Assorted Topics in the Monte Carlo Simulation of Polymers

Nathan Clisby
MASCOS / Department of Mathematics and Statistics
The University of Melbourne

SIAM Conference on Discrete Mathematics
Minneapolis
June 19, 2014

Outline

- Self-avoiding walks

Outline

- Self-avoiding walks
- Pivot algorithm

Outline

- Self-avoiding walks
- Pivot algorithm
- SAW-tree, a data structure for fast pivot moves

Outline

- Self-avoiding walks
- Pivot algorithm
- SAW-tree, a data structure for fast pivot moves
- How can we extend this idea?

Outline

- Self-avoiding walks
- Pivot algorithm
- SAW-tree, a data structure for fast pivot moves
- How can we extend this idea?
  - Dense polymers - new symmetries

Outline

- Self-avoiding walks
- Pivot algorithm
- SAW-tree, a data structure for fast pivot moves
- How can we extend this idea?
  - Dense polymers - new symmetries
  - R_{h} - estimate observables

Outline

- Self-avoiding walks
- Pivot algorithm
- SAW-tree, a data structure for fast pivot moves
- How can we extend this idea?
  - Dense polymers - new symmetries
  - R_{h} - estimate observables
  - μ - neither local nor global

Outline

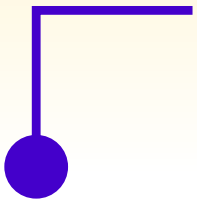
- Self-avoiding walks
- Pivot algorithm
- SAW-tree, a data structure for fast pivot moves
- How can we extend this idea?
  - Dense polymers - new symmetries
  - R_{h} - estimate observables
  - μ - neither local nor global
- Conclusion

Self-avoiding walk model

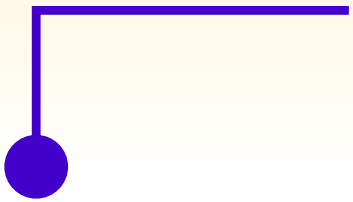
- A walk on a lattice, step to neighbouring site provided it has not already been visited.

