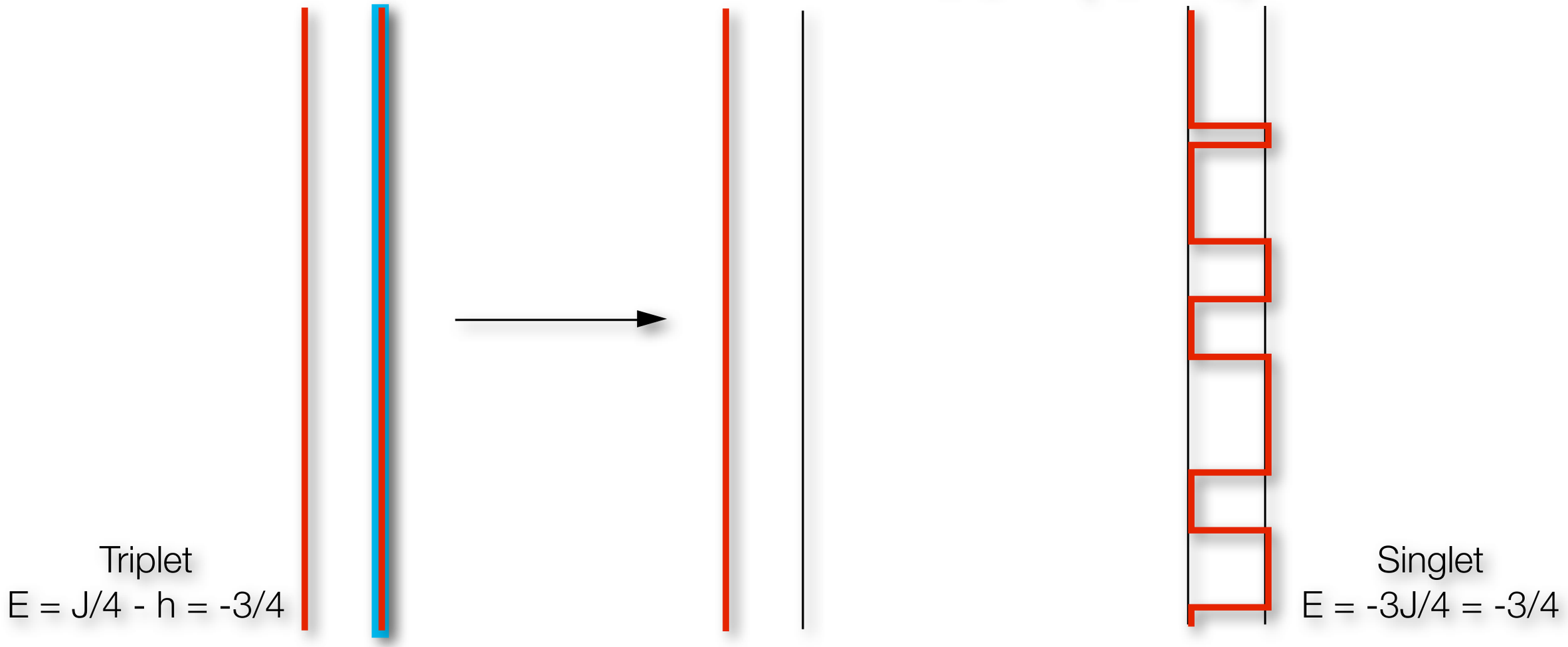
Triplet
E = J/4 - h = -3/4

Singlet
E = -3J/4 = -3/4

Loop algorithm in a magnetic field

- Loop cluster algorithm requires spin inversion symmetry
  - Magnetic field implemented by a-posteriori acceptance rate
- Example: spin dimer at $J = h = 1$

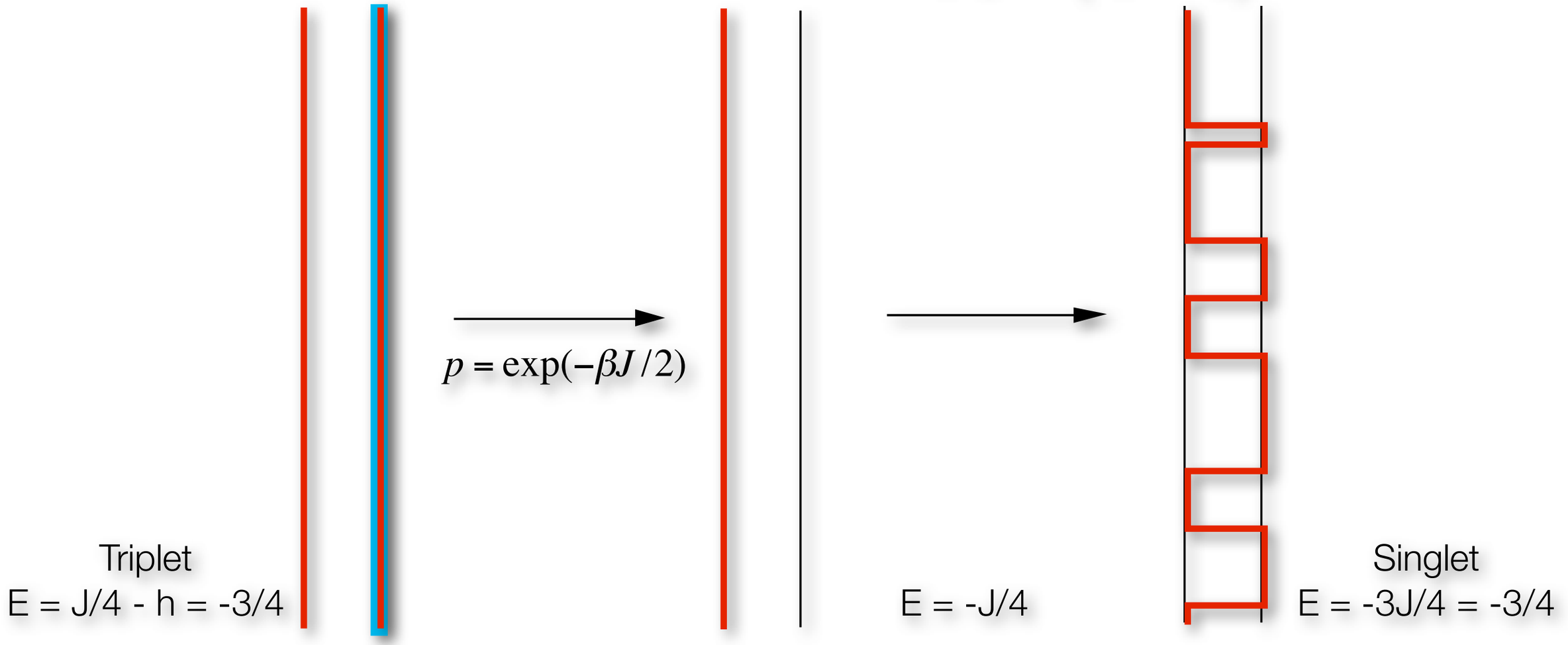
$$H = J\vec{S}_1\vec{S}_2 - h\left(S_1^z + S_2^z\right)$$



Loop algorithm in a magnetic field

- Loop cluster algorithm requires spin inversion symmetry
  - Magnetic field implemented by a-posteriori acceptance rate
- Example: spin dimer at $J = h = 1$

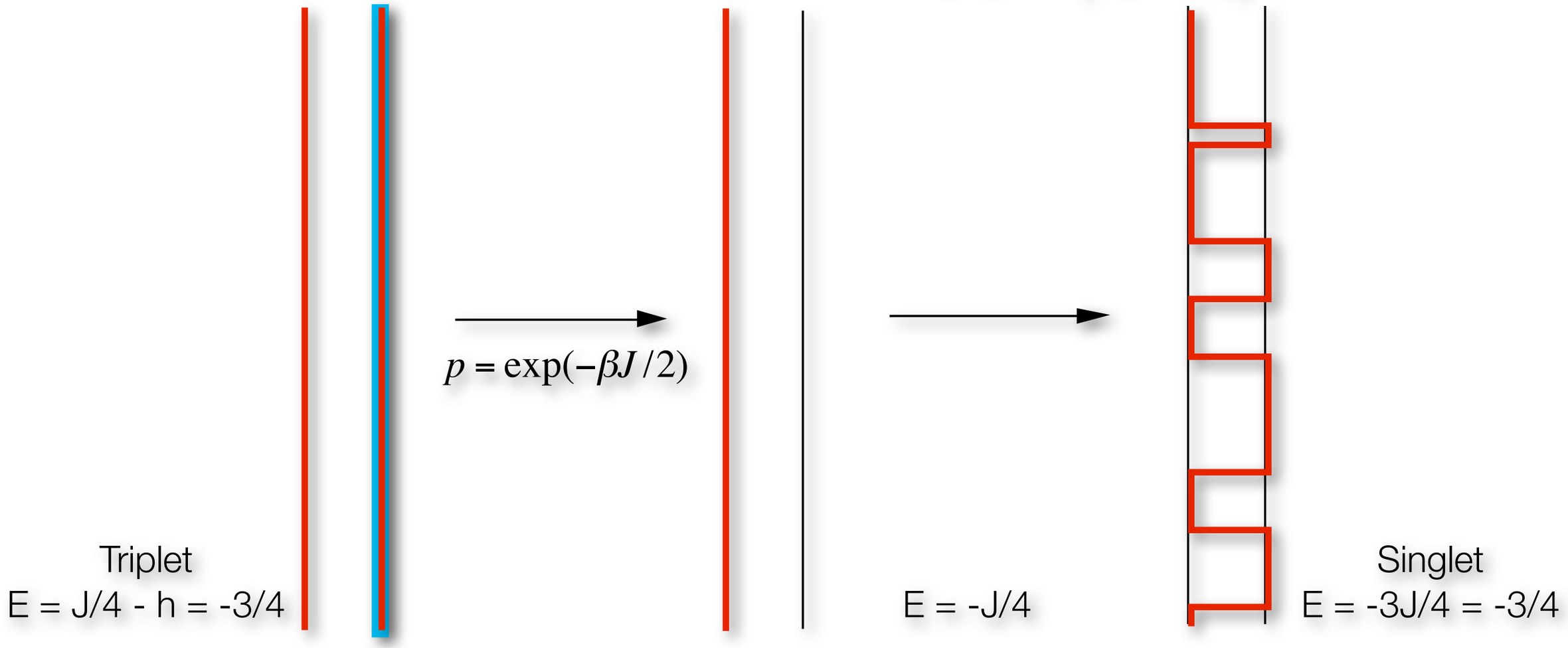
$$H = J\vec{S}_1\vec{S}_2 - h\left(S_1^z + S_2^z\right)$$



Loop algorithm in a magnetic field

- Loop cluster algorithm requires spin inversion symmetry
  - Magnetic field implemented by a-posteriori acceptance rate
- Example: spin dimer at $J = h = 1$

$$H = J\vec{S}_1\vec{S}_2 - h\left(S_1^z + S_2^z\right)$$



Loop algorithm must go through high energy intermediate state

Exponential slowdown

Worm updates (Prokof'ev et al, 1998)

- Break a world line by inserting a pair of creation/annihilation operators

$$H \leftarrow H + \eta \sum_i \left(c_i^\dagger + c_i \right) \qquad H \leftarrow H + \eta \sum_i \left(S_i^+ + S_i^- \right)$$

  - move these operators ("Ira" and "Masha") using local moves
  - until Ira and Masha meet

Worm updates (Prokof'ev et al, 1998)

- Break a world line by inserting a pair of creation/annihilation operators

$$H \leftarrow H + \eta \sum_i \left(c_i^\dagger + c_i \right) \qquad H \leftarrow H + \eta \sum_i \left(S_i^+ + S_i^- \right)$$

  - move these operators ("Ira" and "Masha") using local moves
  - until Ira and Masha meet

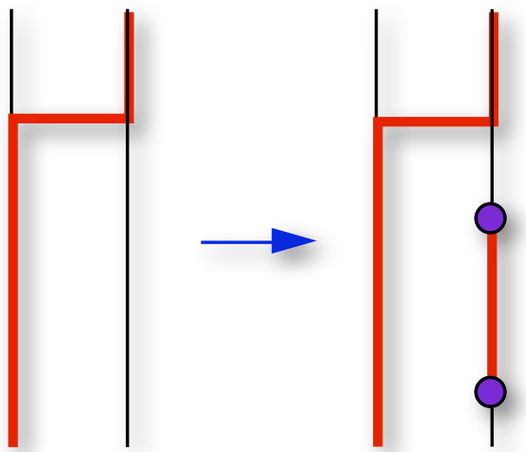
Worm updates (Prokof'ev et al, 1998)

- Break a world line by inserting a pair of creation/annihilation operators

$$H \leftarrow H + \eta \sum_i \left(c_i^\dagger + c_i \right) \qquad H \leftarrow H + \eta \sum_i \left(S_i^+ + S_i^- \right)$$

  - move these operators ("Ira" and "Masha") using local moves
  - until Ira and Masha meet

insert worm

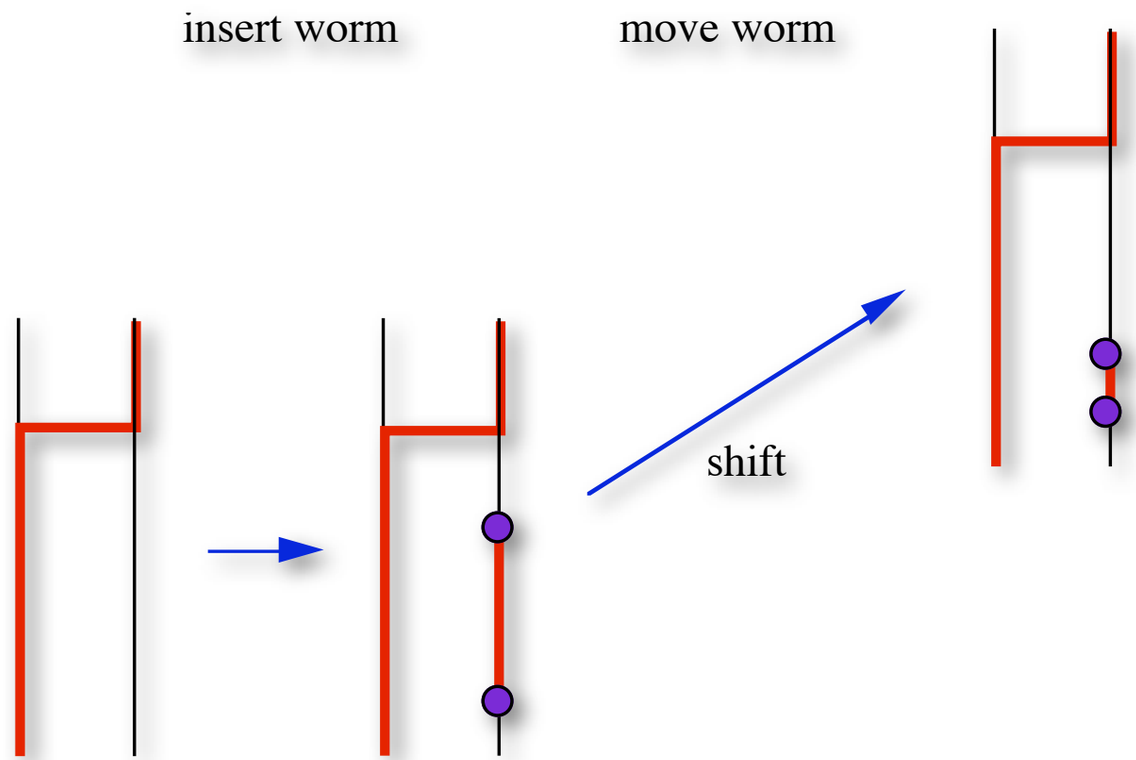


Worm updates (Prokof'ev et al, 1998)