Simple random walk

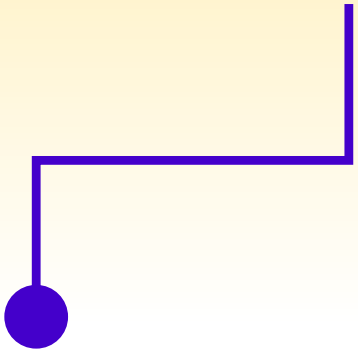
Simple random walk



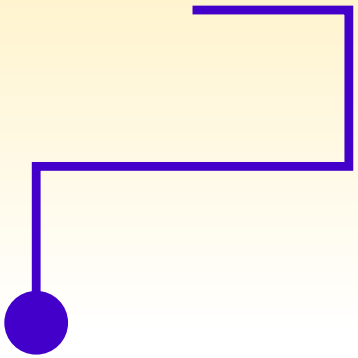
Simple random walk



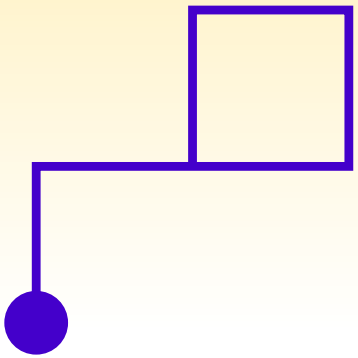
Simple random walk



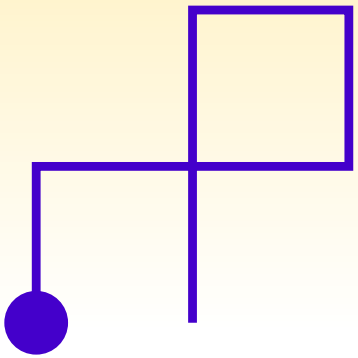
Simple random walk



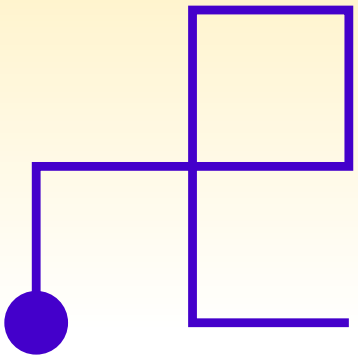
Simple random walk



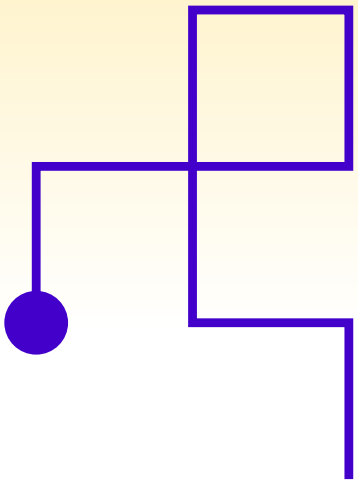
Simple random walk



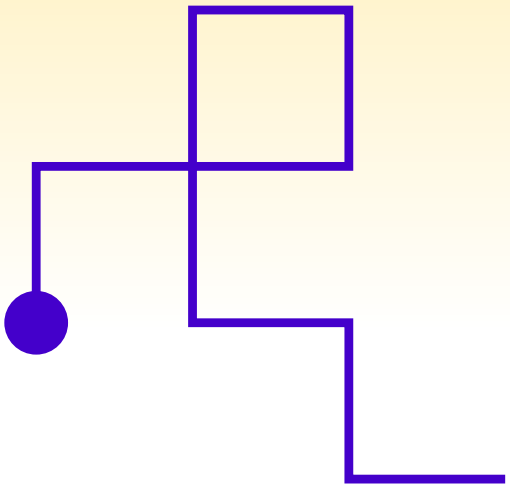
Simple random walk



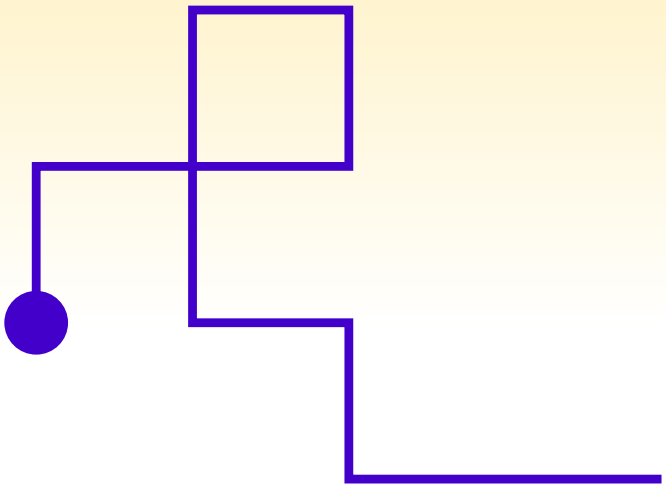
Simple random walk



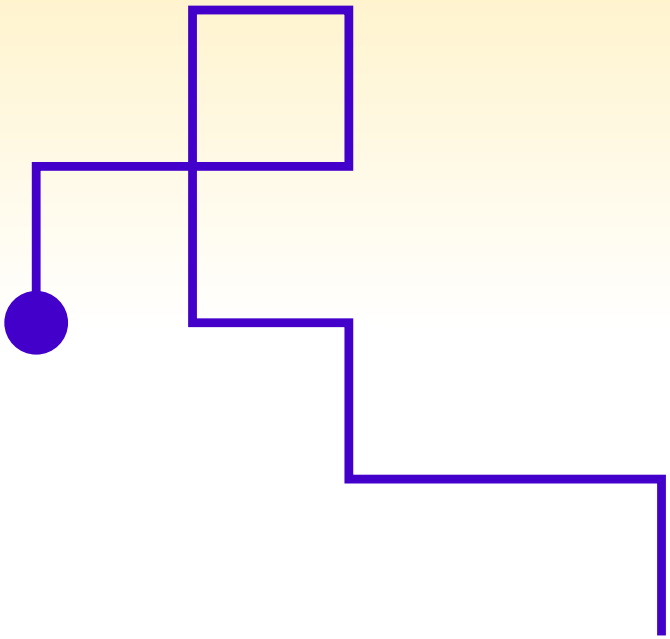
Simple random walk



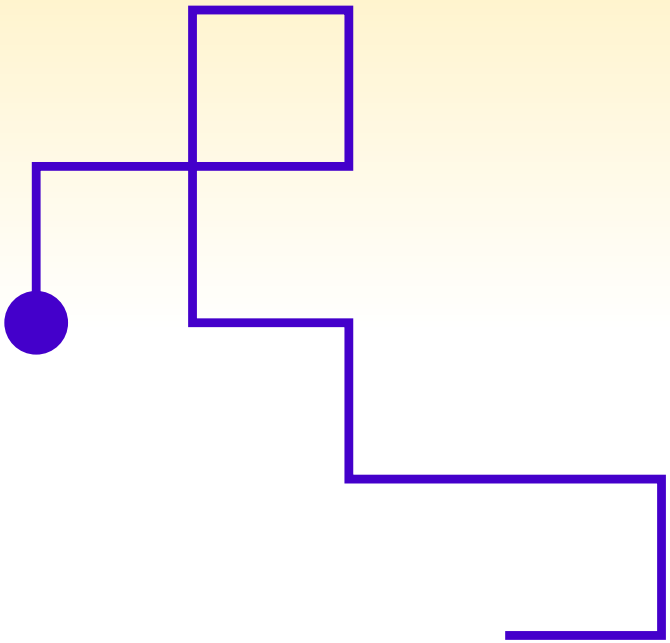
Simple random walk



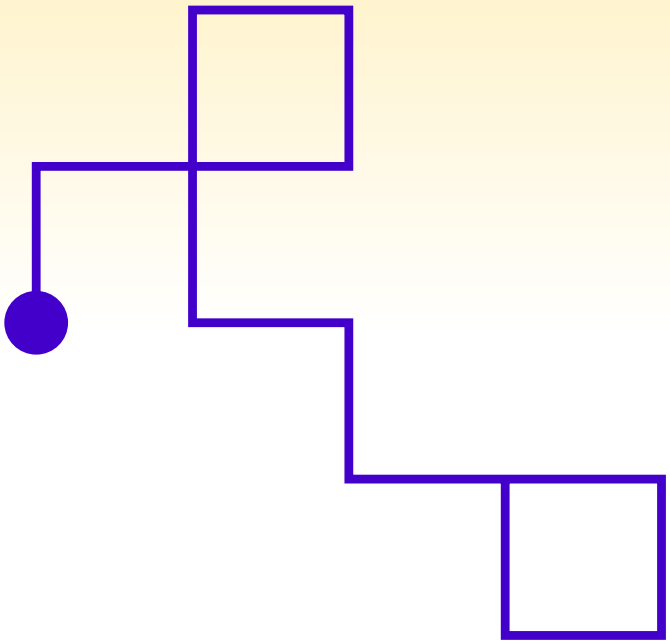
Simple random walk



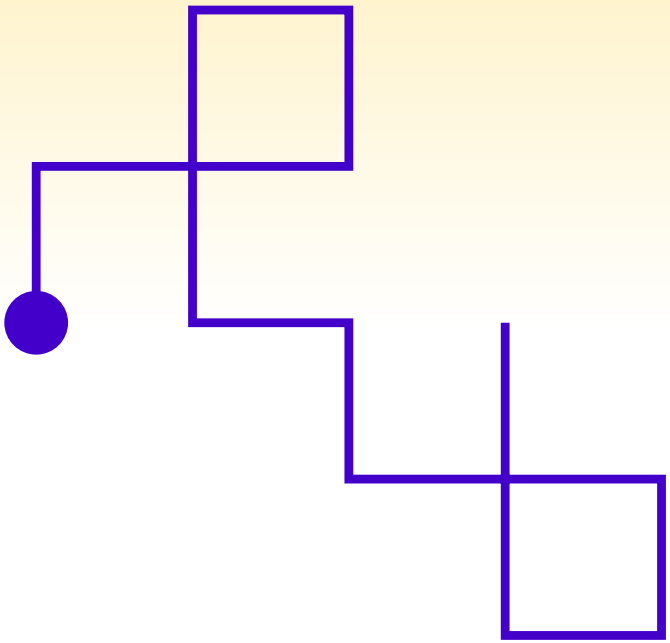
Simple random walk



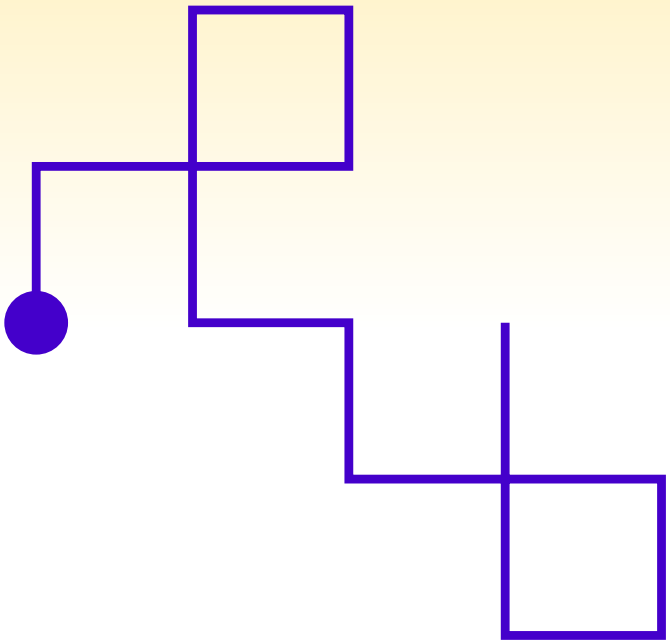
Simple random walk



Simple random walk

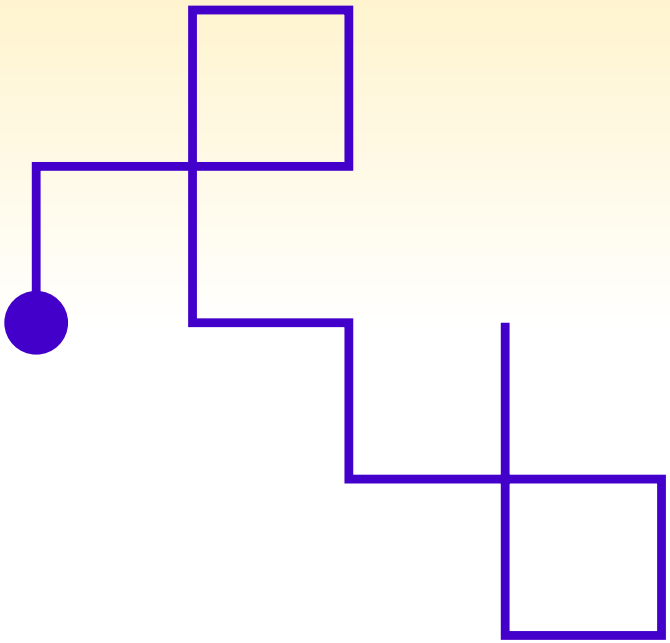


Simple random walk

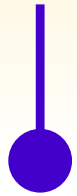


Simple random walk

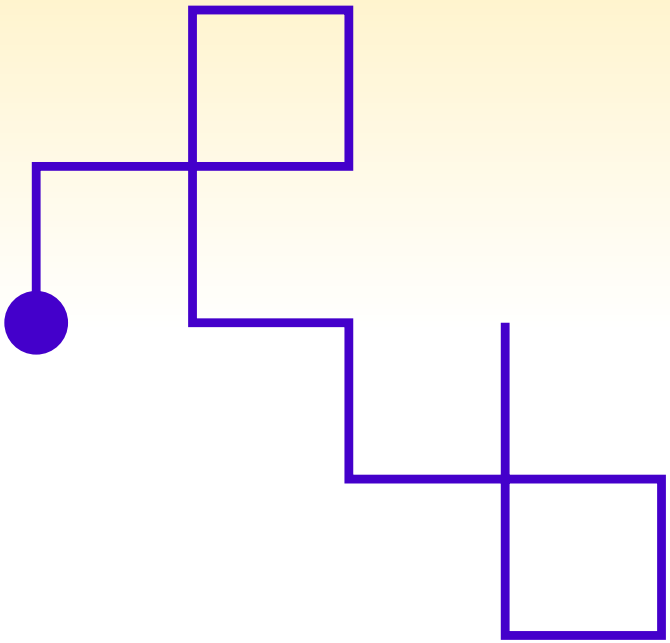
Self-avoiding walk



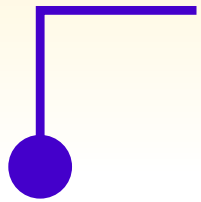
Simple random walk



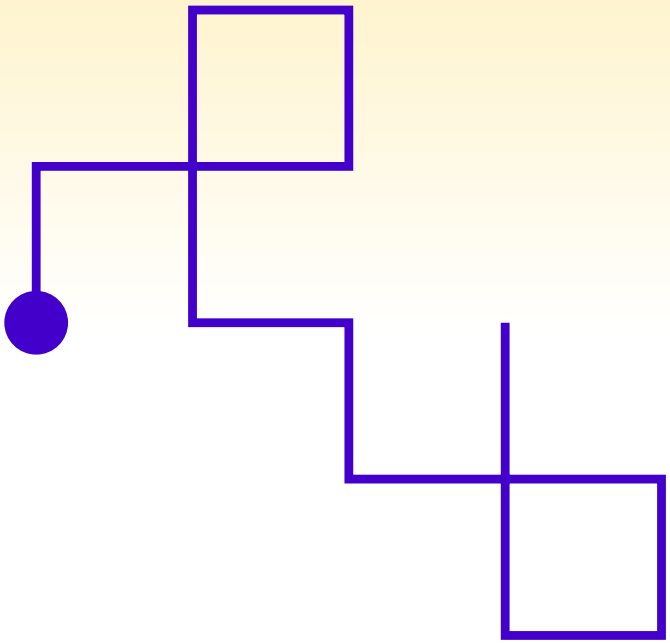
Self-avoiding walk



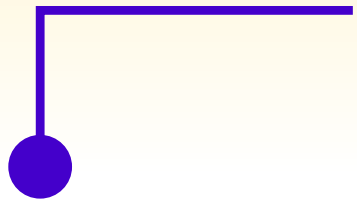
Simple random walk



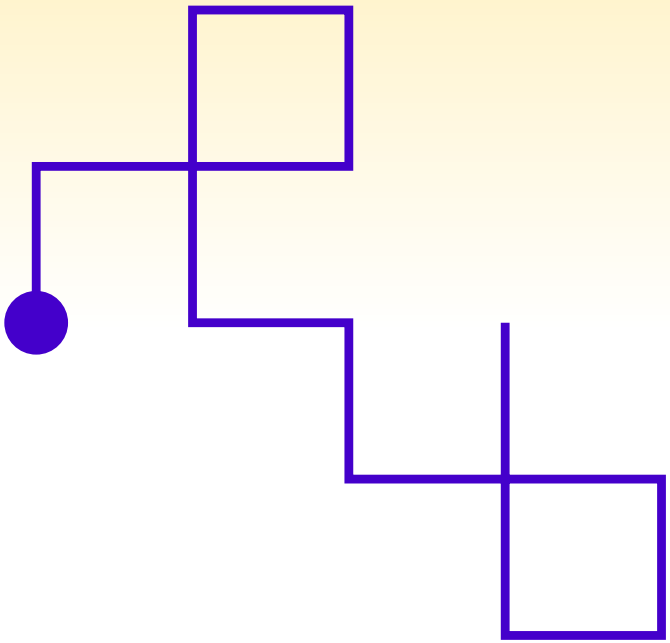
Self-avoiding walk



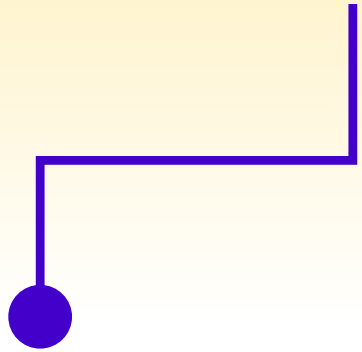
Simple random walk



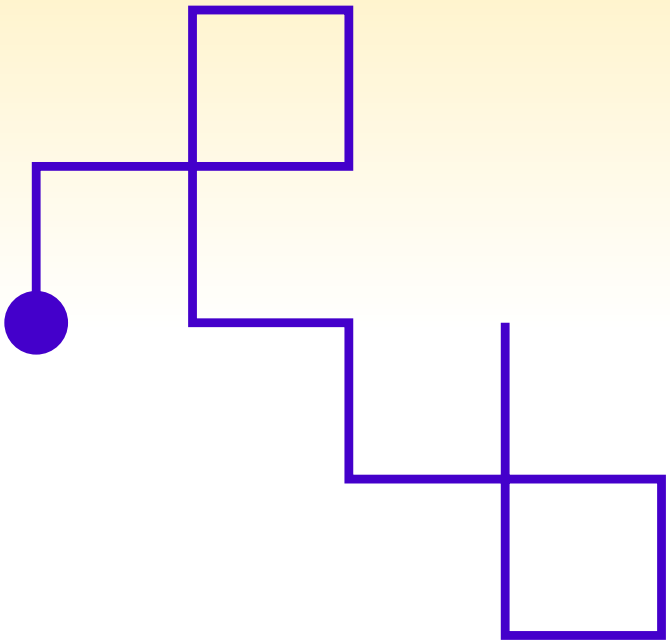
Self-avoiding walk



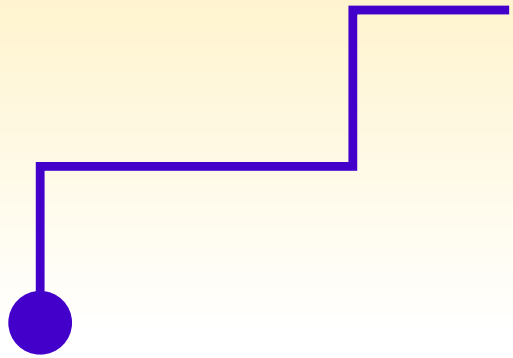
Simple random walk



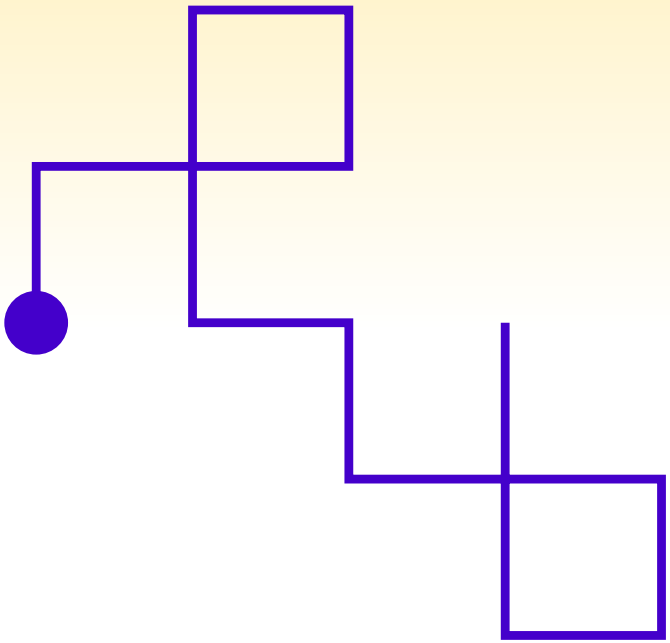
Self-avoiding walk



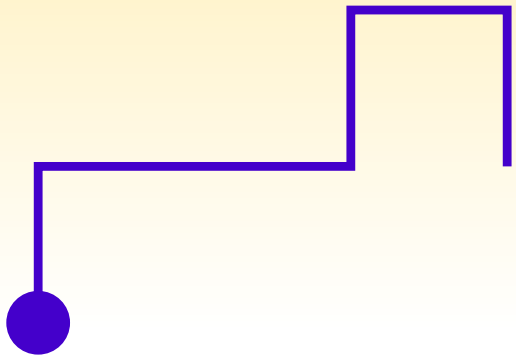
Simple random walk



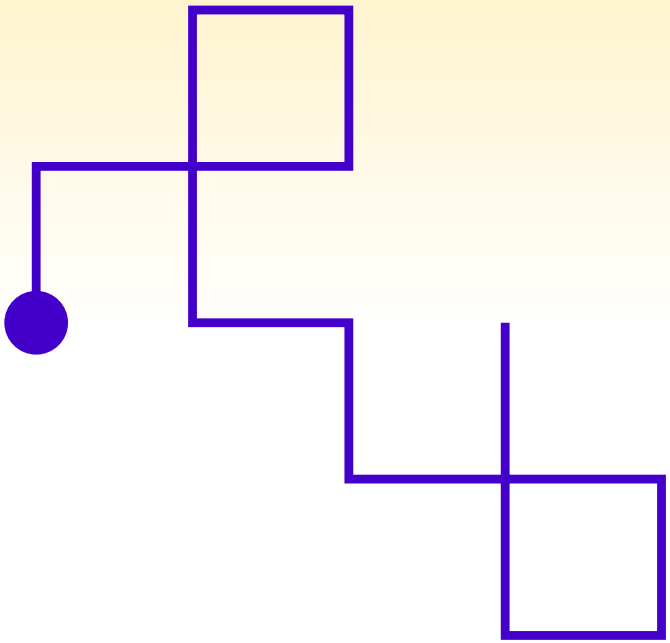
Self-avoiding walk



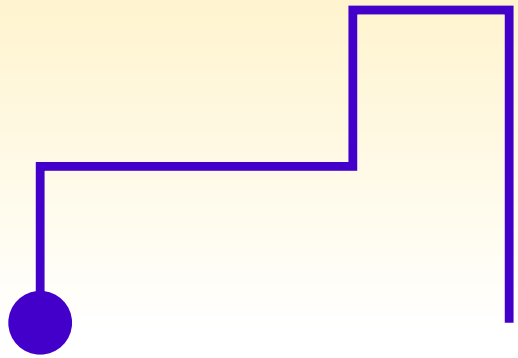
Simple random walk



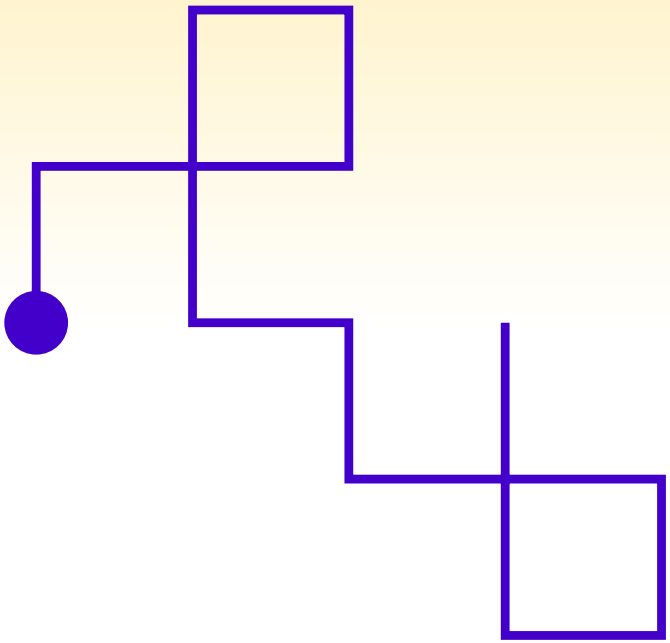
Self-avoiding walk



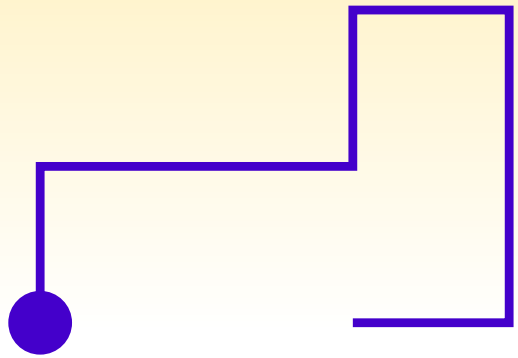
Simple random walk



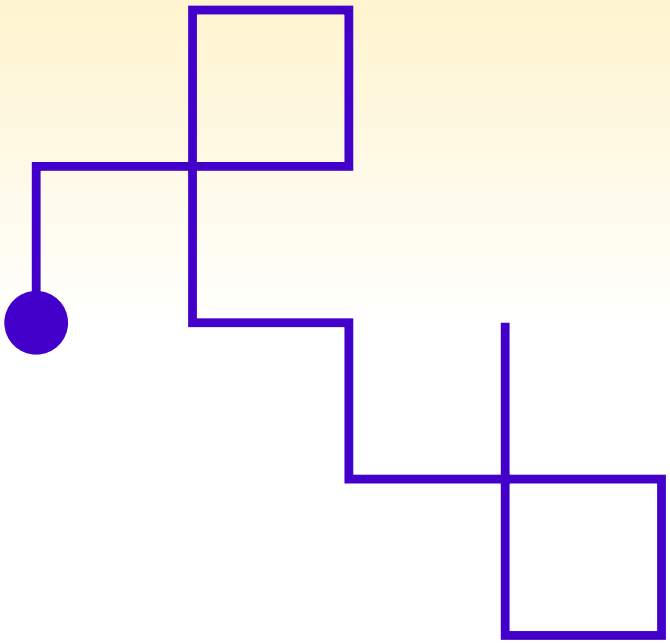
Self-avoiding walk



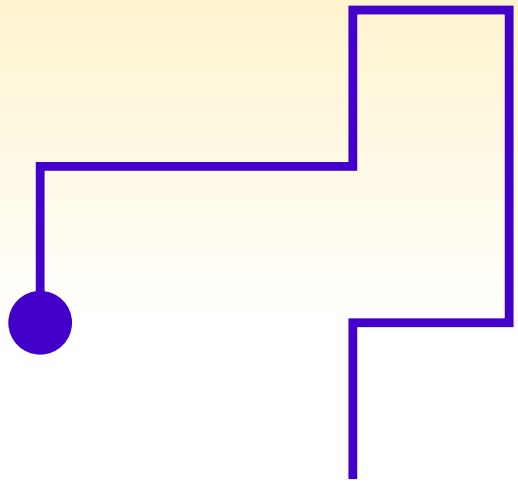
Simple random walk



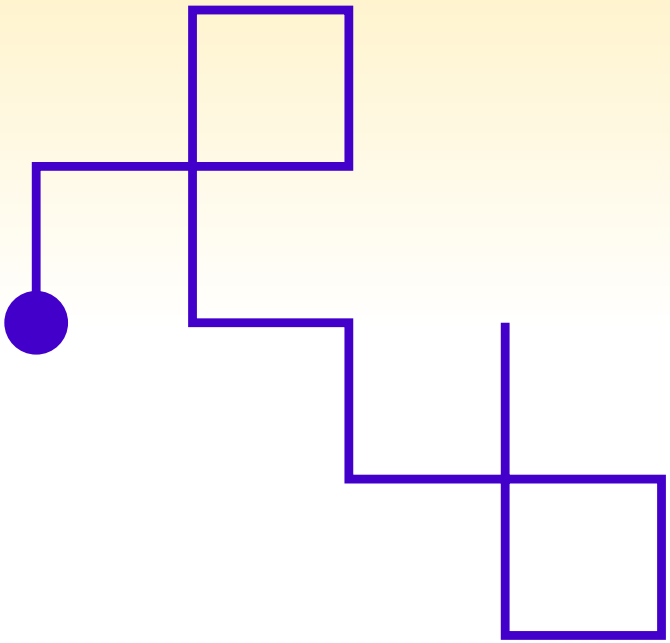
Self-avoiding walk



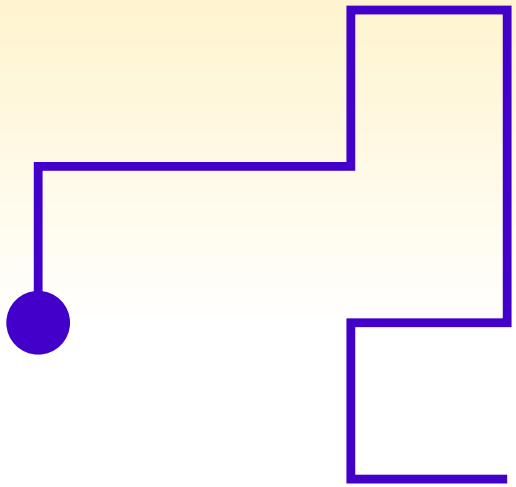
Simple random walk



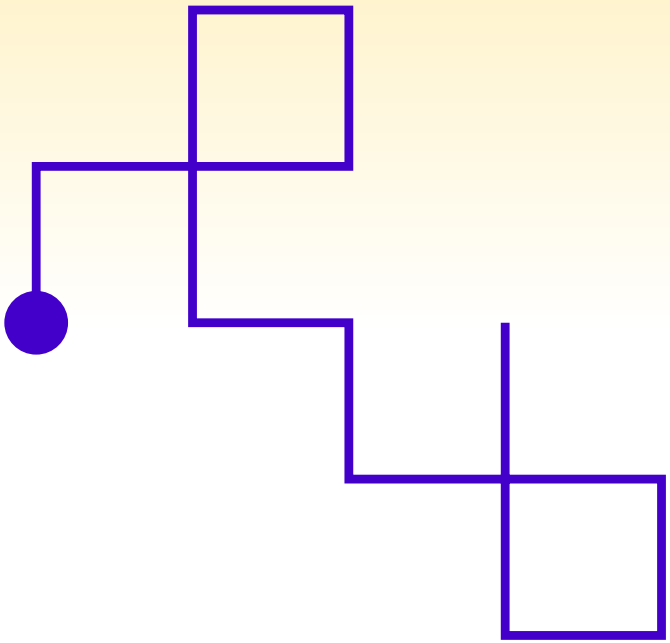
Self-avoiding walk



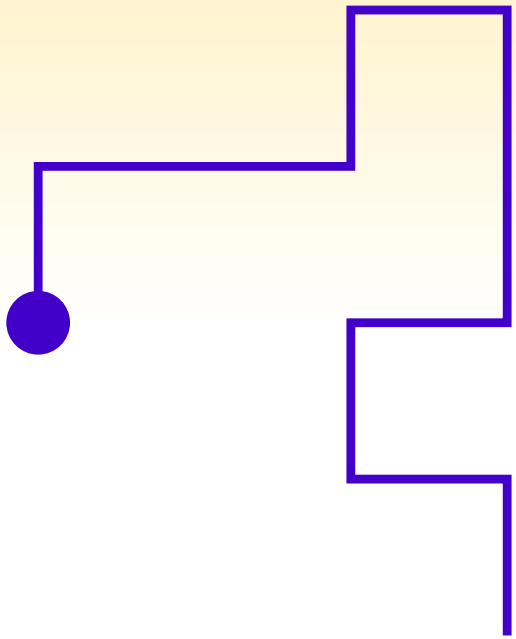
Simple random walk



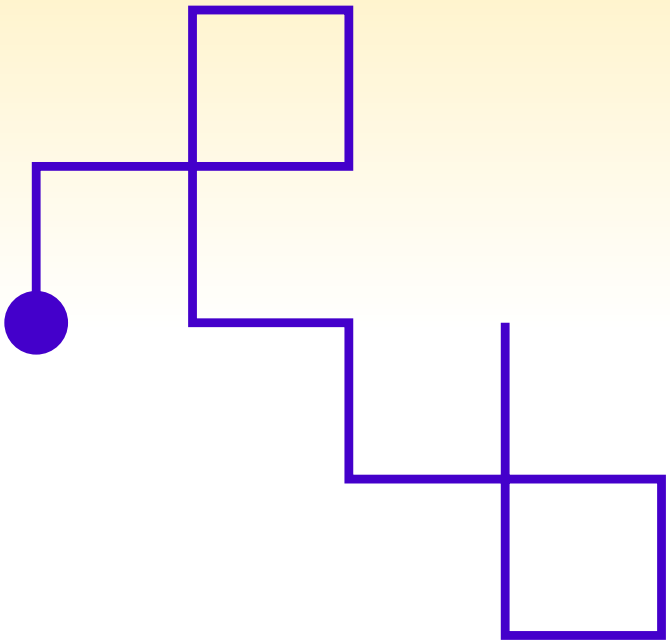
Self-avoiding walk



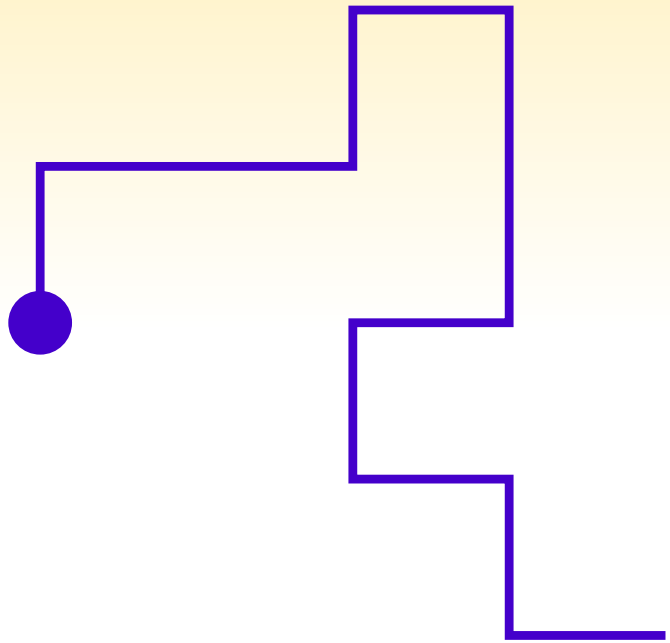
Simple random walk



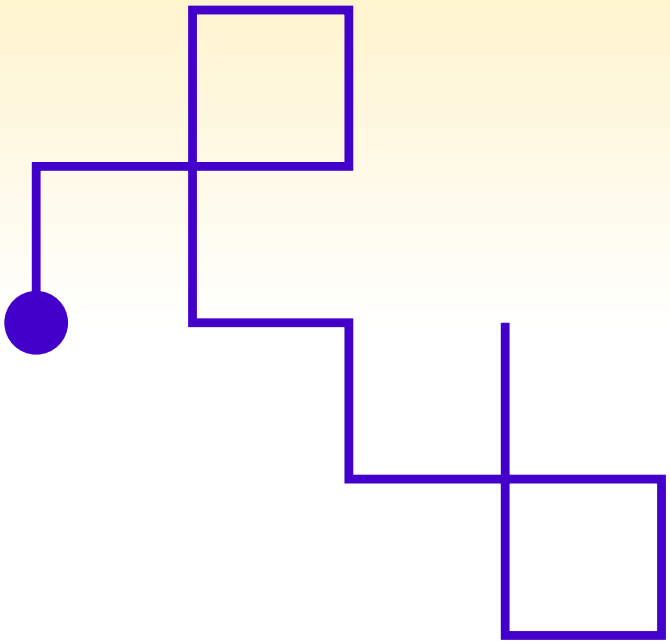
Self-avoiding walk



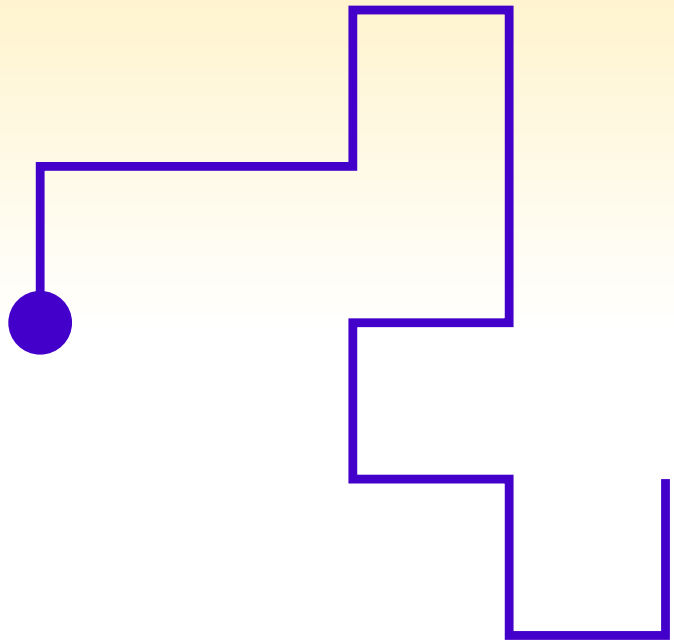
Simple random walk



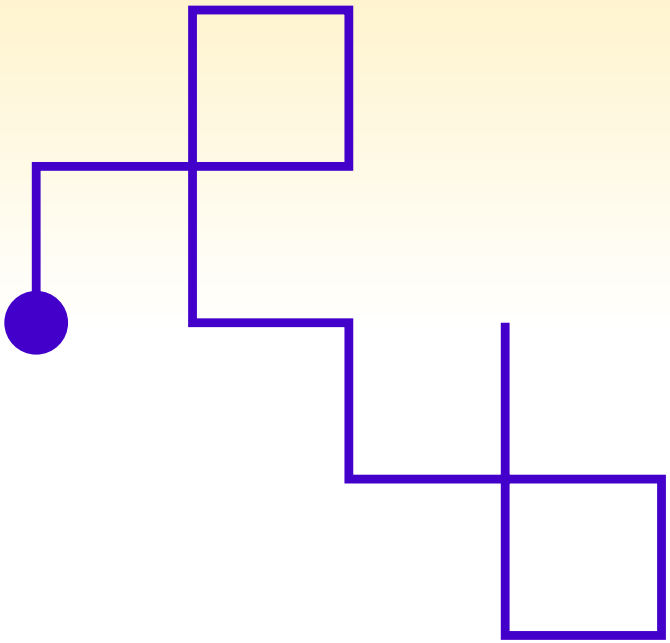
Self-avoiding walk



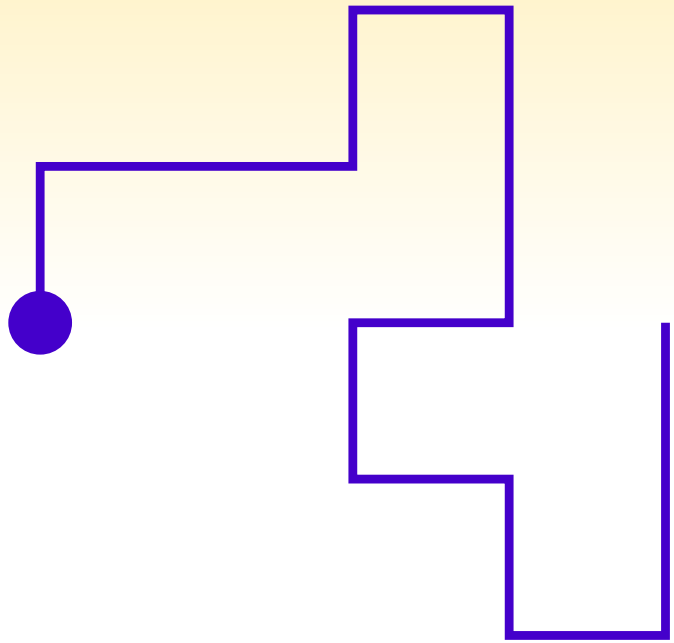
Simple random walk



Self-avoiding walk

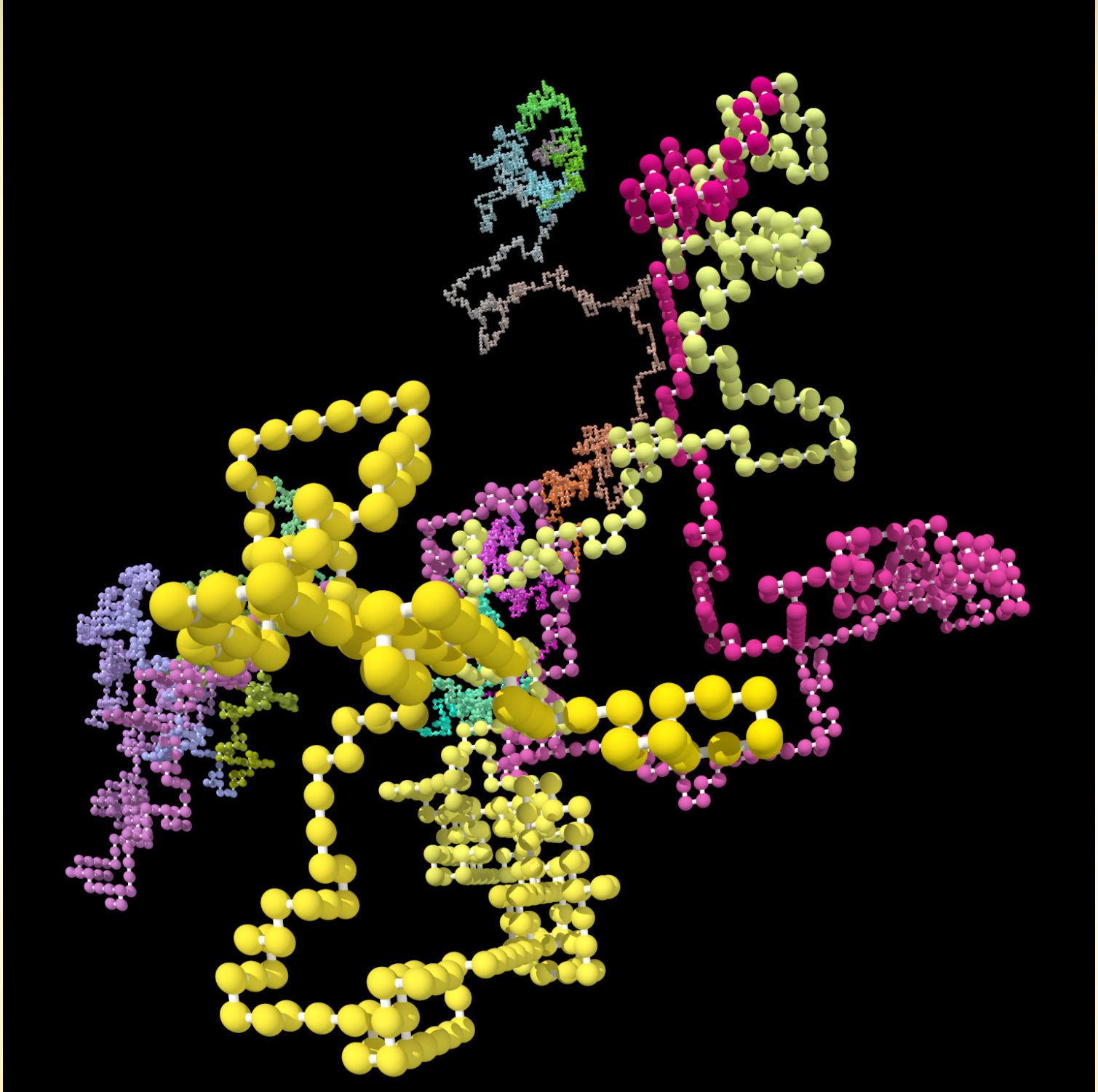


Simple random walk



Self-avoiding walk

A typical SAW of 5000 steps on the simple cubic lattice:



- Polymers can't have two monomers in the same place - key property of polymers in a good solvent.

- Polymers can't have two monomers in the same place - key property of polymers in a good solvent.
- SAW *exactly* captures universal properties of polymers.

- Polymers can't have two monomers in the same place - key property of polymers in a good solvent.
- SAW *exactly* captures universal properties of polymers.
- E.g. typical size of a SAW / polymer grows with the number of monomers, N, as:

$$\mathbf{R} = DN^{\nu}$$

- Polymers can't have two monomers in the same place - key property of polymers in a good solvent.
- SAW *exactly* captures universal properties of polymers.
- E.g. typical size of a SAW / polymer grows with the number of monomers, N, as:

$$\mathbf{R} = DN^{\nu}$$

- D is model dependent, but the critical exponent ν is a universal quantity.

Pivot algorithm

- No exact solution for SAW.

Pivot algorithm

- No exact solution for SAW.
- Markov chain Monte Carlo (MCMC) sampling very powerful for SAW, especially for $d = 3$.

Pivot algorithm

- No exact solution for SAW.