- Break a world line by inserting a pair of creation/annihilation operators

$$H \leftarrow H + \eta \sum_i \left(c_i^\dagger + c_i \right) \qquad H \leftarrow H + \eta \sum_i \left(S_i^+ + S_i^- \right)$$

  - move these operators ("Ira" and "Masha") using local moves
  - until Ira and Masha meet

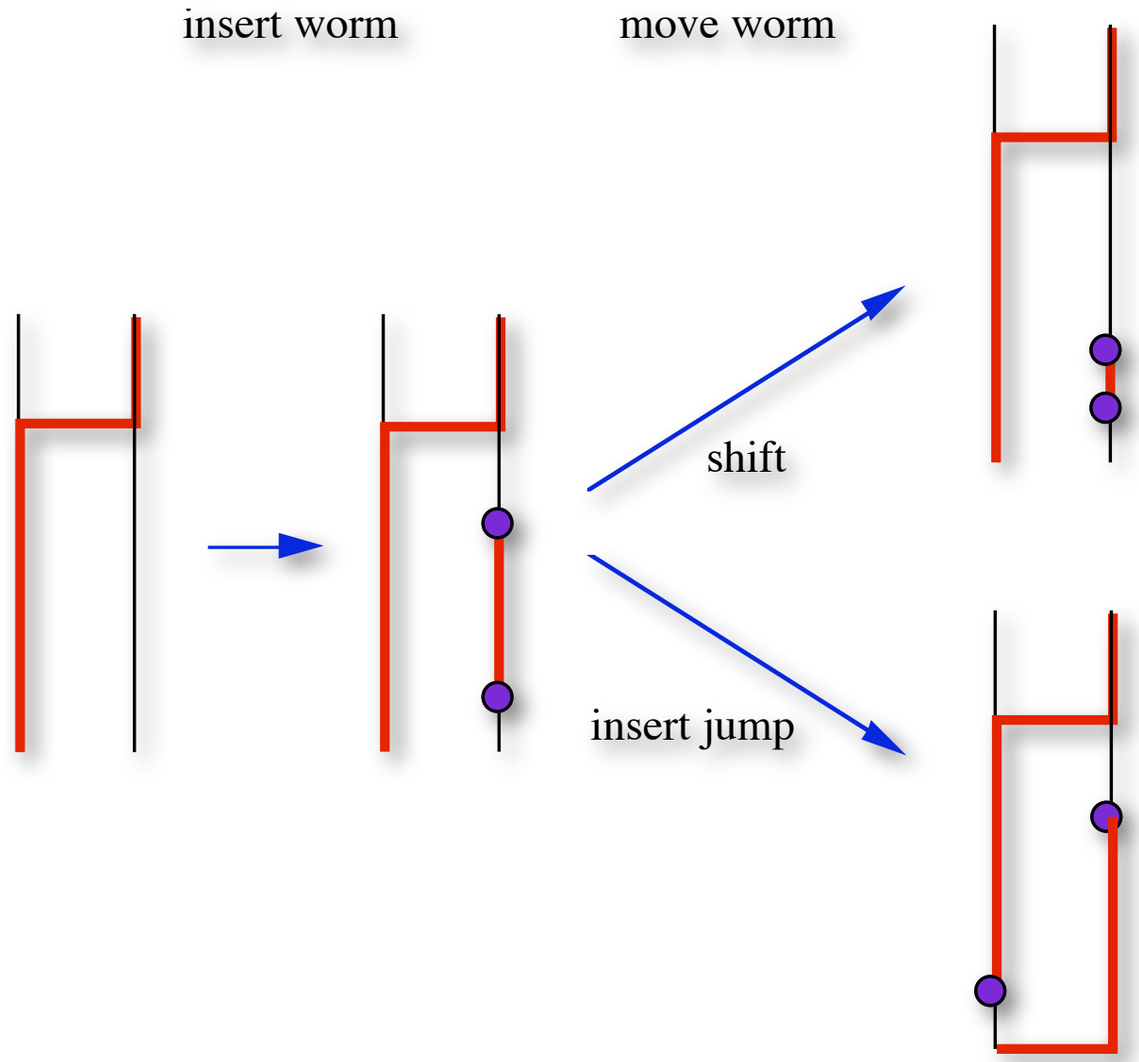


Worm updates (Prokof'ev et al, 1998)

- Break a world line by inserting a pair of creation/annihilation operators

$$H \leftarrow H + \eta \sum_i \left(c_i^\dagger + c_i \right) \qquad H \leftarrow H + \eta \sum_i \left(S_i^+ + S_i^- \right)$$

  - move these operators ("Ira" and "Masha") using local moves
  - until Ira and Masha meet

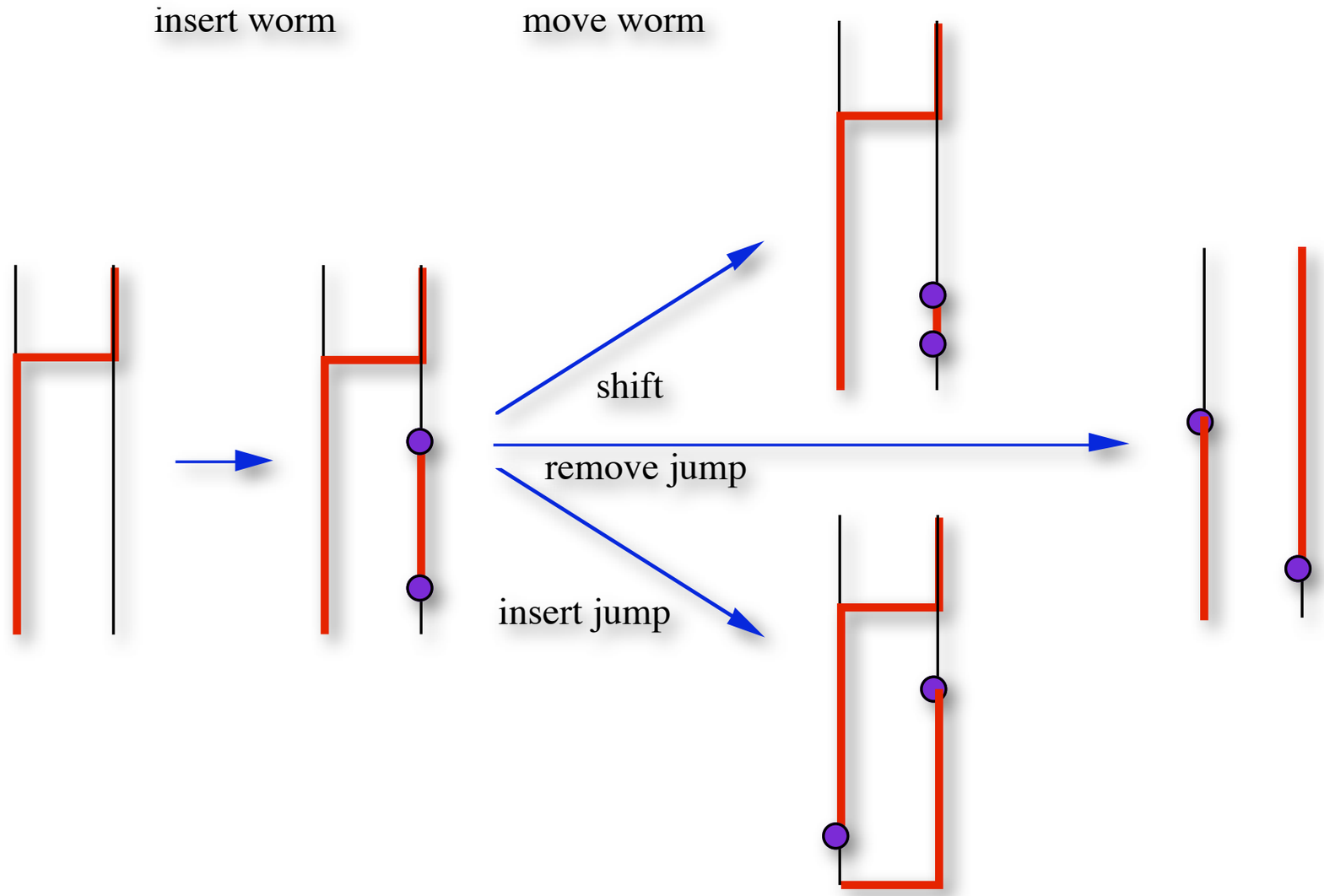


Worm updates (Prokof'ev et al, 1998)

- Break a world line by inserting a pair of creation/annihilation operators

$$H \leftarrow H + \eta \sum_i \left(c_i^\dagger + c_i \right) \qquad H \leftarrow H + \eta \sum_i \left(S_i^+ + S_i^- \right)$$

  - move these operators ("Ira" and "Masha") using local moves
  - until Ira and Masha meet

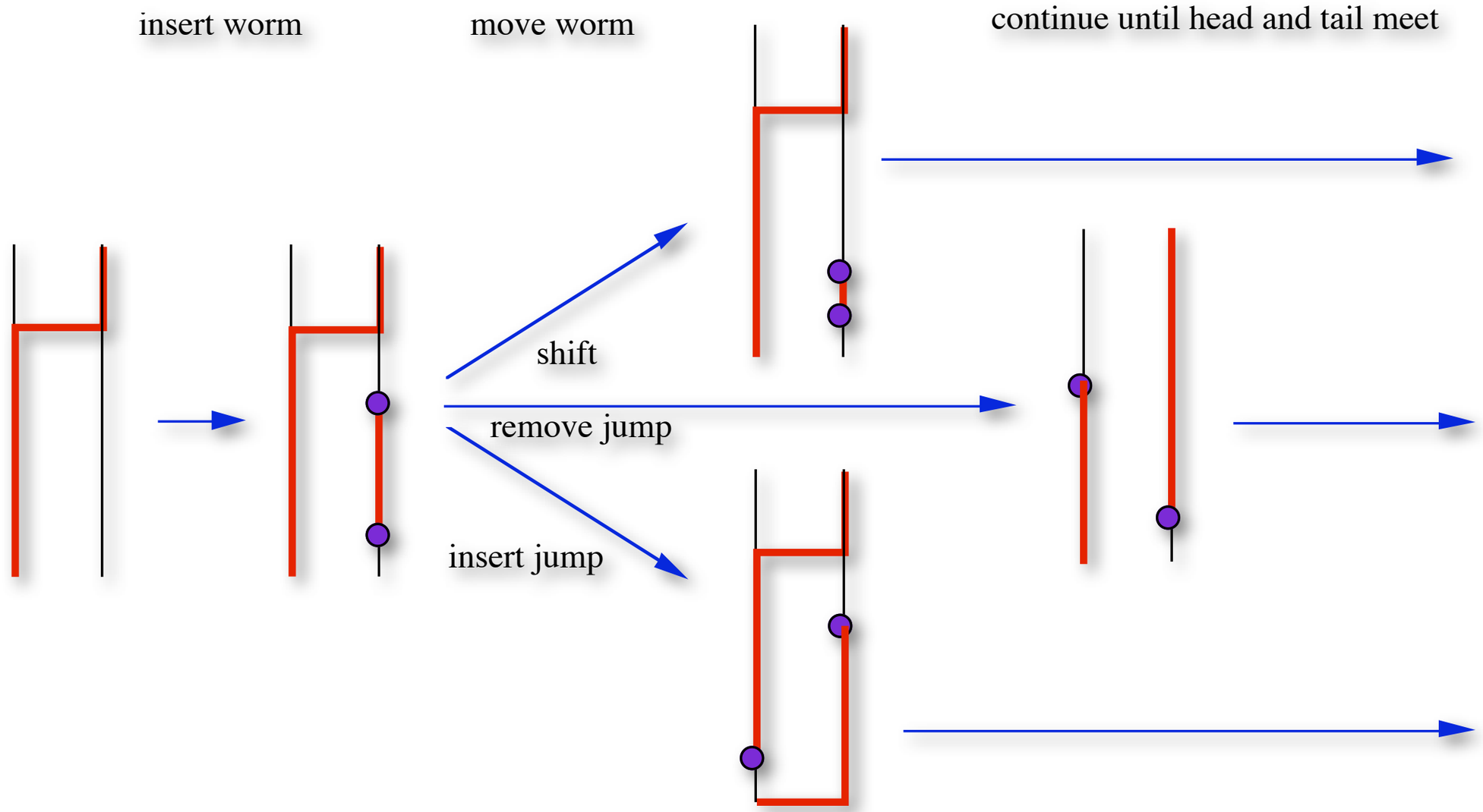


Worm updates (Prokof'ev et al, 1998)

- Break a world line by inserting a pair of creation/annihilation operators

$$H \leftarrow H + \eta \sum_i \left(c_i^\dagger + c_i \right) \qquad H \leftarrow H + \eta \sum_i \left(S_i^+ + S_i^- \right)$$

  - move these operators ("Ira" and "Masha") using local moves
  - until Ira and Masha meet