- Markov chain Monte Carlo (MCMC) sampling very powerful for SAW, especially for $d = 3$.
- Estimate physical properties of system by sampling from all possible configurations (state space).

Pivot algorithm

- No exact solution for SAW.
- Markov chain Monte Carlo (MCMC) sampling very powerful for SAW, especially for $d = 3$.
- Estimate physical properties of system by sampling from all possible configurations (state space).
- Basic idea: generate new configurations by deforming current configuration via a "move".

Pivot algorithm

- Local moves make a small deformation, e.g. adding or removing a monomer, $O(N^2)$ moves to get an "essentially new" configuration.

Pivot algorithm

- Local moves make a small deformation, e.g. adding or removing a monomer, $O(N^2)$ moves to get an "essentially new" configuration.
- Global moves can do much better.

Pivot algorithm

- Local moves make a small deformation, e.g. adding or removing a monomer, $O(N^2)$ moves to get an "essentially new" configuration.
- Global moves can do much better.
- Pivot move: $O(1)$ successful moves for an essentially new configuration.

Pivot algorithm

- Local moves make a small deformation, e.g. adding or removing a monomer, $O(N^2)$ moves to get an "essentially new" configuration.
- Global moves can do much better.
- Pivot move: $O(1)$ successful moves for an essentially new configuration.
- Lal (1969) invented pivot algorithm, but key paper is by Madras and Sokal (1988).

Pivot algorithm

- Procedure:

Pivot algorithm

- Procedure:
  - Choose a pivot site at random

Pivot algorithm

- Procedure:
  - Choose a pivot site at random
  - Then rotate or reflect one of the two parts of the walk.

Pivot algorithm

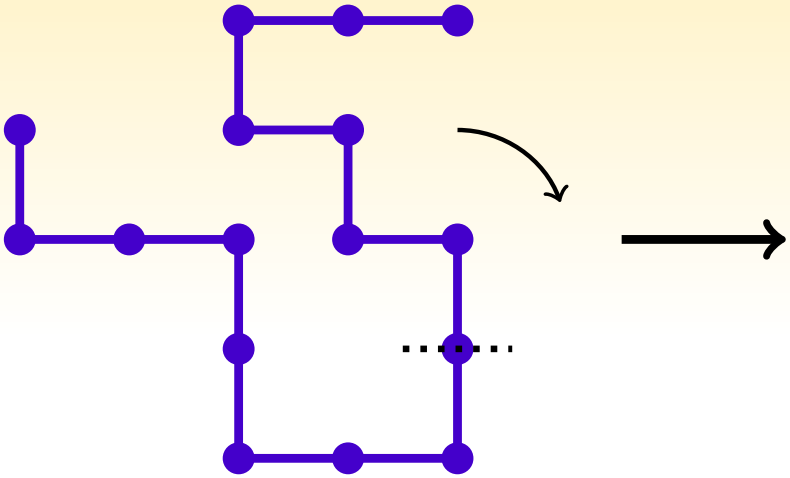
- Procedure:
  - Choose a pivot site at random
  - Then rotate or reflect one of the two parts of the walk.
  - Retain new walk if it is self-avoiding, otherwise restore original walk.

Pivot algorithm

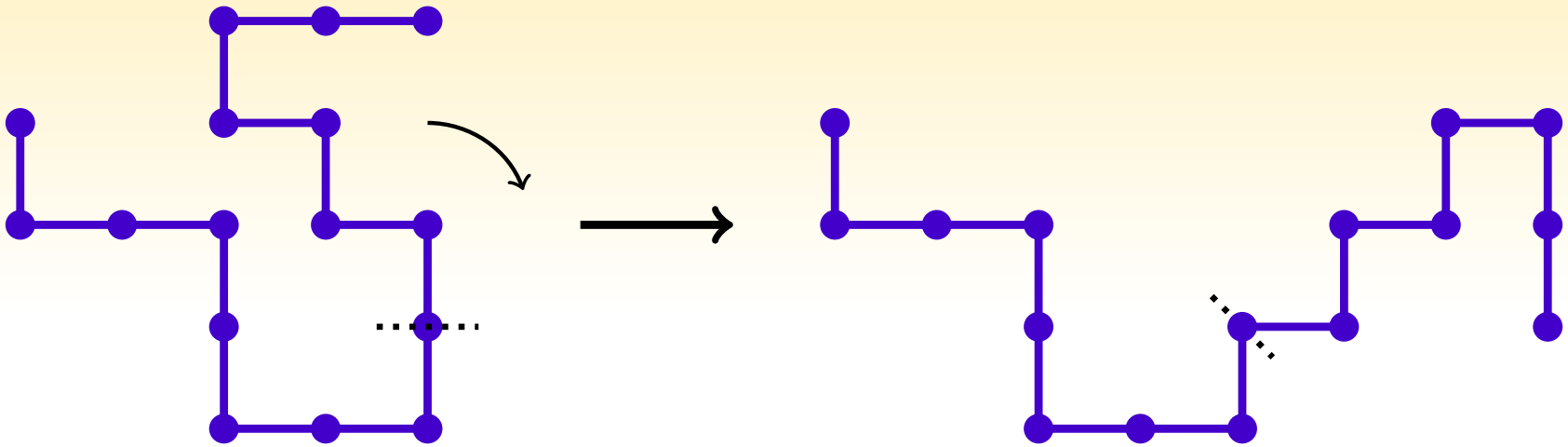
- Procedure:
  - Choose a pivot site at random
  - Then rotate or reflect one of the two parts of the walk.
  - Retain new walk if it is self-avoiding, otherwise restore original walk.
- "Global" because on average half of the monomers are moved.

Pivot algorithm

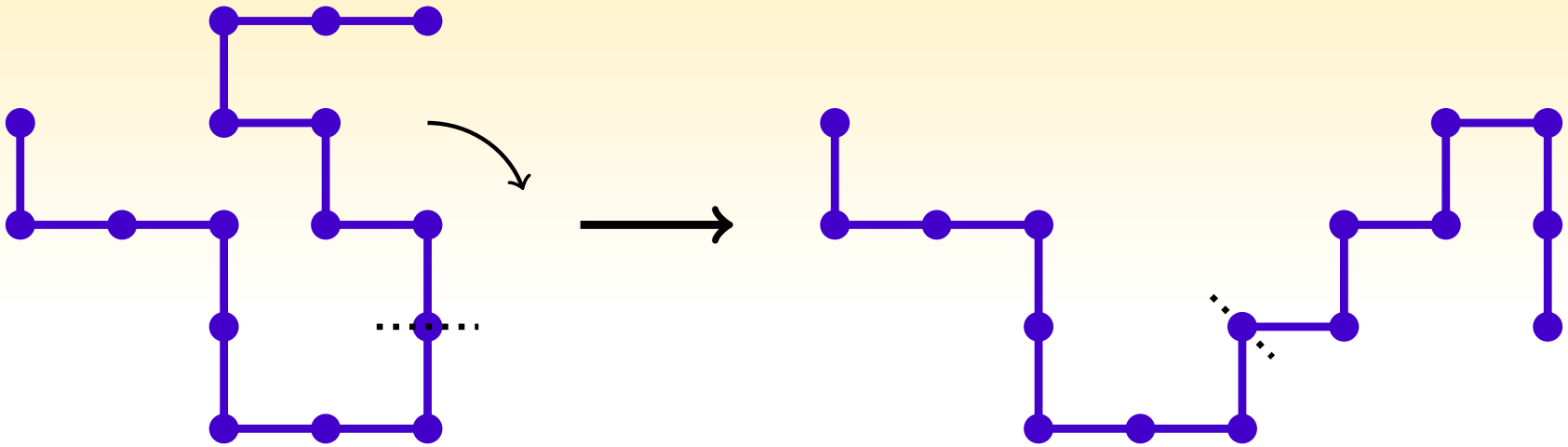
- Procedure:
  - Choose a pivot site at random
  - Then rotate or reflect one of the two parts of the walk.
  - Retain new walk if it is self-avoiding, otherwise restore original walk.
- "Global" because on average half of the monomers are moved.
- Ergodic, samples SAWs uniformly at random.



Example pivot move



Example pivot move



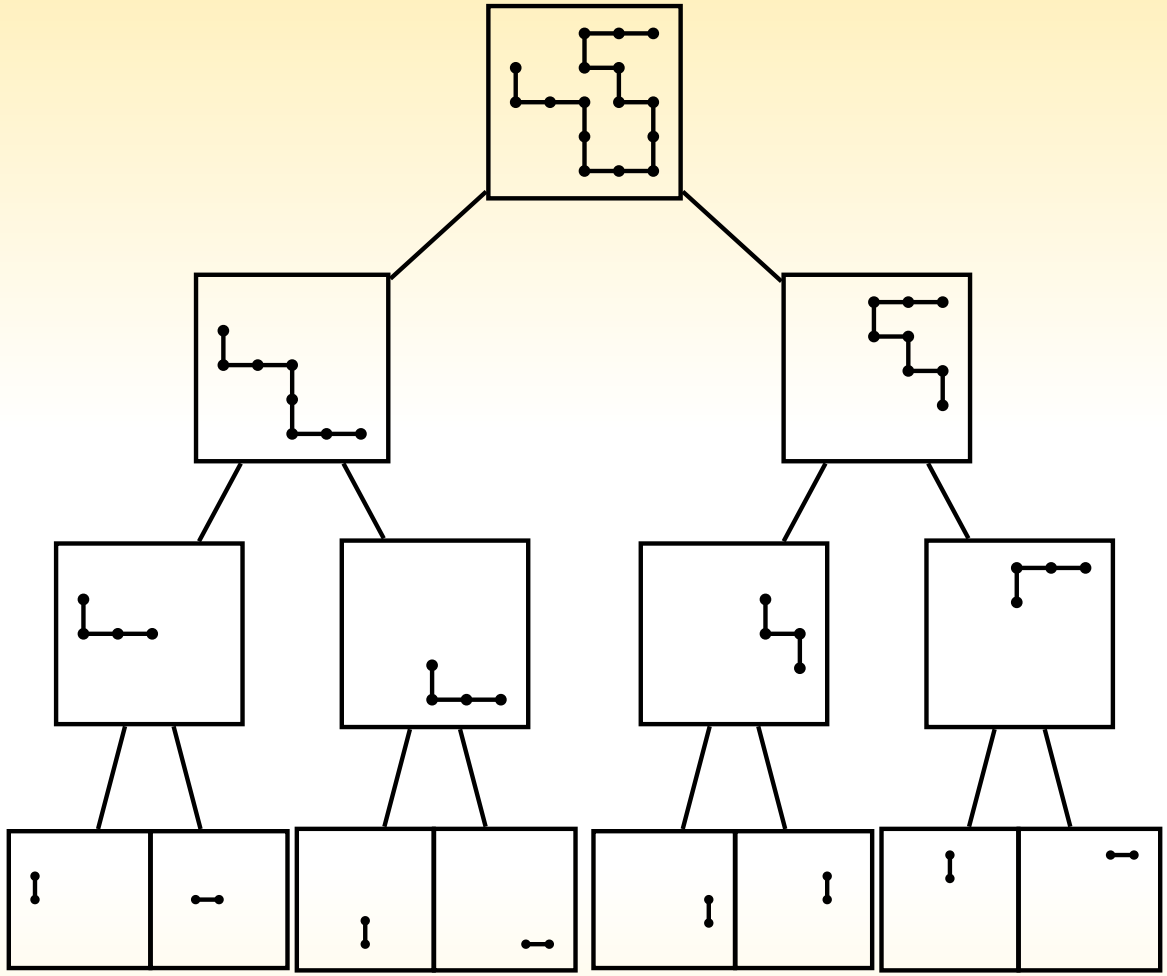
Example pivot move

Run simulation

- Time $O(N)$ to write down an N-step walk, so this must be best possible for pivot move?

- Time $O(N)$ to write down an N-step walk, so this must be best possible for pivot move?
- In fact, don't need to write down! $\Rightarrow o(N)$ (Kennedy, 2002)

- Time $O(N)$ to write down an N-step walk, so this must be best possible for pivot move?
- In fact, don't need to write down! $\Rightarrow o(N)$ (Kennedy, 2002)
- Bookkeeping can be handled efficiently in binary tree structure. $\Rightarrow O(\log N)$ (C., 2010)



SAW-tree representation of a walk.

Key properties of the SAW-tree data structure.

- Each node contains:

Key properties of the SAW-tree data structure.

- Each node contains:
  - Symmetry information (for rotations / reflections);

Key properties of the SAW-tree data structure.

- Each node contains:
  - Symmetry information (for rotations / reflections);
  - "Bounding box" information (for intersection testing);

Key properties of the SAW-tree data structure.

- Each node contains:
  - Symmetry information (for rotations / reflections);
  - "Bounding box" information (for intersection testing);
  - Information about moments of positions, allowing for *exact* calculation of R_{e}^2 and R_{g}^2.

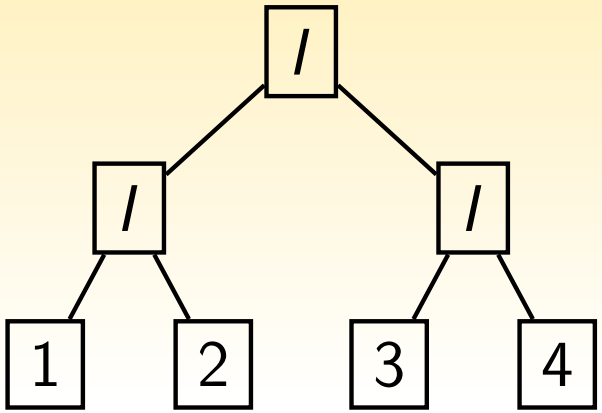
Key properties of the SAW-tree data structure.

- Each node contains:
  - Symmetry information (for rotations / reflections);
  - "Bounding box" information (for intersection testing);
  - Information about moments of positions, allowing for *exact* calculation of R_{e}^2 and R_{g}^2.
- Tree structure can be altered via "tree rotations", so that symmetry operations can be applied to any section of the walk.

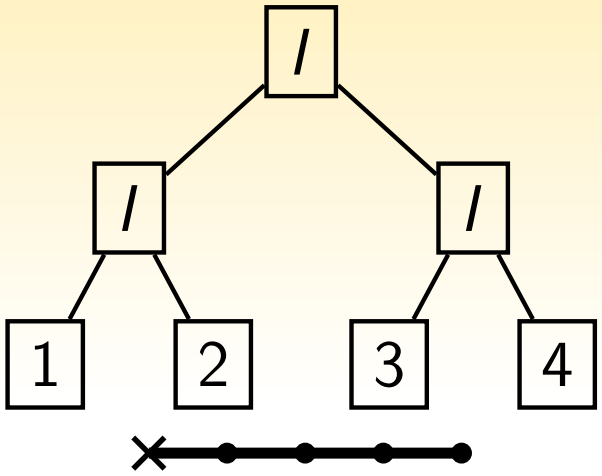
Key properties of the SAW-tree data structure.

- Each node contains:
  - Symmetry information (for rotations / reflections);
  - "Bounding box" information (for intersection testing);
  - Information about moments of positions, allowing for *exact* calculation of R_{e}^2 and R_{g}^2.
- Tree structure can be altered via "tree rotations", so that symmetry operations can be applied to any section of the walk.
- Binary tree has typical height $O(\log N)$.