Worm algorithm in a magnetic field

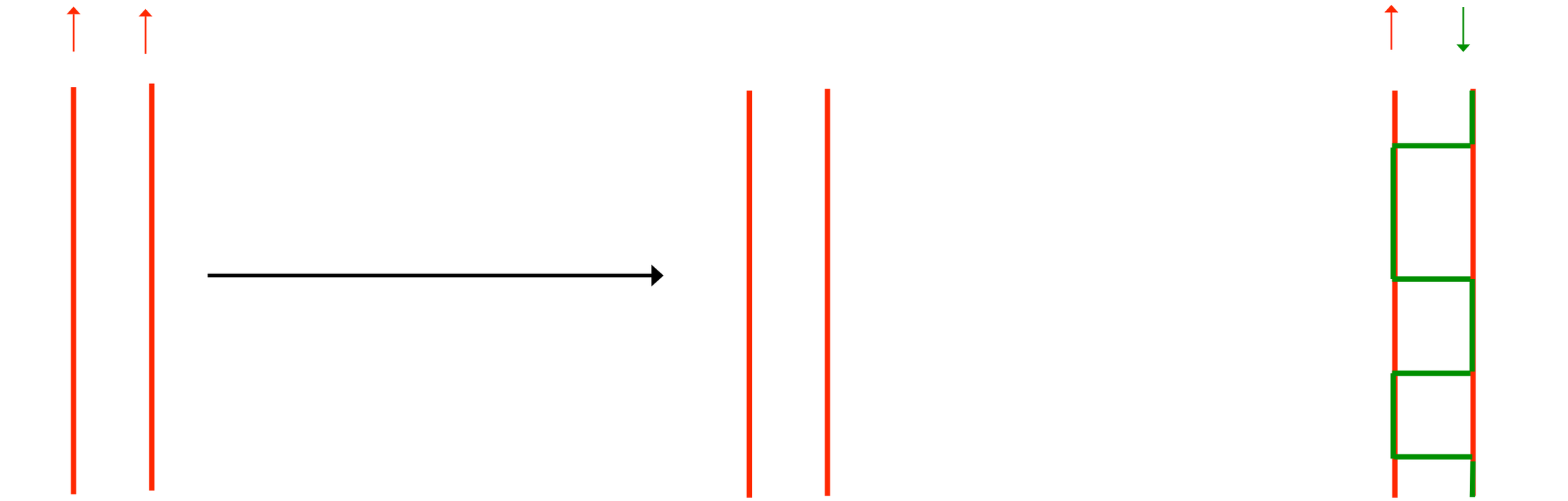
- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