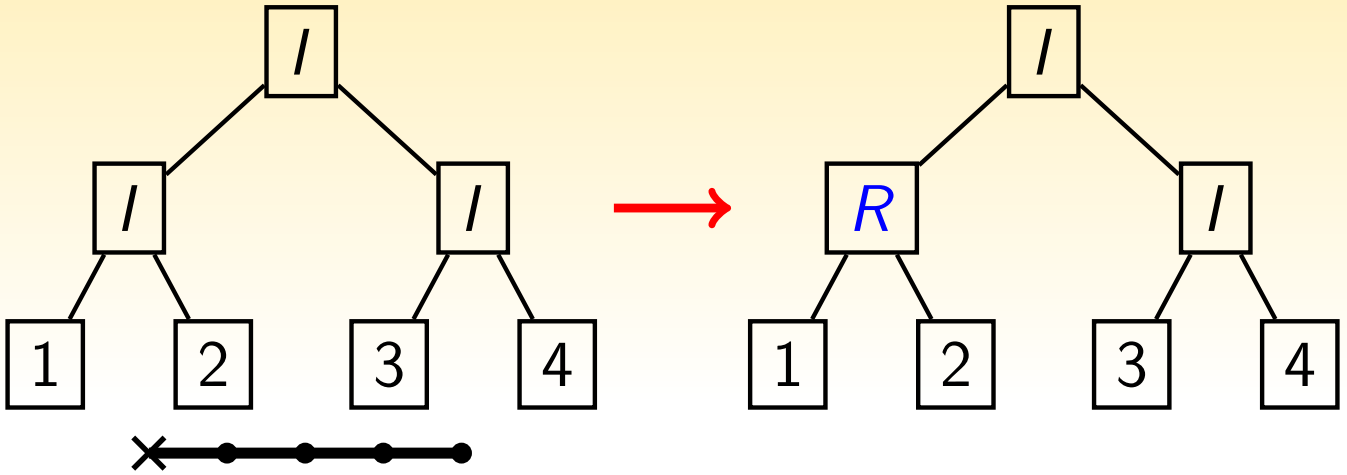
Example SAW-tree moves.



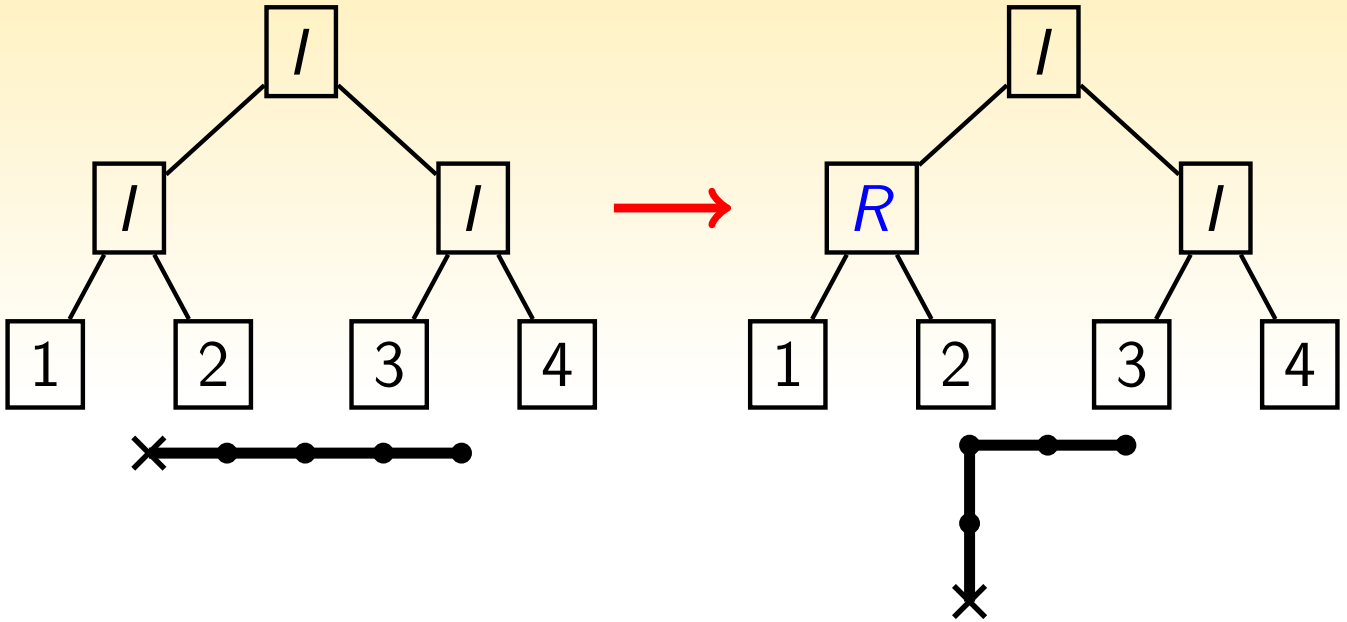
Example SAW-tree moves.



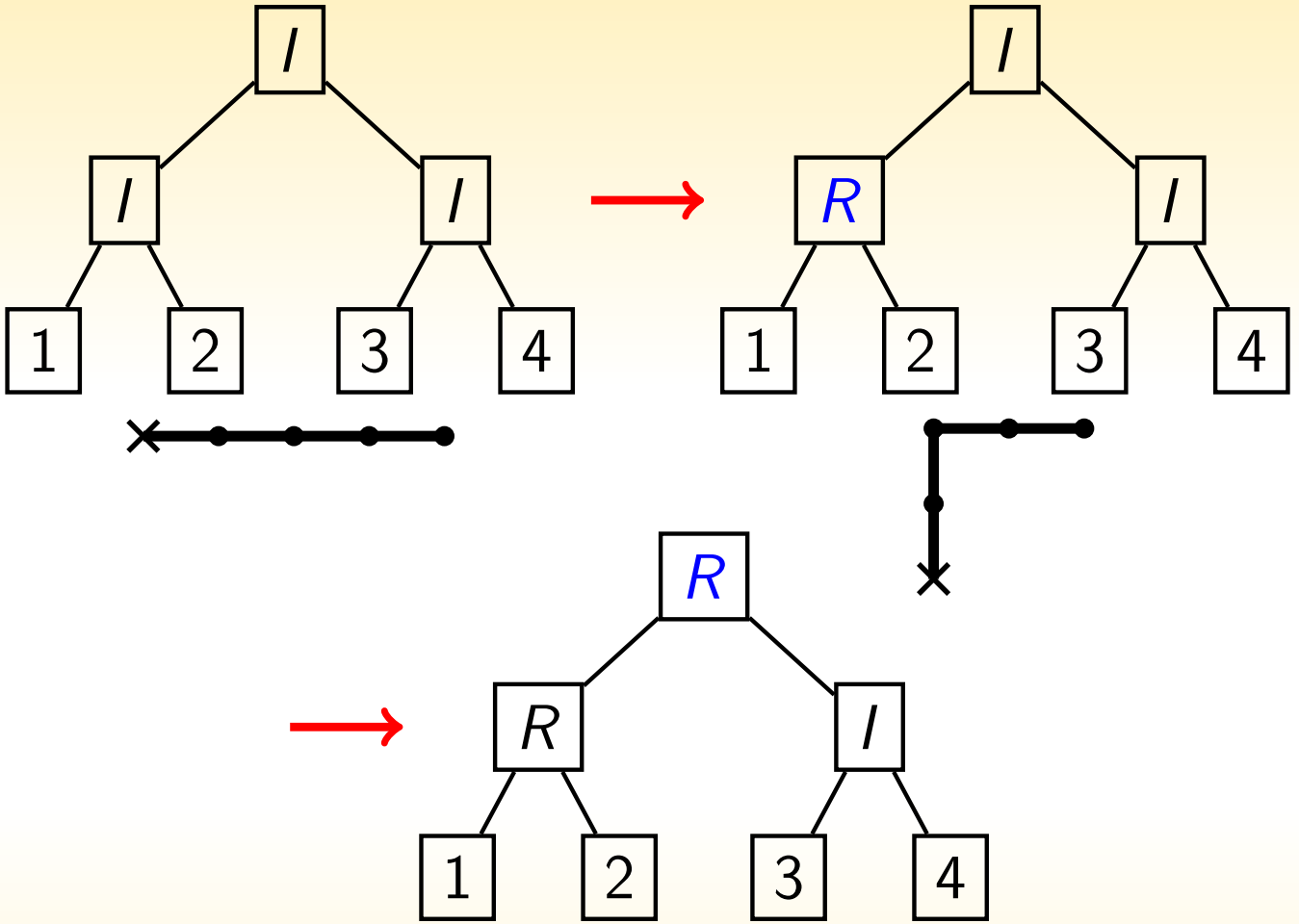
Example SAW-tree moves.



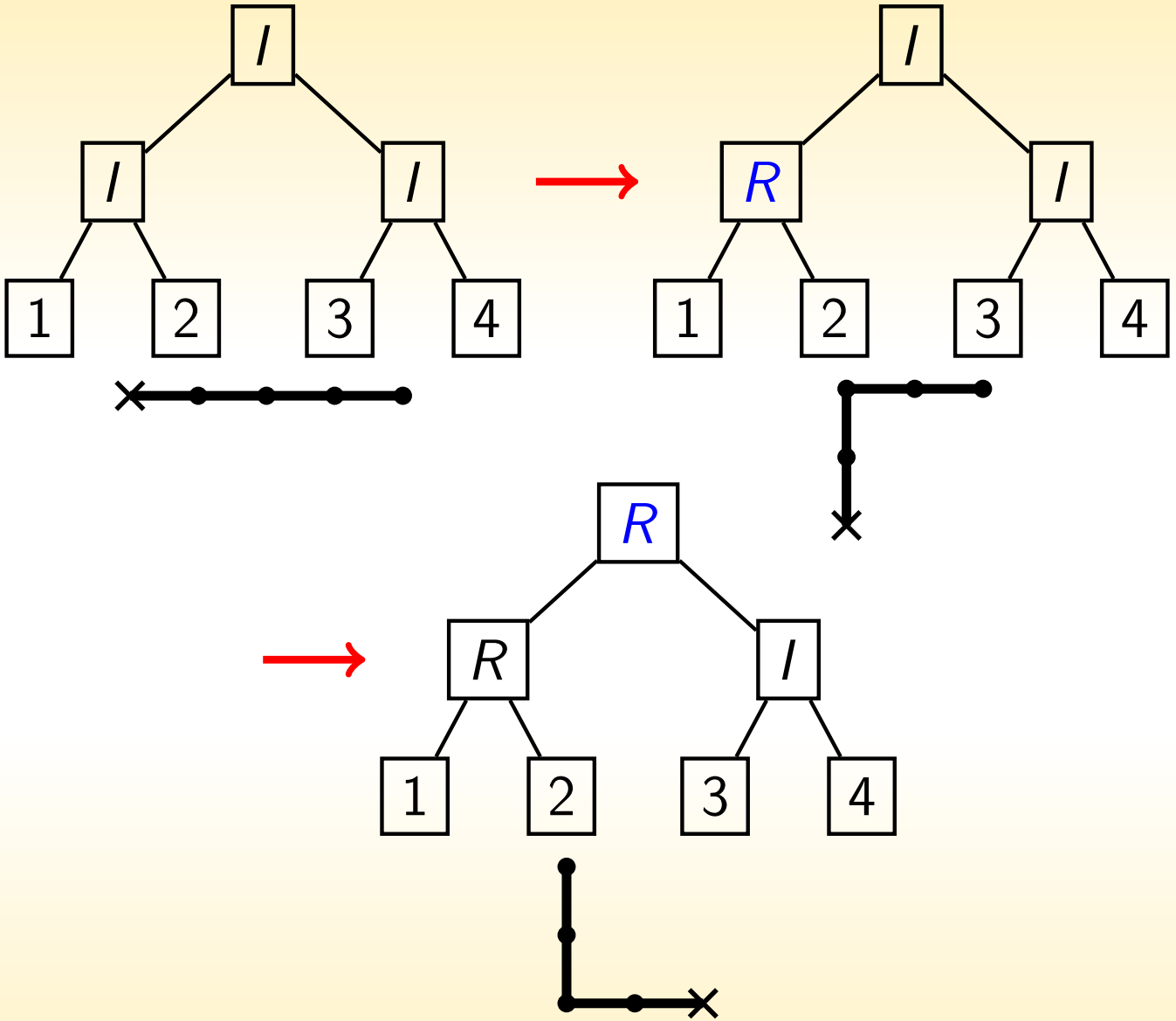
Example SAW-tree moves.



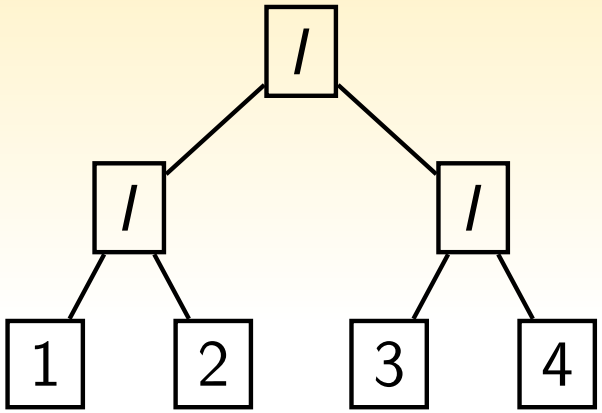
Example SAW-tree moves.