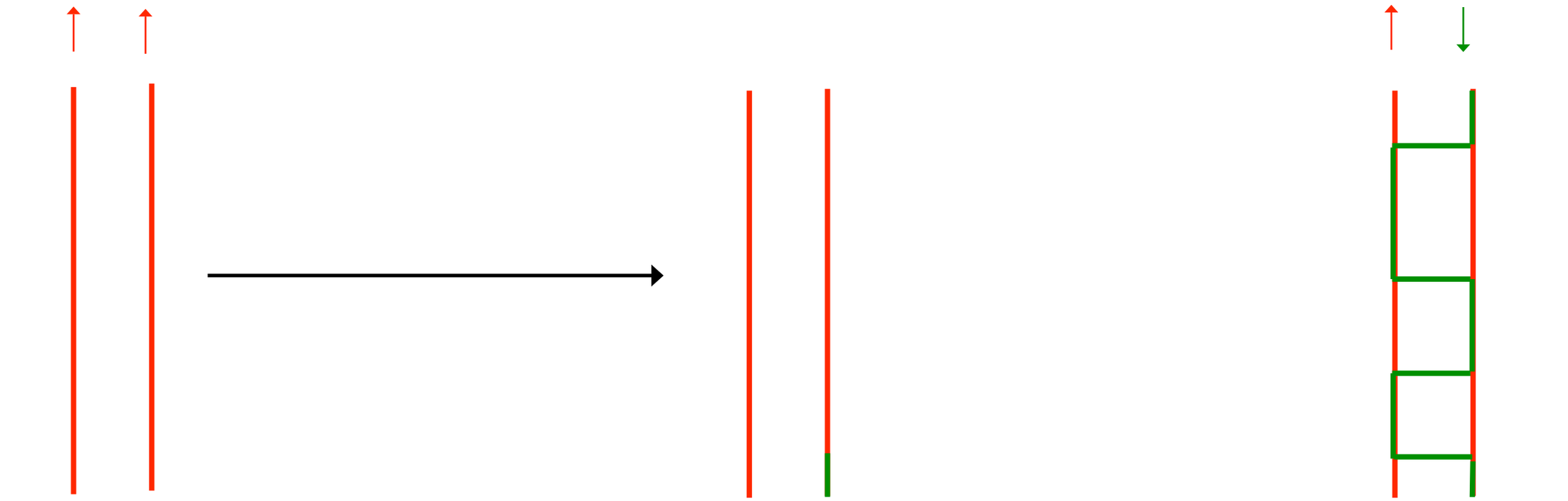


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

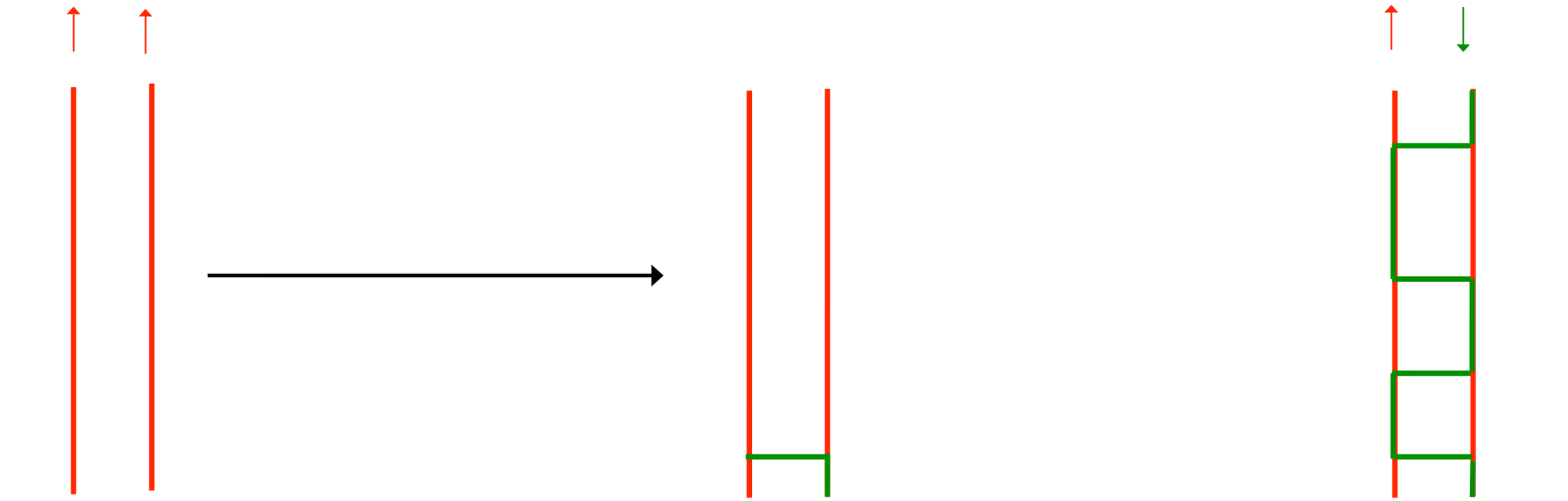


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

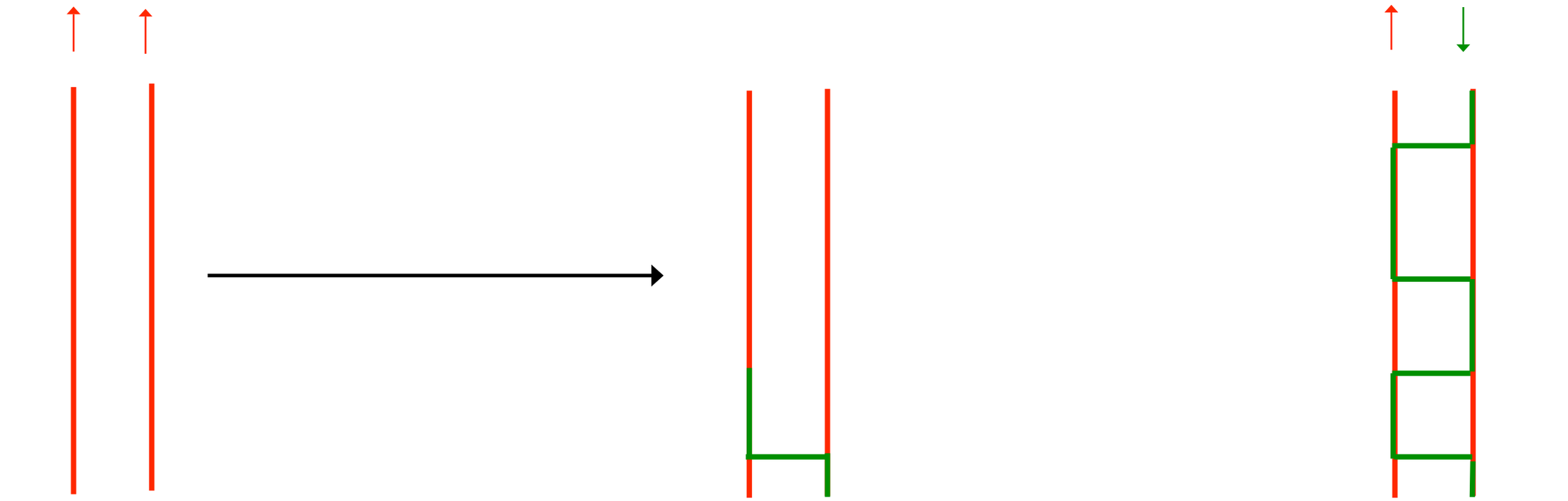


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

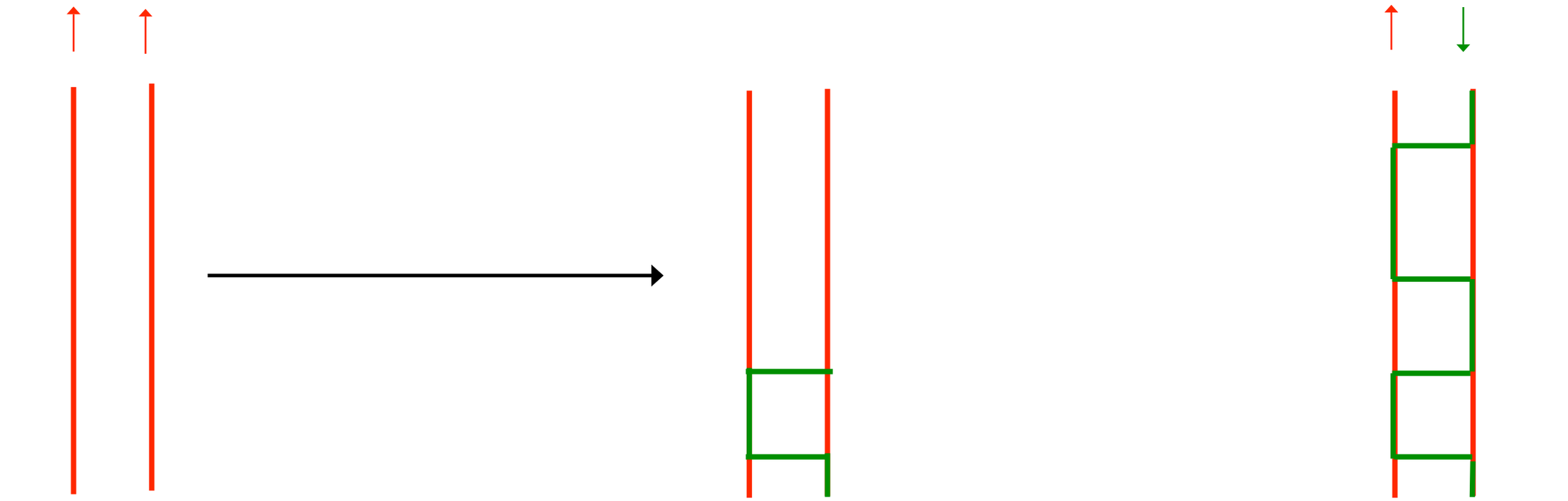


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

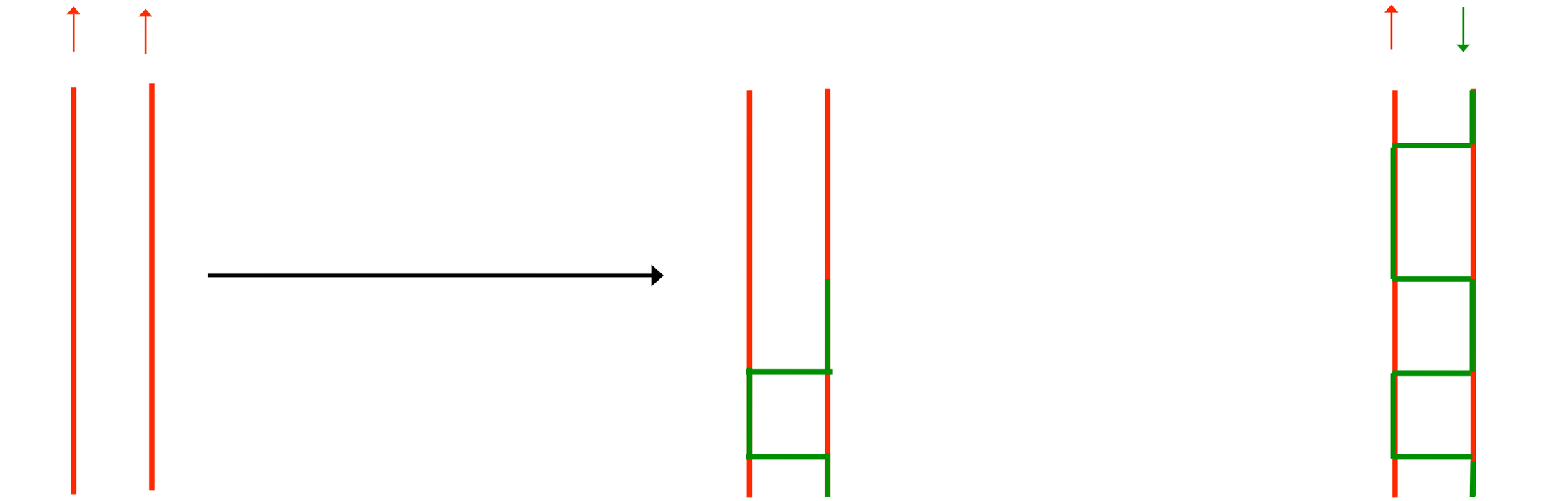


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

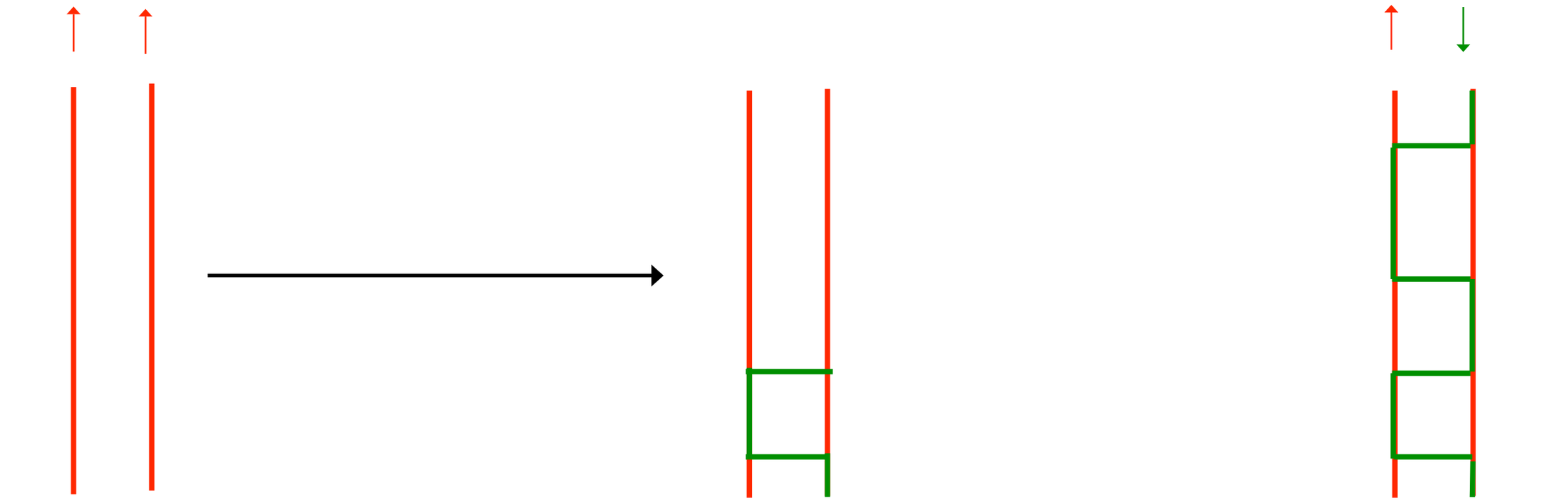


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

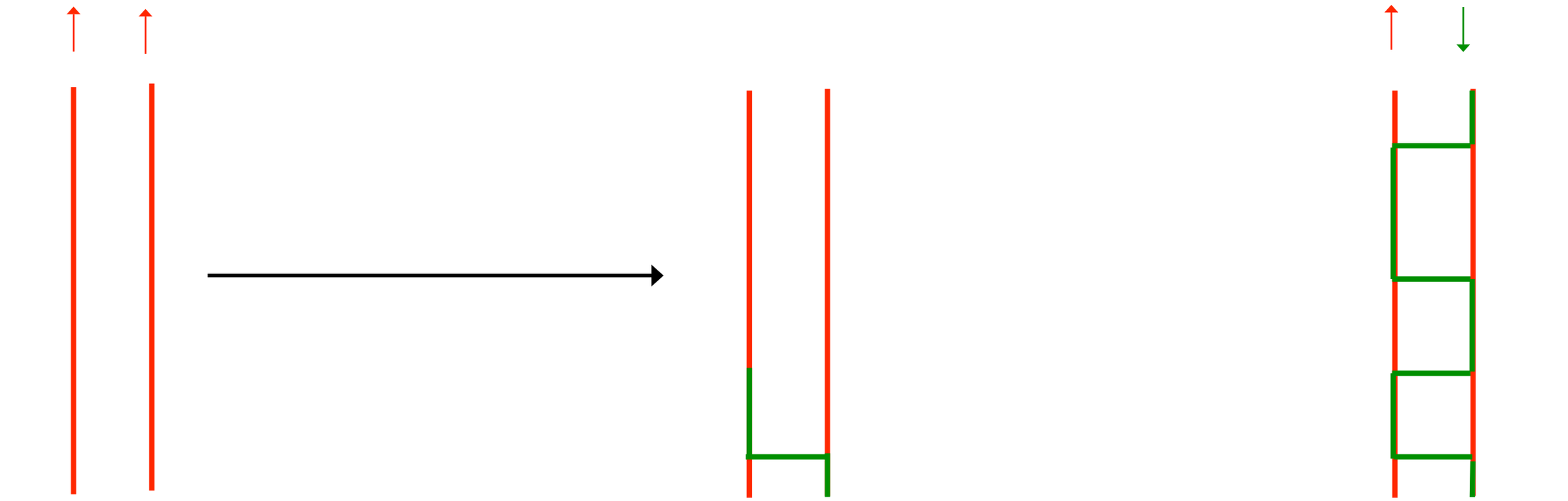


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

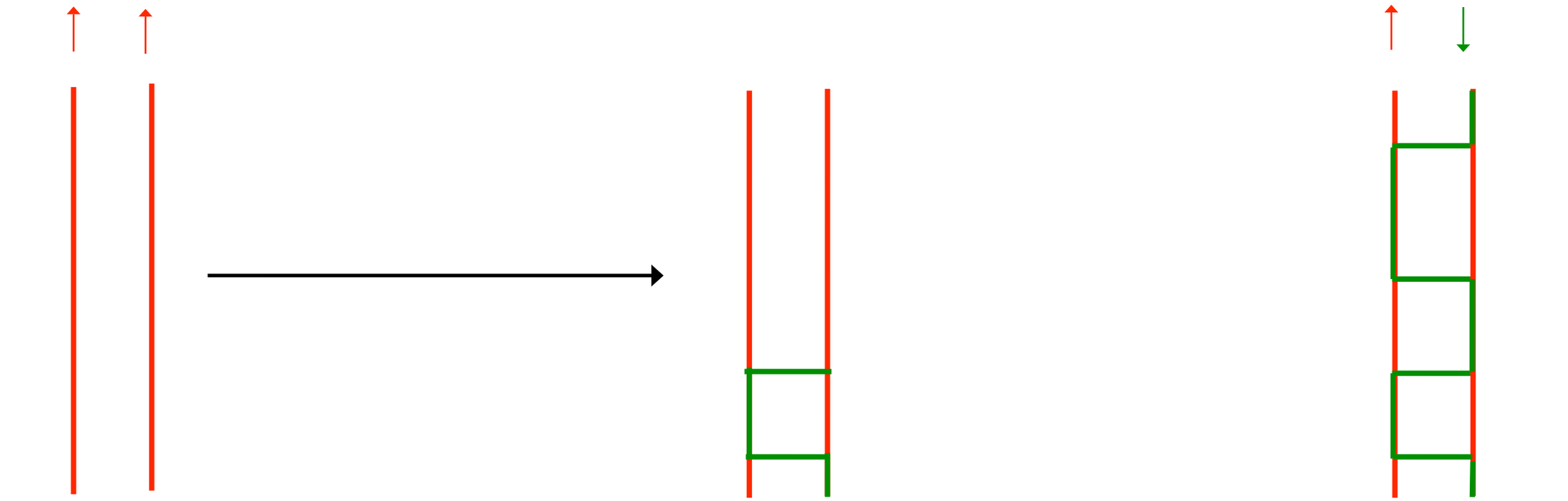


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

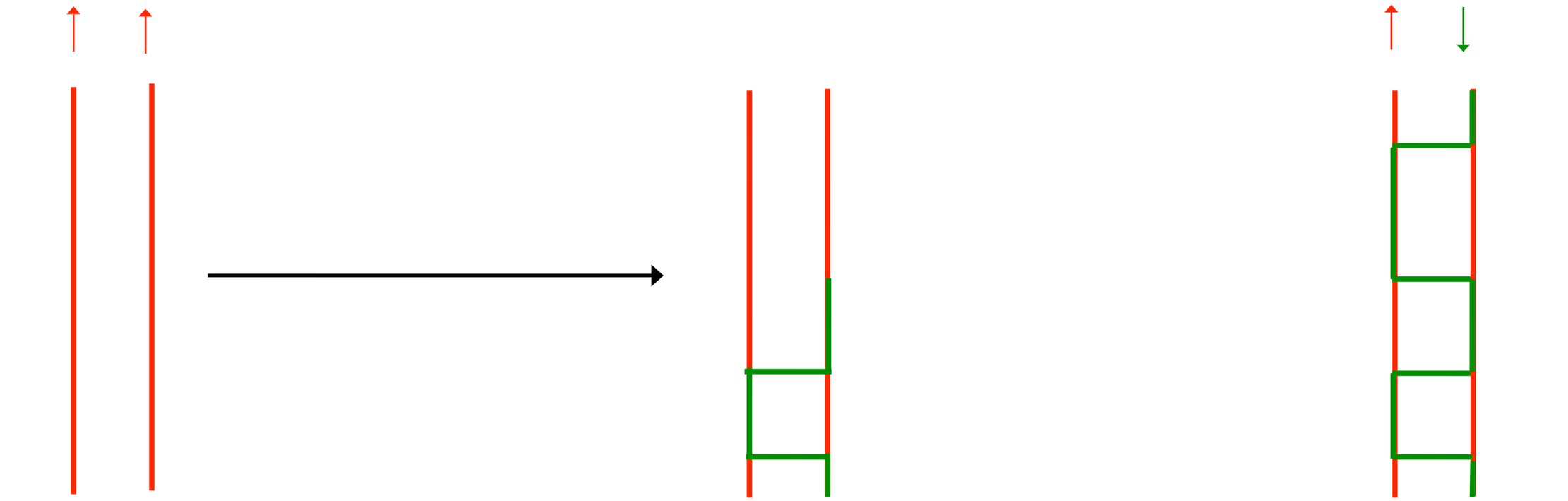


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

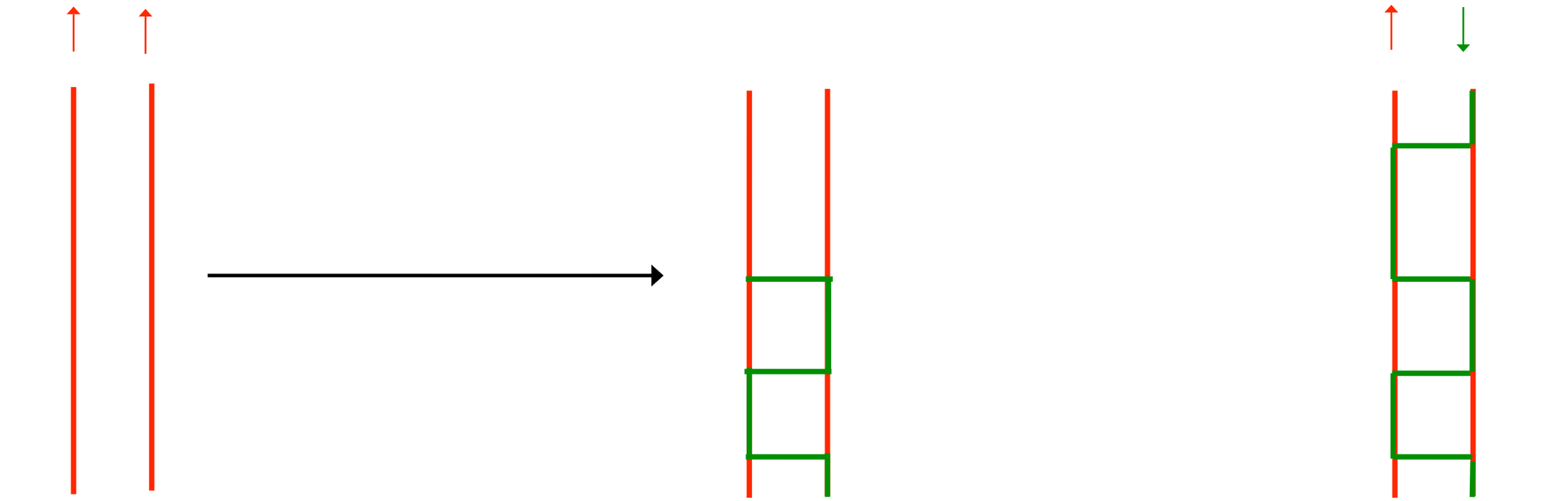


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

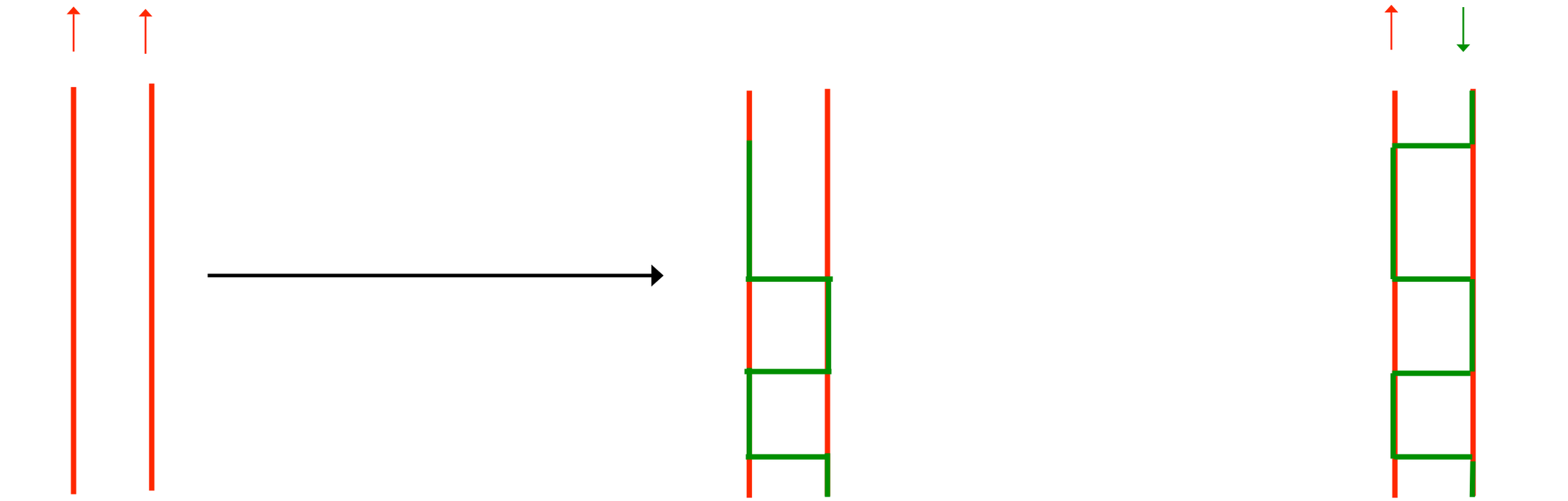


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

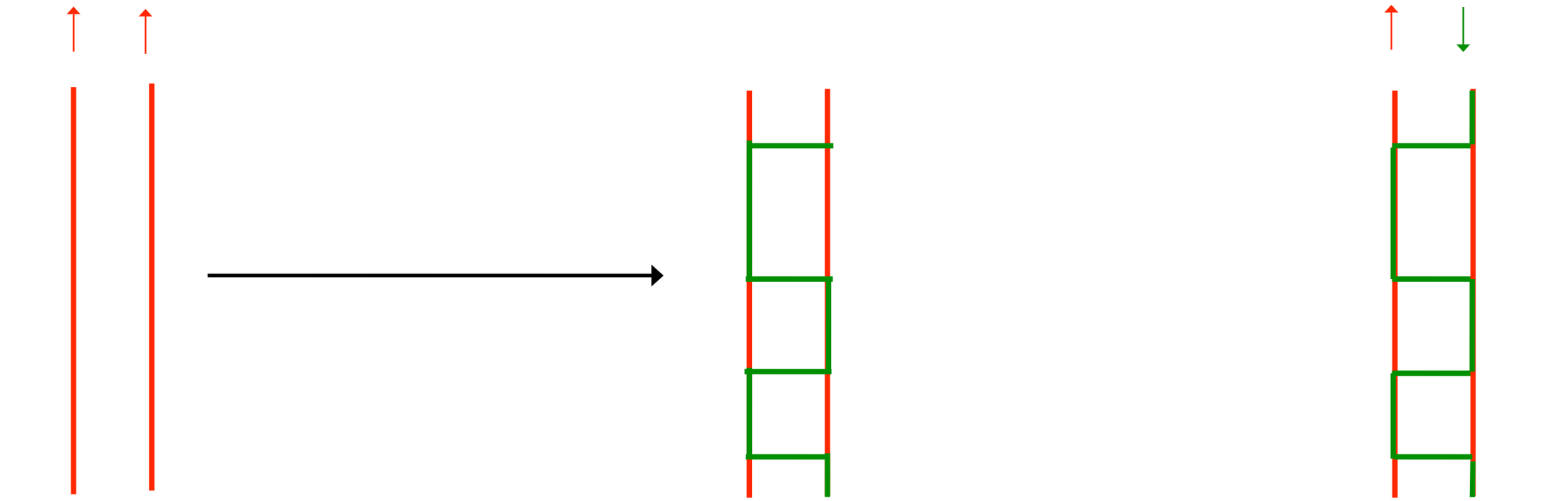


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

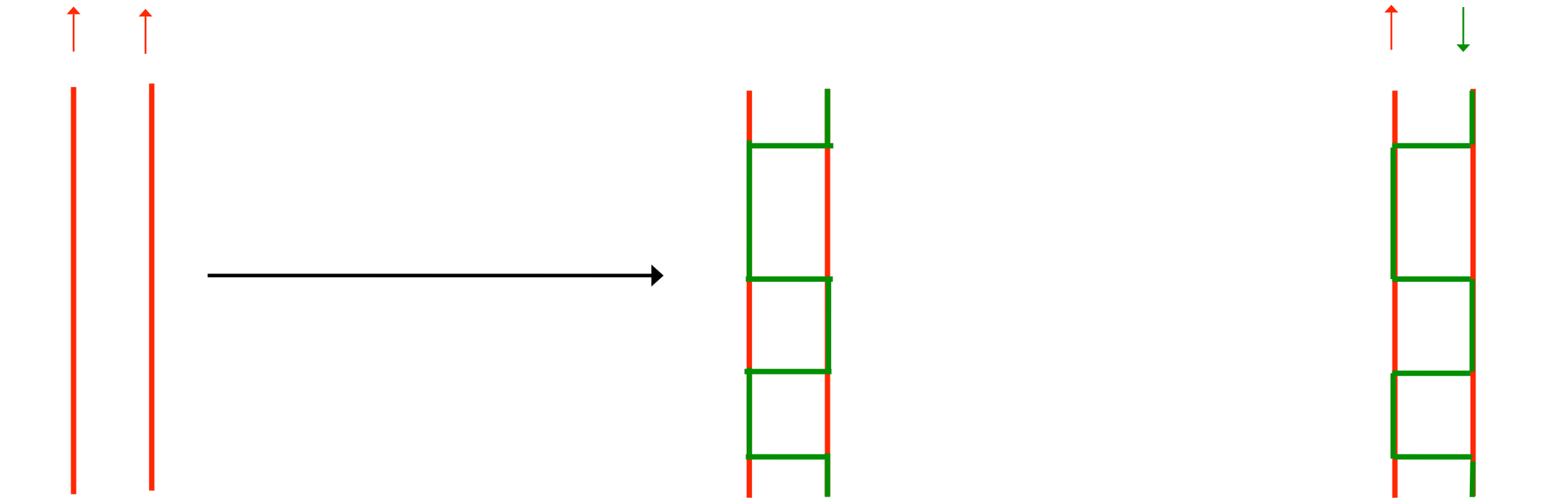


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

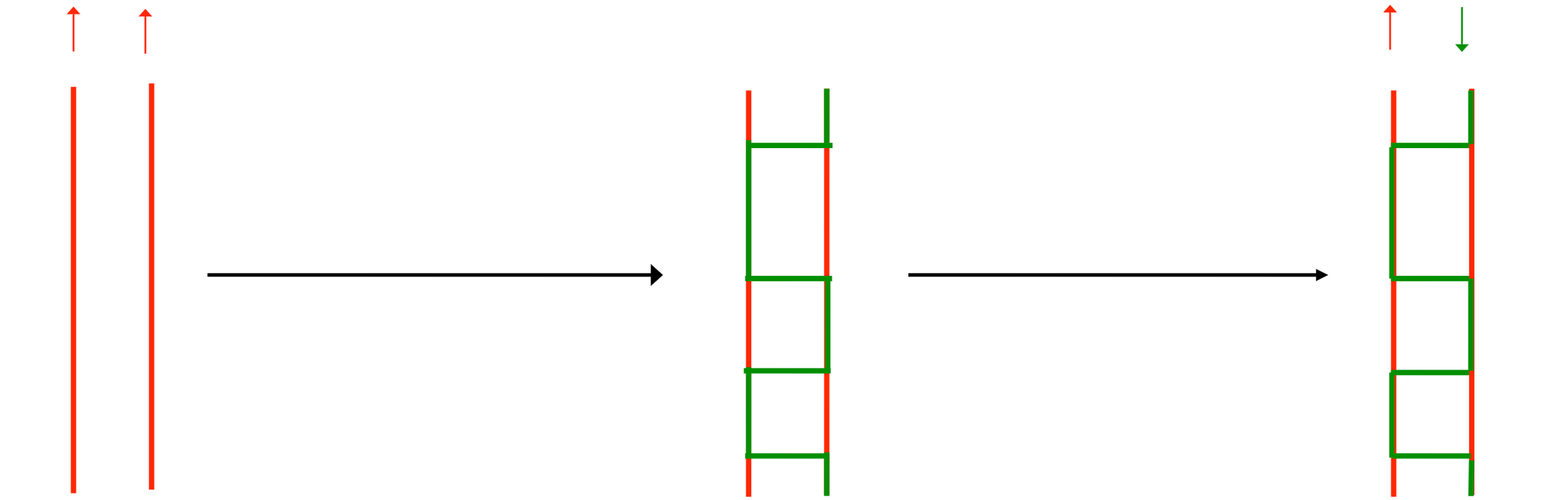


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1

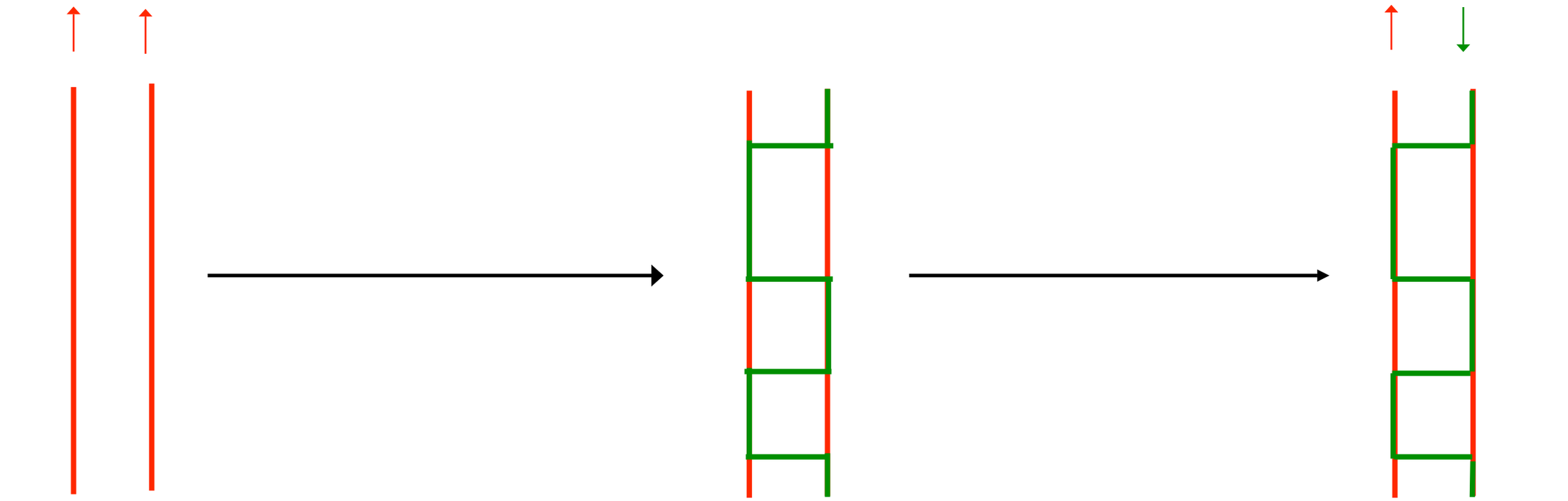


Triplet
$E = J/4 - h = -3/4$

Singlet
$E = -3J/4 = -3/4$

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1



Triplet
$E = J/4 - h = -3/4$

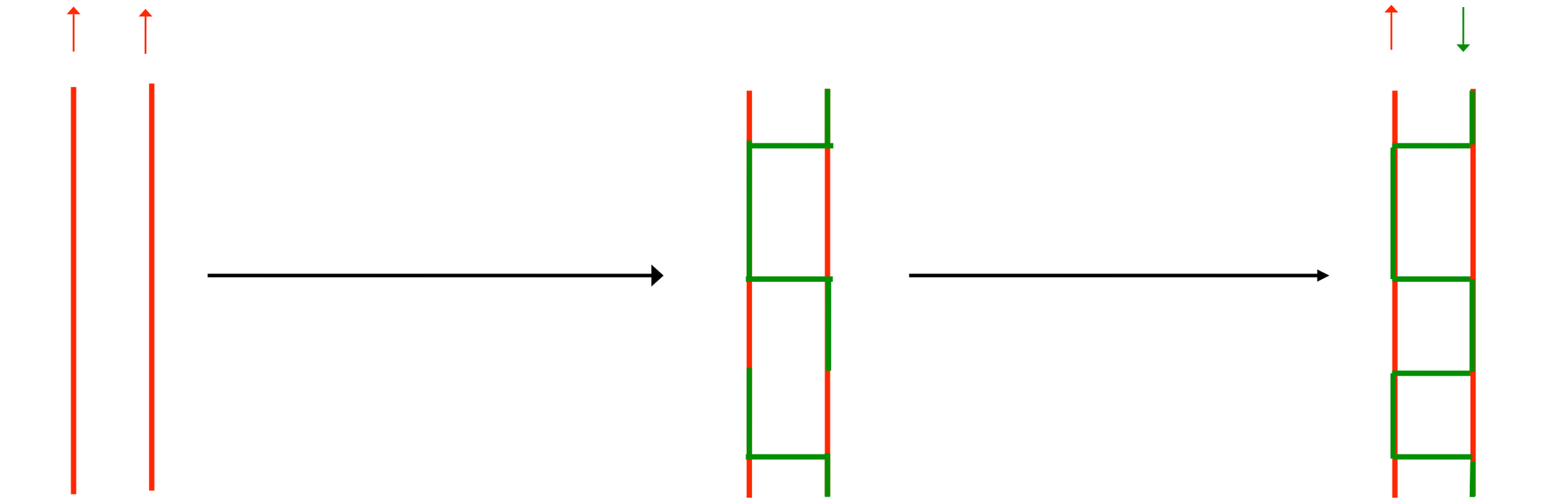
Singlet
$E = -3J/4 = -3/4$

No high energy intermediate state

Efficient update in presence of a magnetic field

Worm algorithm in a magnetic field

- Worm algorithm performs a random walk
  - Change of configuration done in small steps
- Example: spin dimer at J = h =1



No high energy intermediate state

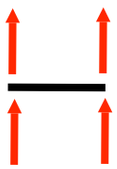
Efficient update in presence of a magnetic field

Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method

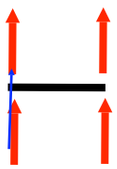
Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method



Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method

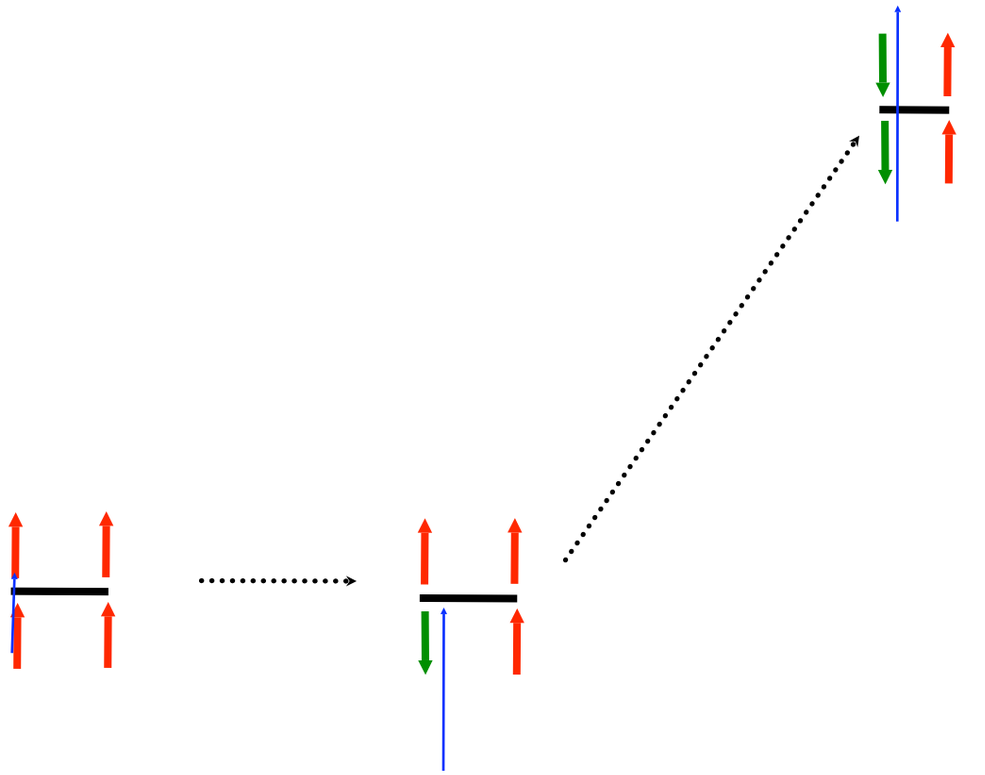


Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method

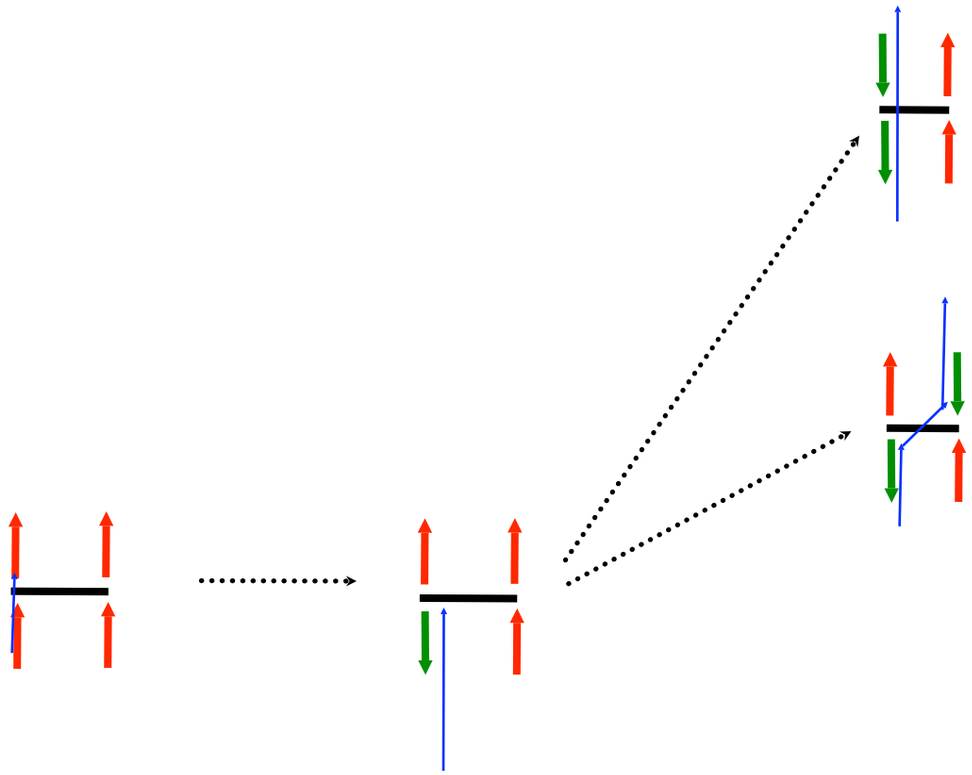
Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method