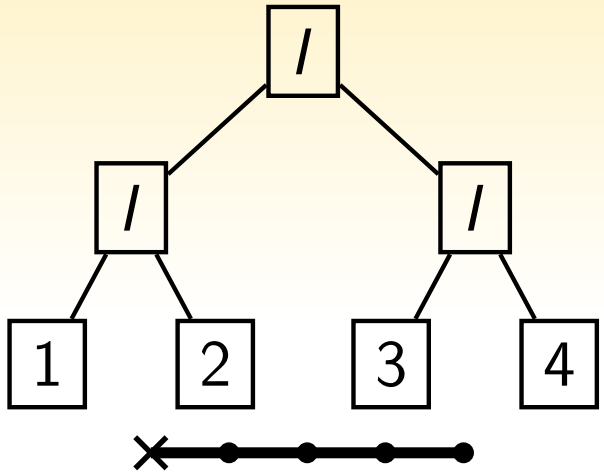
Example SAW-tree moves.



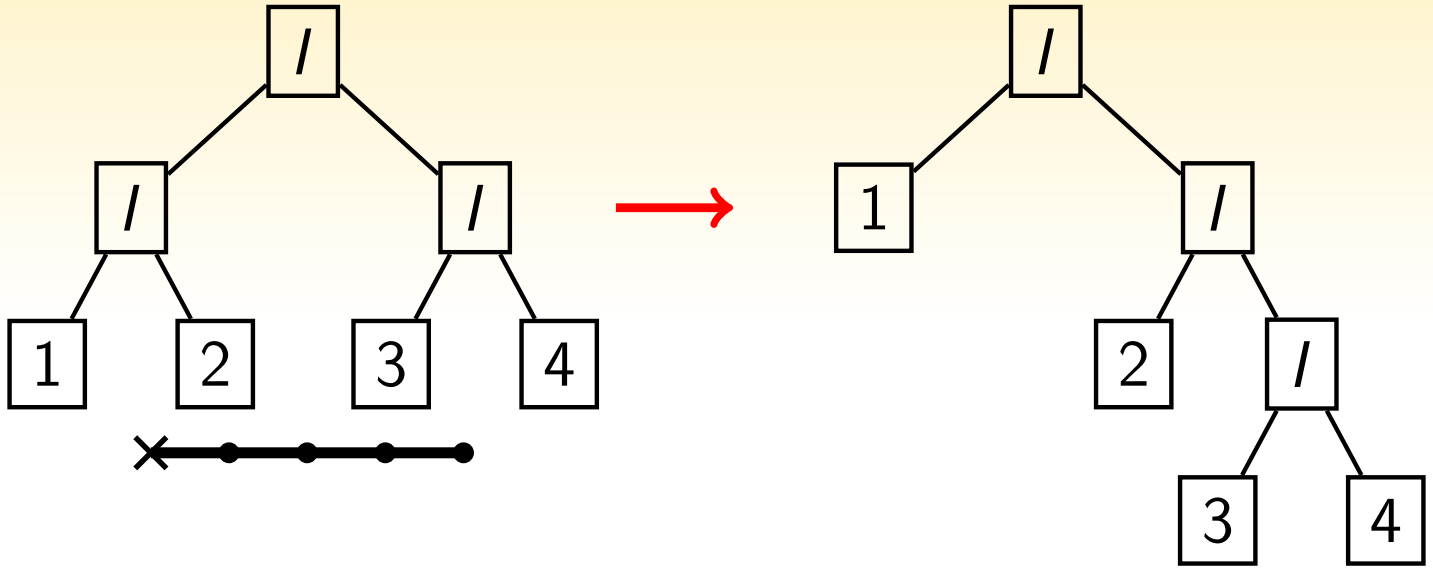
How do we apply symmetry to steps 2,3,4? Restructure the binary tree via a "tree rotation".



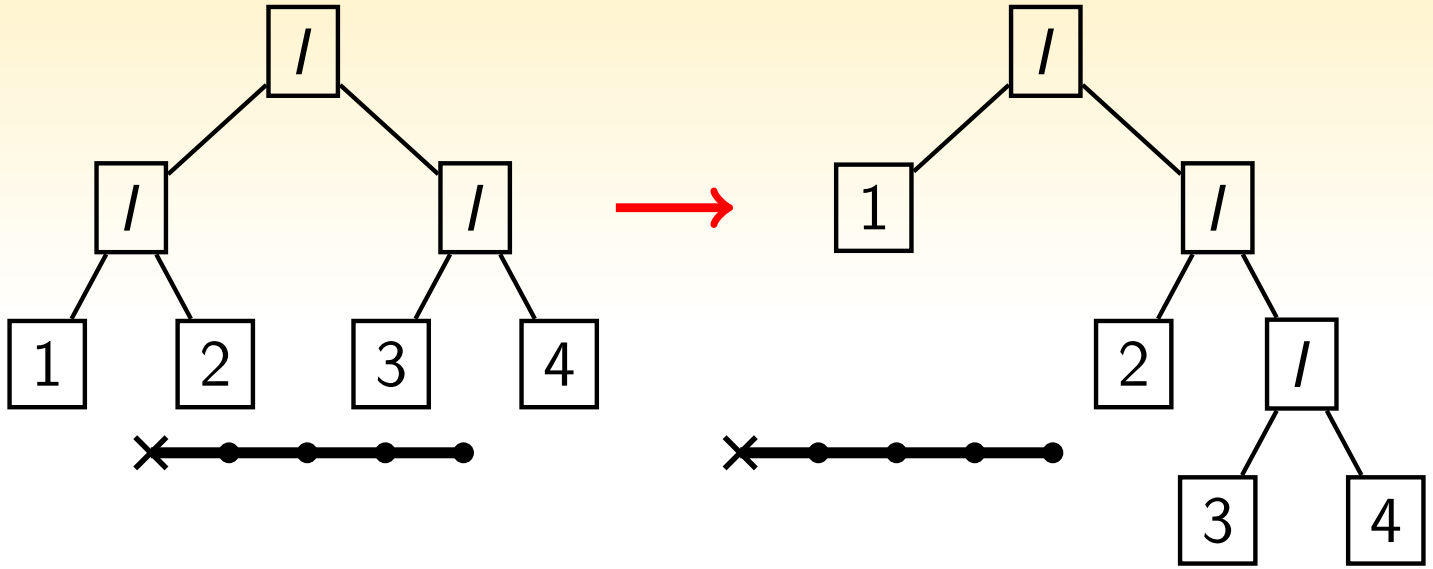
How do we apply symmetry to steps 2,3,4? Restructure the binary tree via a "tree rotation".



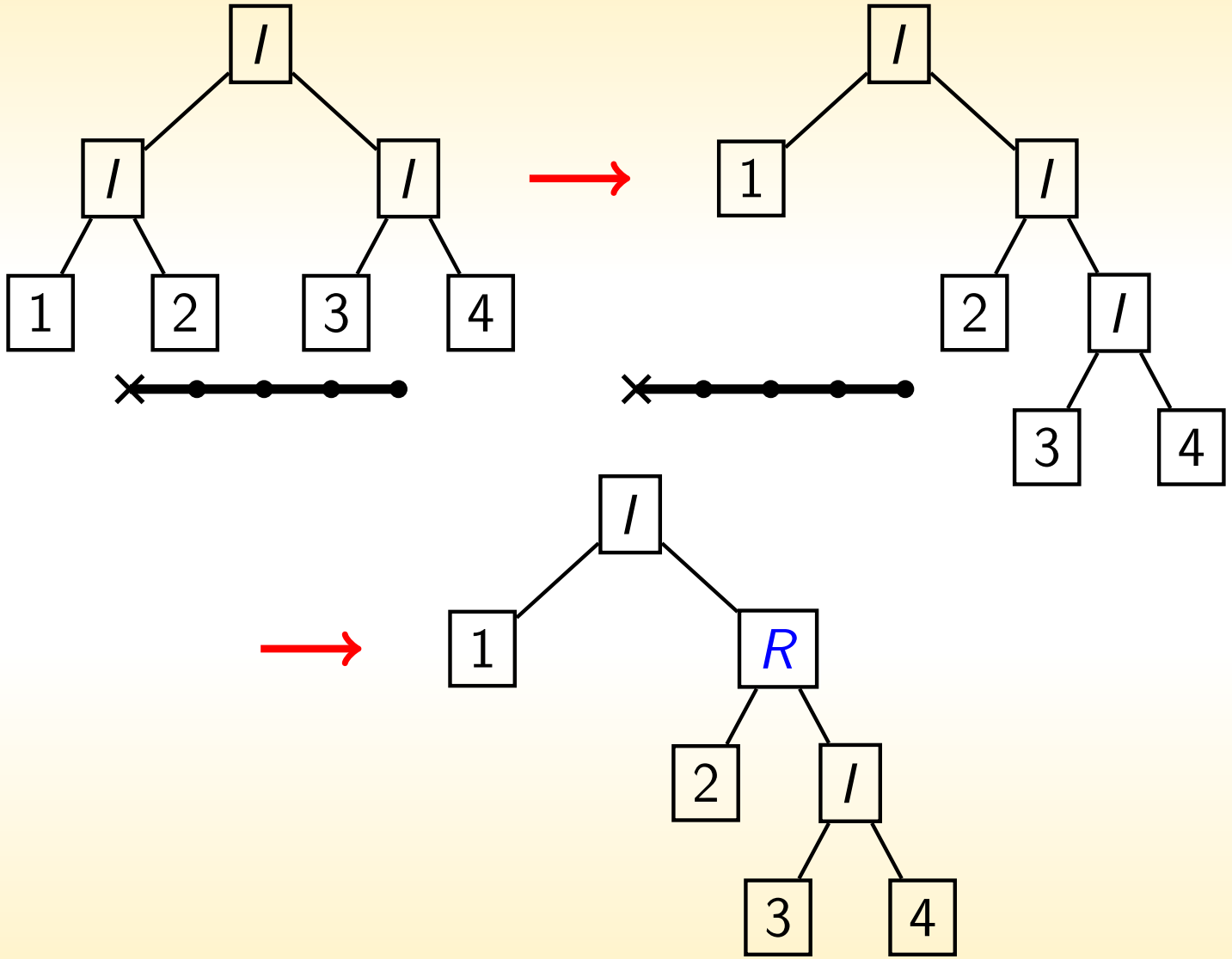
How do we apply symmetry to steps 2,3,4? Restructure the binary tree via a "tree rotation".



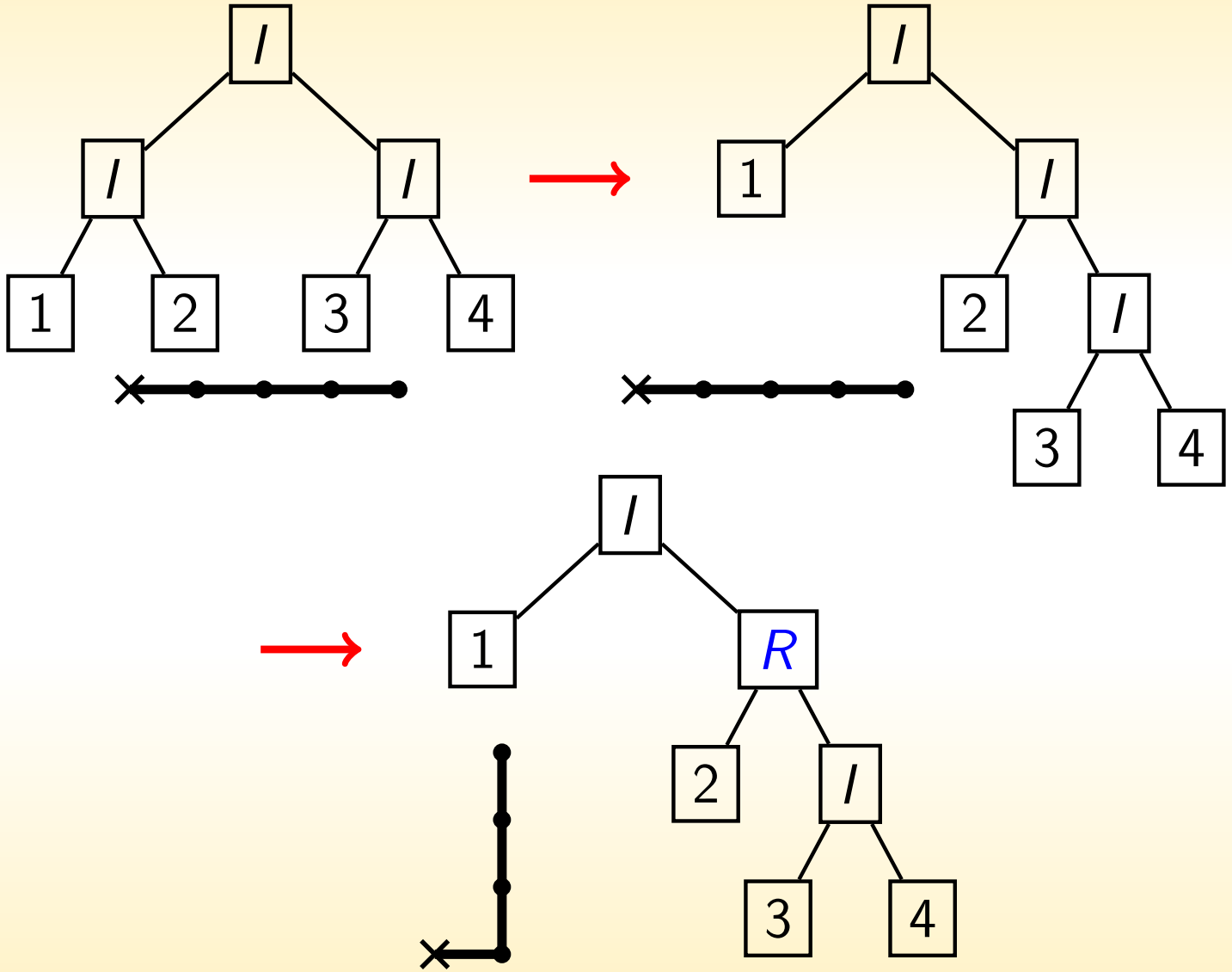
How do we apply symmetry to steps 2,3,4? Restructure the binary tree via a "tree rotation".