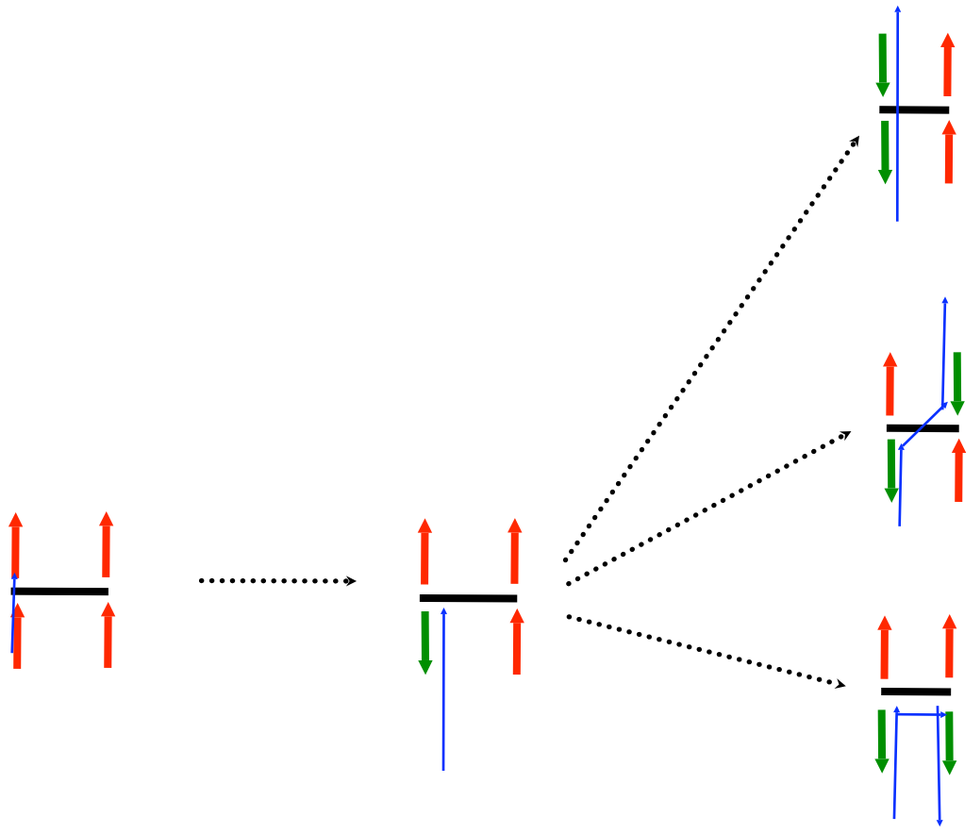
Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method



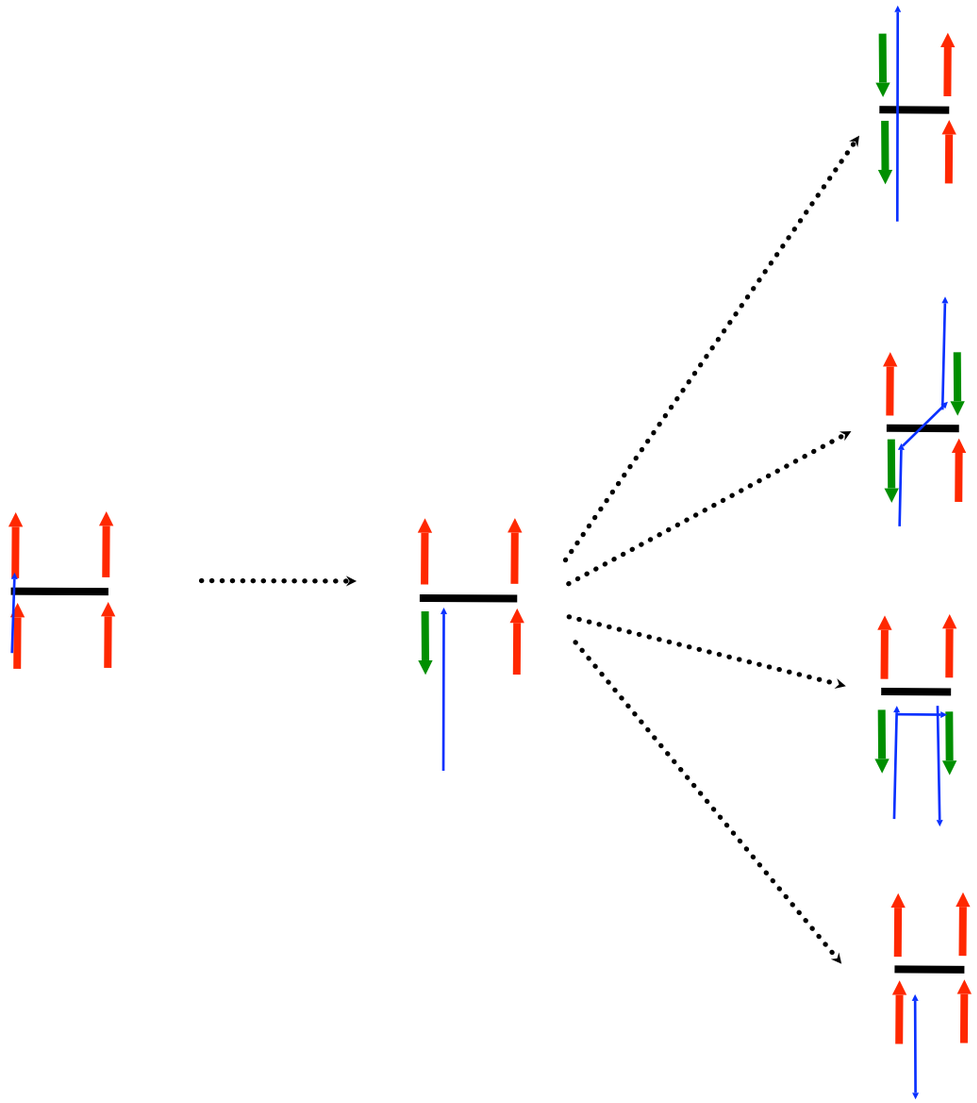
Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method



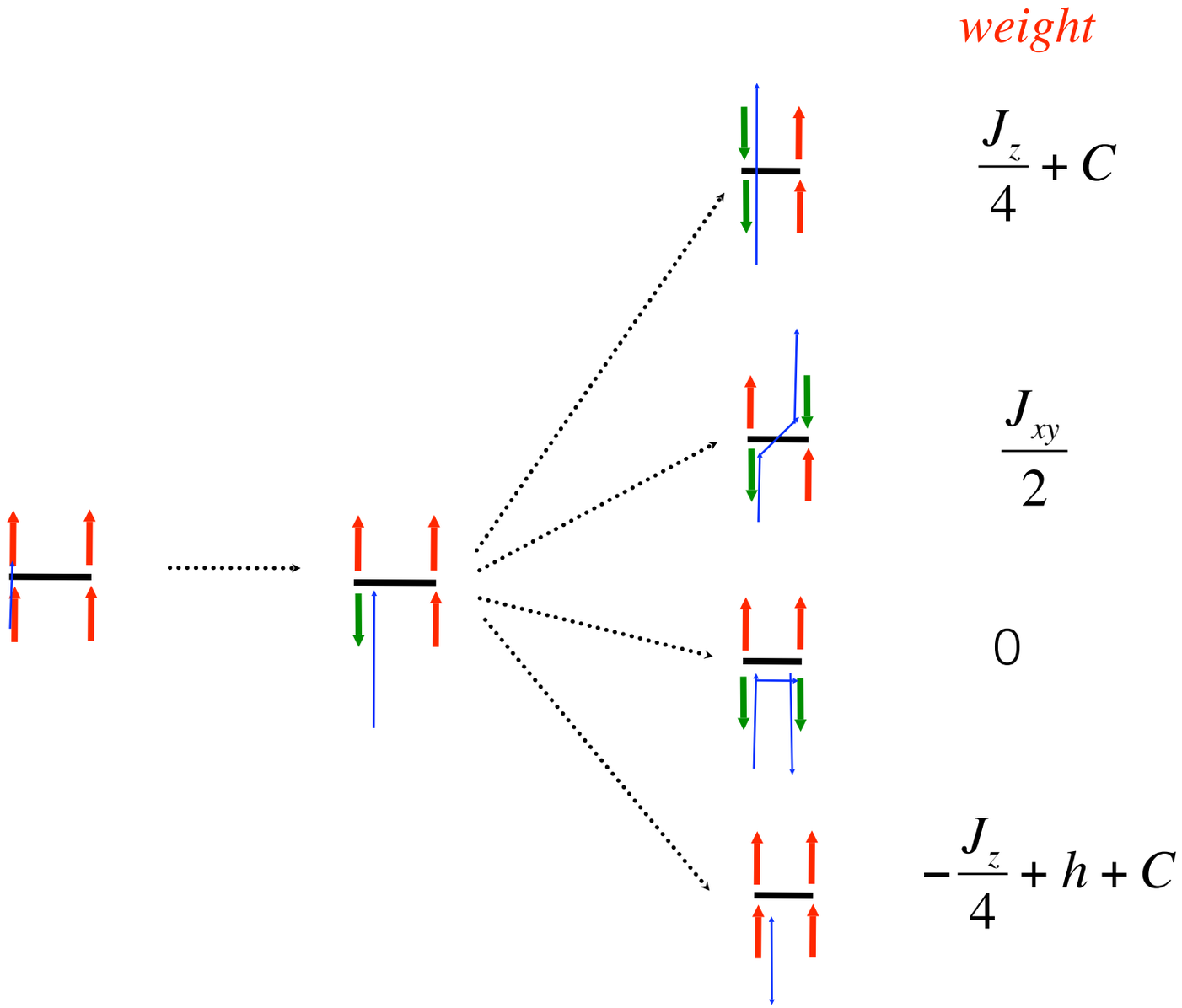
Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method



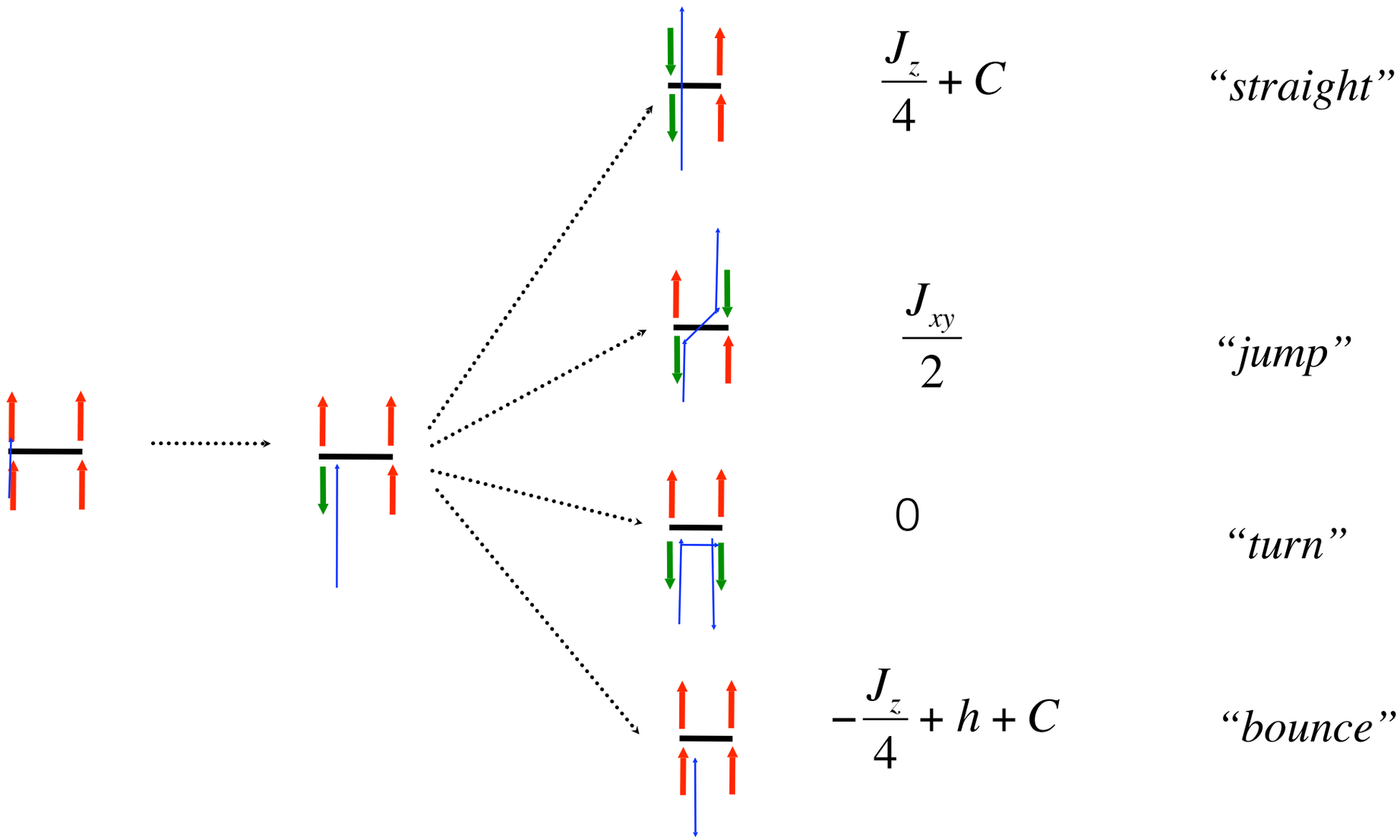
Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method



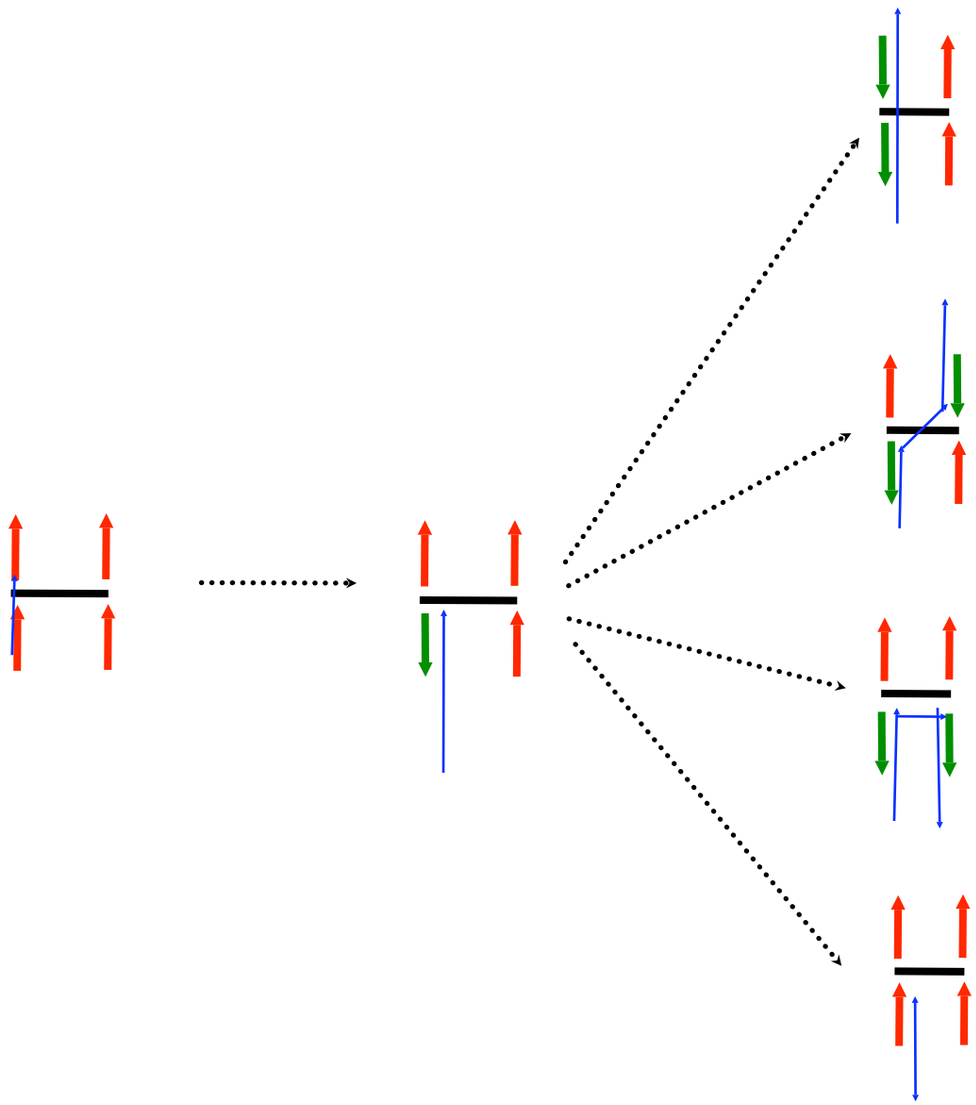
Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method



Directed loops in SSE

- Instead of following a pre-chosen path given by graphs we pick randomly
- E.g. using heat bath method



	weight		*probability*
	$\frac{J_z}{4} + C$	*"straight"*	$\frac{J_z/4 + C}{2C + J_{xy}/2 + h}$
	$\frac{J_{xy}}{2}$	*"jump"*	$\frac{J_{xy}/2}{2C + J_{xy}/2 + h}$
	0	*"turn"*	0
	$-\frac{J_z}{4} + h + C$	*"bounce"*	$\frac{-J_z/4 + h + C}{2C + J_{xy}/2 + h}$

Better directed loop schemes

Better directed loop schemes

- Bounces are bad since they undo the last change
- If bounce can be eliminated ⇒ loop algorithm possible
  - Directed loops give loop algorithm as a limit for some models
- Bounce path can be minimized
  - In models where there is no loop algorithm
  - Better choices for path selection
    - O.F. Syljuåsen and A.W. Sandvik, PRE (2002)
    - O.F. Syljuåsen , PRE (2003)
    - F. Alet, S. Wessel and M. Troyer, PRE (2004)

When to use which algorithm?

- Stochastic Series Expansion (SSE) is simpler to implement
- Continuous-time path integrals needs lower orders
- Use SSE for local actions with not too large diagonal terms

	SSE	Path Integrals
Loop algorithm	Spin models	Spin models with dissipation
Worm algorithm	Spin models in magnetic field	Bose-Hubbard models

Summary: Non local QMC algorithms

- The loop algorithm:
  - ✶ A generalization of Swendsen-Wang algorithm to quantum case
  - ✶ Solves problem of critical slowing down
  - ✶ Implementation choices : Discrete / Continuous time, Single / Multi loops, SSE or path integrals
  - ✶ Exponential slowing down for non particle-hole symmetric systems (magnetic field)

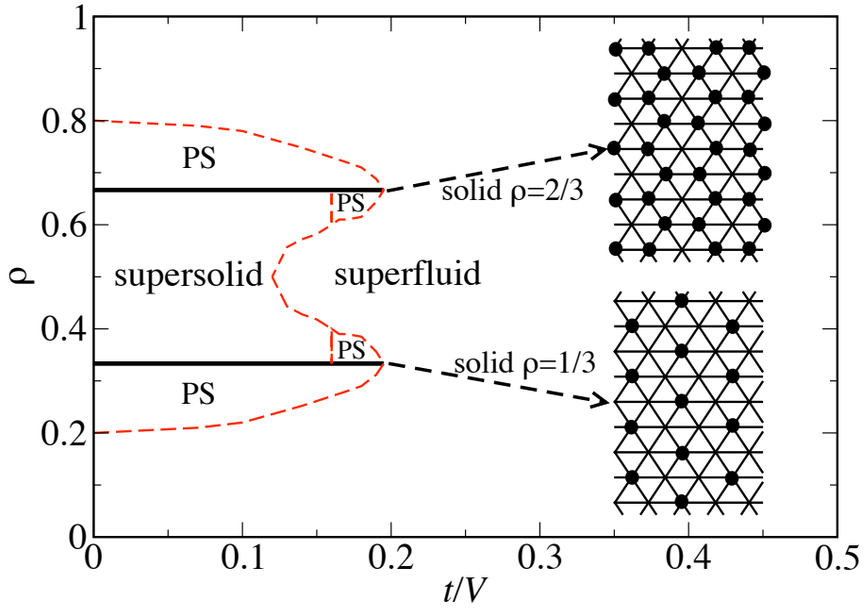
- Worm and directed loop algorithms:
  - ✶ Relax loop-building rules : A partial loop (“worm”) performs a random walk on the lattice
  - ✶ Detailed balance fulfilled at each step : Many solutions and room for optimization (directed loops)
  - ✶ Once “head” and “tail” meet, a loop is finished
  - ✶ Very efficient even for non-particle hole symmetric systems
  - ✶ Relation to loop algorithm : loop algorithm smoothly recovered as field vanishes
  - ✶ Might not work well when the relevant Green’s function is short ranged (solids, paired states).

Applications of Quantum Monte Carlo
Supersolids on the triangular lattice

- Antiferromagnetic Ising Model on the triangular lattice augmented by a ferromagnetic transverse exchange
  SSE QMC is possible without a sign problem !

- AF Ising model has a macroscopic degeneracy at T=0 (Wannier, Houtappel).

- In a whole region of magnetizations $-1/3 < m/m_{\mathrm{sat}} < 1/3$
  the system becomes supersolid upon switching on the ferromagnetic transverse exchange



In a bosonic language a supersolid is a state which shows charge order and superfluidity at the same time

G. Murthy et al, Phys. Rev. B **55**, 3104 (1997).
S. Wessel & M. Troyer, Phys. Rev. Lett. **95** 127205 (2005).
D. Heidarian & K. Damle, Rev. Lett. **95** 127206 (2005).
R. Melko et al, Phys. Rev. Lett. **95** 127207 (2005).

5. Green's function Monte Carlo

Green's function Monte Carlo

- Rather different approach compared to the "finite temperature" QMC methods we have seen.

- Stochastic application of the Hamiltonian matrix on a wave function in order to filter out the ground state.

- aka Diffusion Monte Carlo, Projector Monte Carlo

- Currently the only QMC method able to treat Quantum Dimer models on reasonably large lattices, even in the absence of a sign problem.

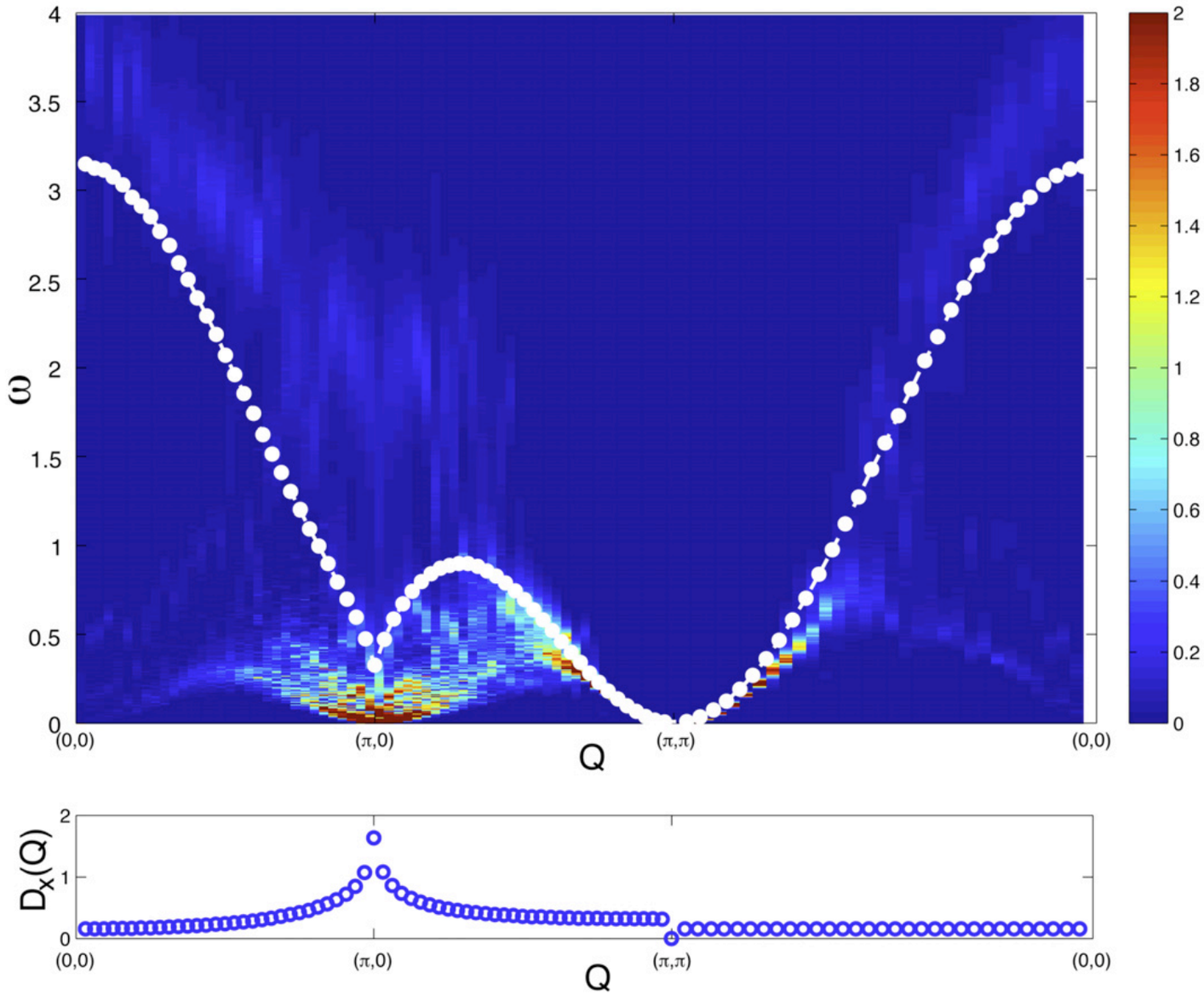
- Relies on the quality of a guiding function. When guiding function is exact, the Quantum Monte Carlo process becomes Classical. When formulated in continuous time it becomes possible to study the imaginary time evolution of a quantum system.

Green's function MC at the RK point

- At the RK point the guiding function is exact.
- Let us study then the dimer excitation spectrum at the RK point

Green's function MC at the RK point

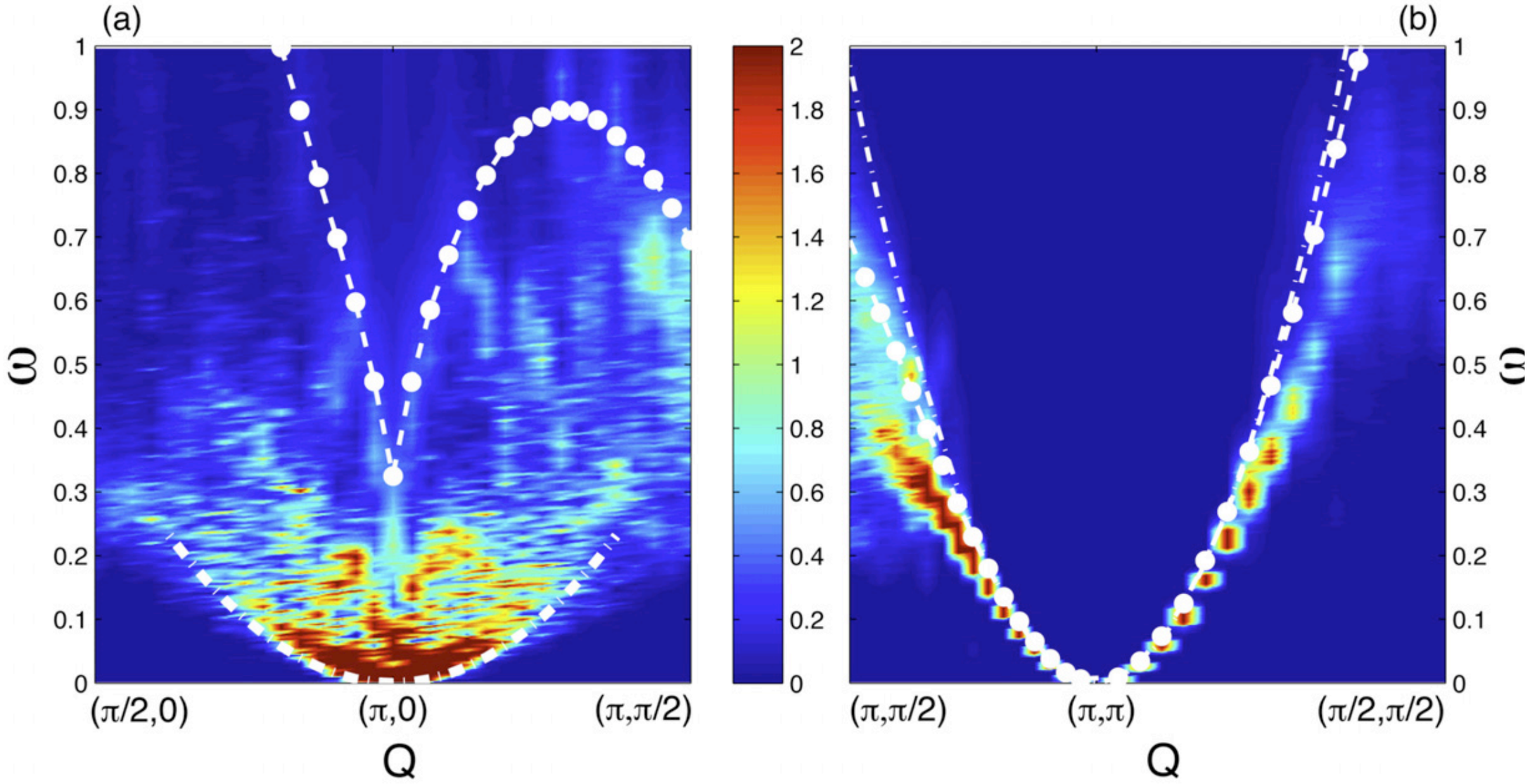
- At the RK point the guiding function is exact.
- Let us study then the dimer excitation spectrum at the RK point



AML, S. Capponi, F. Assaad, JSTAT 08

Green's function MC at the RK point

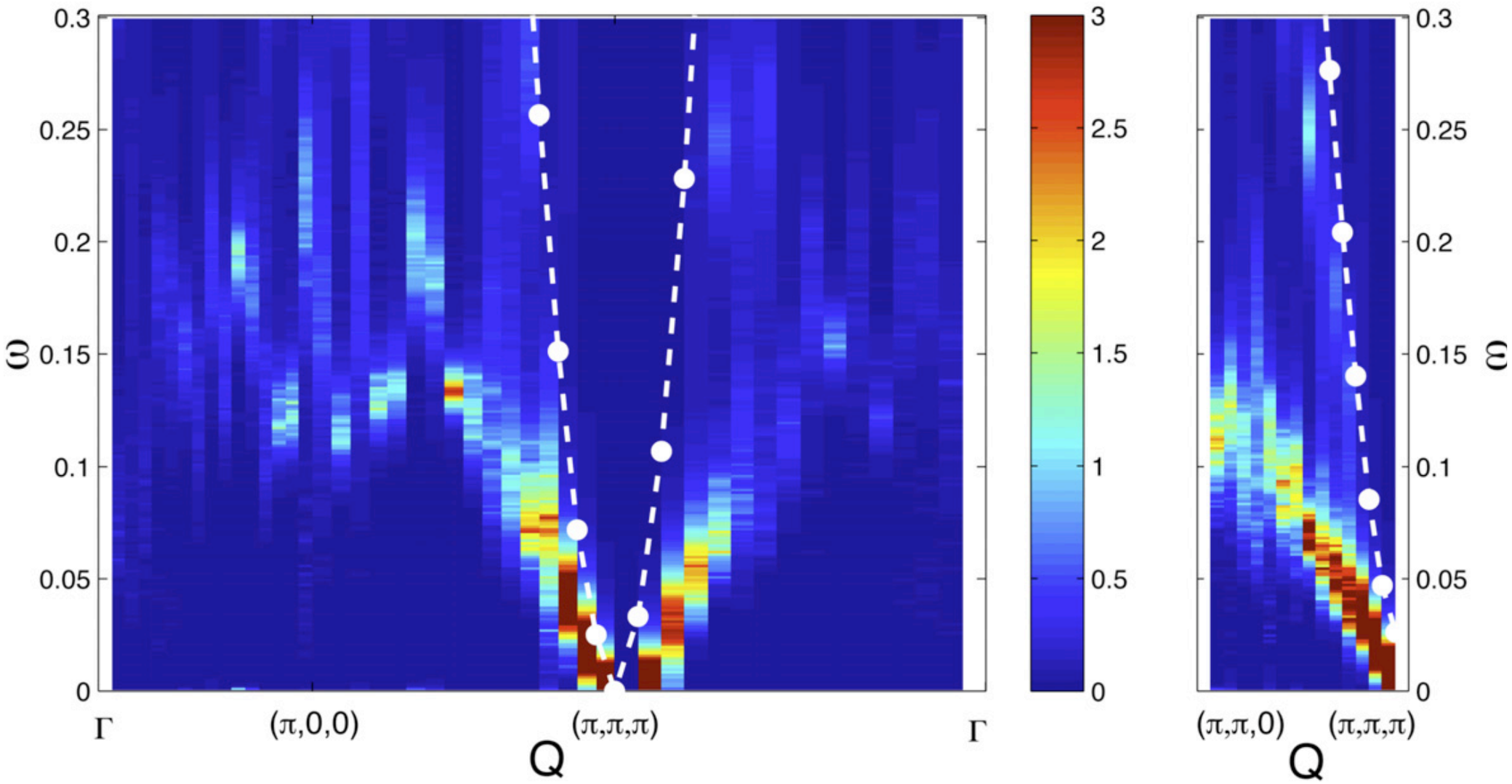
- At the RK point the guiding function is exact.
- Let us study then the dimer excitation spectrum at the RK point



AML, S. Capponi, F. Assaad, JSTAT 08

Green's function MC at the RK point

- At the RK point the guiding function is exact.
- Let us study then the dimer excitation spectrum at the RK point



AML, S. Capponi, F. Assaad, JSTAT 08

6. The negative sign problem in QMC

Quantum Monte Carlo

- Not as easy as classical Monte Carlo

$$Z = \sum_c e^{-E_c / k_B T}$$

- Calculating the eigenvalues E_c is equivalent to solving the problem

- Need to find a mapping of the quantum partition function to a classical problem

$$Z = \mathrm{Tr}\, e^{-\beta H} \equiv \sum_c p_c$$

- “Negative sign” problem if some $p_c < 0$

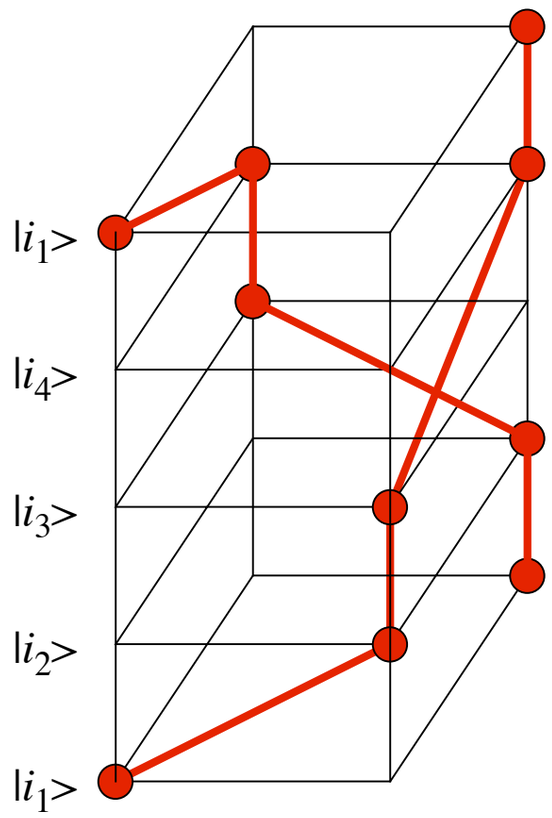
The negative sign problem

- In mapping of quantum to classical system

$$Z = \mathrm{Tr} e^{-\beta H} = \sum_i p_i$$

- there is a “sign problem” if some of the $p_i < 0$

  - Appears e.g. in simulation of electrons when two electrons exchange places (Pauli principle)



The negative sign problem

- Sample with respect to absolute values of the weights

$$\langle A \rangle = \sum_i A_i p_i \Big/ \sum_i p_i = \frac{\sum_i A_i \operatorname{sgn} p_i |p_i| \Big/ \sum_i |p_i|}{\sum_i \operatorname{sgn} p_i |p_i| \Big/ \sum_i |p_i|} \equiv \frac{\langle A \cdot sign \rangle_{|p|}}{\langle sign \rangle_{|p|}}$$

- Exponentially growing cancellation in the sign

$$\langle sign \rangle = \frac{\sum_i p_i}{\sum_i |p_i|} = Z/Z_{|p|} = e^{-\beta V (f - f_{|p|})}$$

- Exponential growth of errors

$$\frac{\Delta sign}{\langle sign \rangle} = \frac{\sqrt{\langle sign^2 \rangle - \langle sign \rangle^2}}{\sqrt{M} \langle sign \rangle} \approx \frac{e^{\beta V (f - f_{|p|})}}{\sqrt{M}}$$

- NP-hard problem (no general solution) [Troyer and Wiese, PRL 2005]

Is the sign problem exponentially hard?

- The sign problem is basis-dependent
  - Diagonalize the Hamiltonian matrix $H|i\rangle = \varepsilon_i |i\rangle$

$$\langle A \rangle = \mathrm{Tr}[A \exp(-\beta H)] / \mathrm{Tr}[\exp(-\beta H)] = \sum_i \langle i|A_i|i\rangle \exp(-\beta\varepsilon_i) \Big/ \sum_i \exp(-\beta\varepsilon_i)$$

  - All weights are positive
  - But this is an *exponentially hard problem* since $\dim(H)=2^N$!
  - Good news: the sign problem is basis-dependent!
- But: the sign problem is still not solved
  - Despite decades of attempts
- Reminiscent of the NP-hard problems
  - No proof that they are exponentially hard
  - No polynomial solution either

Solving an NP-hard problem by QMC

- Take 3D Ising spin glass $$H = \sum_{\langle i,j \rangle} J_{ij} \sigma_j \sigma_j \text{ with } J_{ij} = 0, \pm 1$$

- View it as a quantum problem in basis where H it is not diagonal

$$H^{(SG)} = \sum_{\langle i,j \rangle} J_{ij} \sigma^x{}_j \sigma^x{}_j \text{ with } J_{ij} = 0, \pm 1$$

  - The randomness ends up in the sign of offdiagonal matrix elements
  - Ignoring the sign gives the ferromagnet and loop algorithm is in P

$$H^{(FM)} = -\sum_{\langle i,j \rangle} \sigma^x{}_j \sigma^x{}_j$$

  - The sign problem causes NP-hardness

- solving the sign problem solves all the NP-complete problems and prove NP=P

Summary: negative sign problem

Summary: negative sign problem

- A “solution to the sign problem” solves all problems in NP

Summary: negative sign problem

- A “solution to the sign problem” solves all problems in NP
- Hence a general solution to the sign problem does not exist unless P=NP
  - If you still find one and thus prove that NP=P you will get
    - 1 million US $ from the Clay foundation !
    - A Nobel prize?
    - A Fields medal?

Summary: negative sign problem

- A “solution to the sign problem” solves all problems in NP
- Hence a general solution to the sign problem does not exist unless P=NP
  - If you still find one and thus prove that NP=P you will get
    - 1 million US $ from the Clay foundation !
    - A Nobel prize?
    - A Fields medal?
- What does this imply?
  - A general method cannot exist
  - Look for specific solutions to the sign problem or model-specific methods e.g. Meron algorithm

The origin of the sign problem

The origin of the sign problem

- We sample with the wrong distribution by ignoring the sign!

The origin of the sign problem

- We sample with the wrong distribution by ignoring the sign!
- We simulate bosons and expect to learn about fermions?
  - will only work in insulators and superfluids

The origin of the sign problem

- We sample with the wrong distribution by ignoring the sign!
- We simulate bosons and expect to learn about fermions?
  - will only work in insulators and superfluids
- We simulate a ferromagnet and expect to learn something useful about a frustrated antiferromagnet?

The origin of the sign problem

- We sample with the wrong distribution by ignoring the sign!
- We simulate bosons and expect to learn about fermions?
  - will only work in insulators and superfluids
- We simulate a ferromagnet and expect to learn something useful about a frustrated antiferromagnet?
- We simulate a ferromagnet and expect to learn something about a spin glass?
  - This is the idea behind the proof of NP-hardness

Working around the sign problem

Working around the sign problem

1. Simulate “bosonic” systems

   - Bosonic atoms in optical lattices, Fermions with n.n. hopping in 1D
   - Helium-4 supersolids
   - Nonfrustrated magnets

Working around the sign problem

1. Simulate “bosonic” systems
   - Bosonic atoms in optical lattices, Fermions with n.n. hopping in 1D
   - Helium-4 supersolids
   - Nonfrustrated magnets
2. Simulate sign-problem free fermionic systems
   - Attractive on-site interactions
   - Half-filled Mott insulators

Working around the sign problem

1. Simulate “bosonic” systems
   - Bosonic atoms in optical lattices, Fermions with n.n. hopping in 1D
   - Helium-4 supersolids
   - Nonfrustrated magnets
2. Simulate sign-problem free fermionic systems
   - Attractive on-site interactions
   - Half-filled Mott insulators
3. Restriction to quasi-1D systems
   - Use the density matrix renormalization group method (DMRG)

Working around the sign problem

1. Simulate “bosonic” systems
   - Bosonic atoms in optical lattices, Fermions with n.n. hopping in 1D
   - Helium-4 supersolids
   - Nonfrustrated magnets
2. Simulate sign-problem free fermionic systems
   - Attractive on-site interactions
   - Half-filled Mott insulators
3. Restriction to quasi-1D systems
   - Use the density matrix renormalization group method (DMRG)
4. Use approximate methods
   - Dynamical mean field theory (DMFT)

Summary of today's lecture

- Quantum Monte Carlo (World line)
- Loop Algorithm
- Stochastic Series Expansion
- Worm Algorithm
- Green's function Monte Carlo
- Sign Problem

Thank you !