How do we apply symmetry to steps 2,3,4? Restructure the binary tree via a "tree rotation".



How do we apply symmetry to steps 2,3,4? Restructure the binary tree via a "tree rotation".



- With a binary tree implementation, pivot move has *global effect* for *local cost*.

- With a binary tree implementation, pivot move has *global effect* for *local cost*.
- $O(\log N)$ for:

- With a binary tree implementation, pivot move has *global effect* for *local cost*.
- $O(\log N)$ for:
  - Rotating part of the walk.

- With a binary tree implementation, pivot move has *global effect* for *local cost*.
- $O(\log N)$ for:
  - Rotating part of the walk.
  - Checking for self-intersections between two pieces.

- With a binary tree implementation, pivot move has *global effect* for *local cost*.
- $O(\log N)$ for:
  - Rotating part of the walk.
  - Checking for self-intersections between two pieces.
  - Calculating global observables such as R_{e}^2 and R_{g}^2.

- With a binary tree implementation, pivot move has *global effect* for *local cost*.
- $O(\log N)$ for:
  - Rotating part of the walk.
  - Checking for self-intersections between two pieces.
  - Calculating global observables such as R_{e}^2 and R_{g}^2.
- Very fast, can rapidly simulate SAW with many millions of steps.

- With a binary tree implementation, pivot move has *global effect* for *local cost*.
- $O(\log N)$ for:
  - Rotating part of the walk.
  - Checking for self-intersections between two pieces.
  - Calculating global observables such as R_{e}^2 and R_{g}^2.
- Very fast, can rapidly simulate SAW with many millions of steps.
- How can we extend this key idea - fast global moves - to other models?

Hamiltonian paths

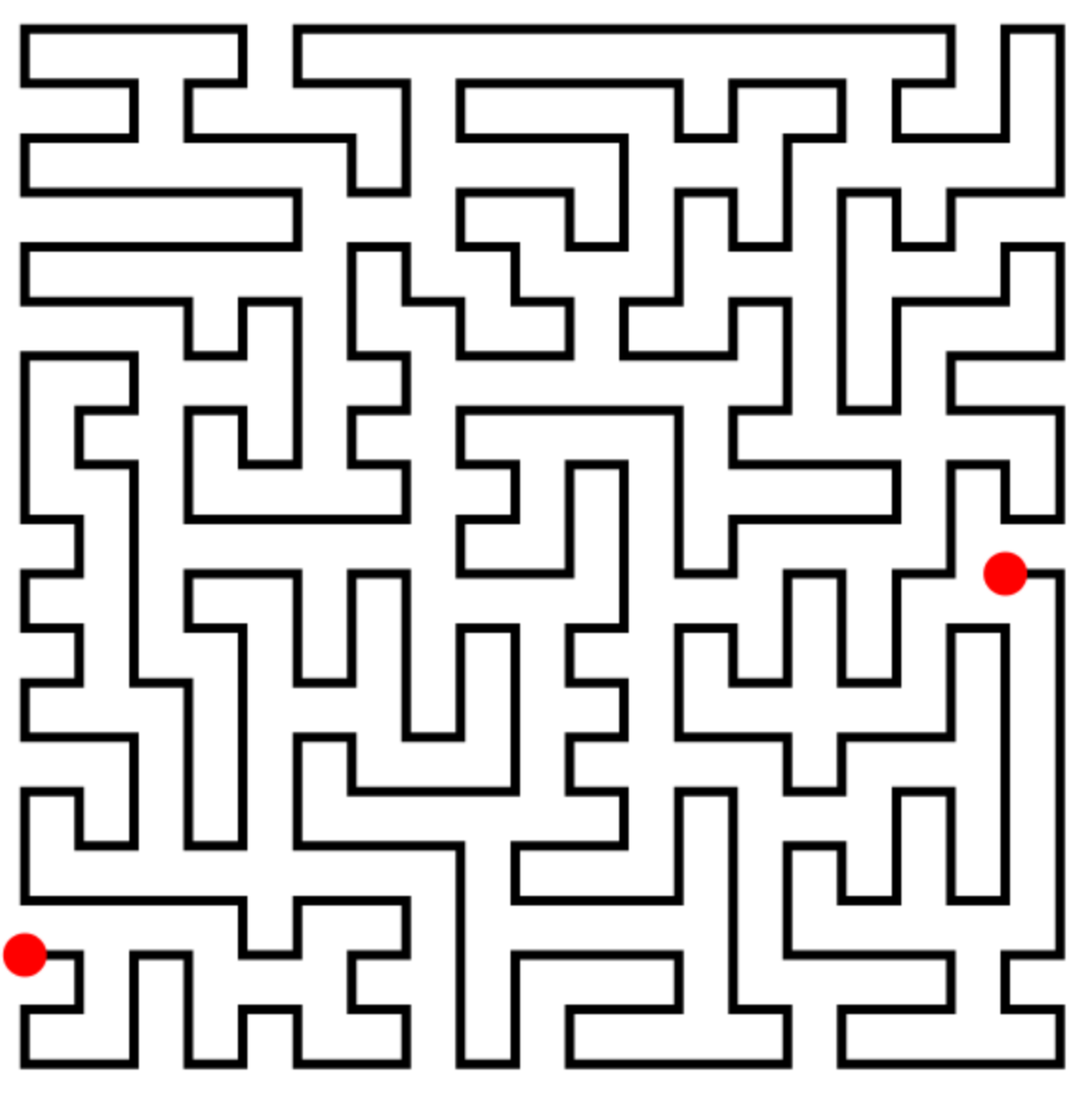
- Hamiltonian paths are self-avoiding walks which visit every site in a graph.

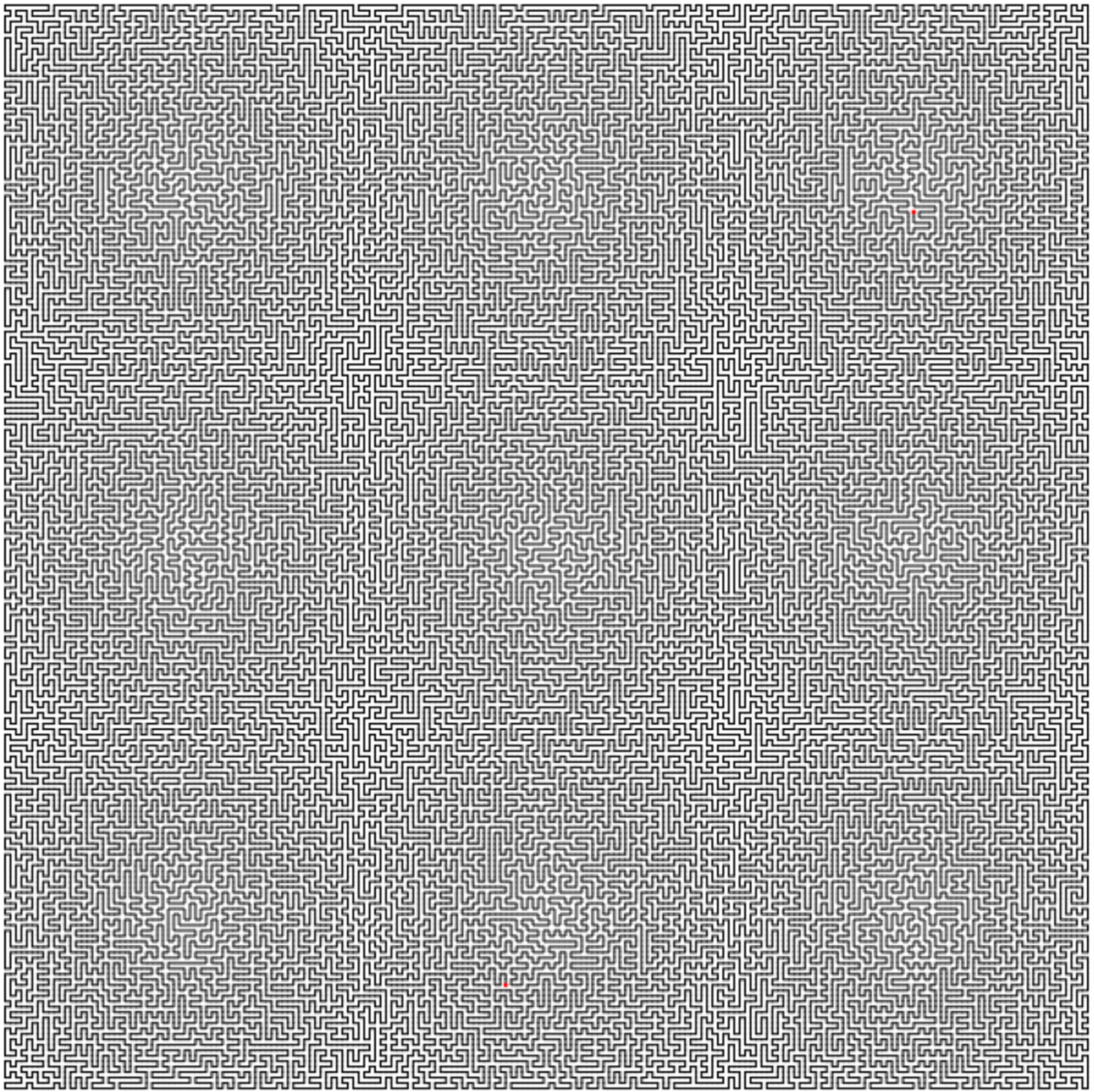
Hamiltonian paths

- Hamiltonian paths are self-avoiding walks which visit every site in a graph.
- Models crystal phase of polymers.

Hamiltonian paths

- Hamiltonian paths are self-avoiding walks which visit every site in a graph.
- Models crystal phase of polymers.
- Examples on 20×20 and 200×200 lattice (from generator at:
  `http://lattice.complex.unimelb.edu.au/hamiltonian_path`)





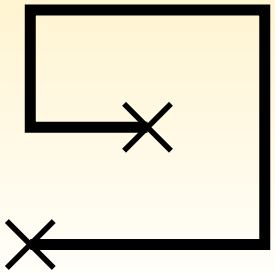
Hamiltonian paths

- Some effective MCMC moves for sampling dense polymers are based on changing the topology of configurations.

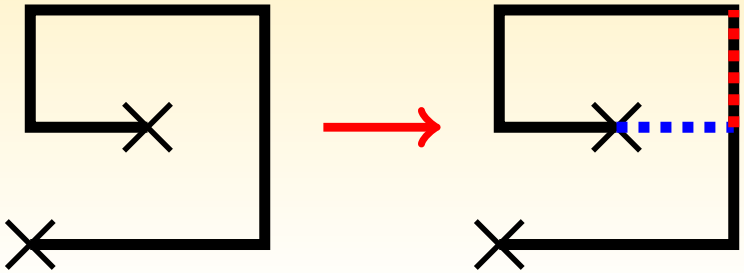
Hamiltonian paths

- Some effective MCMC moves for sampling dense polymers are based on changing the topology of configurations.
- One example: backbite move, which is likely to be ergodic for the simple cubic lattice.

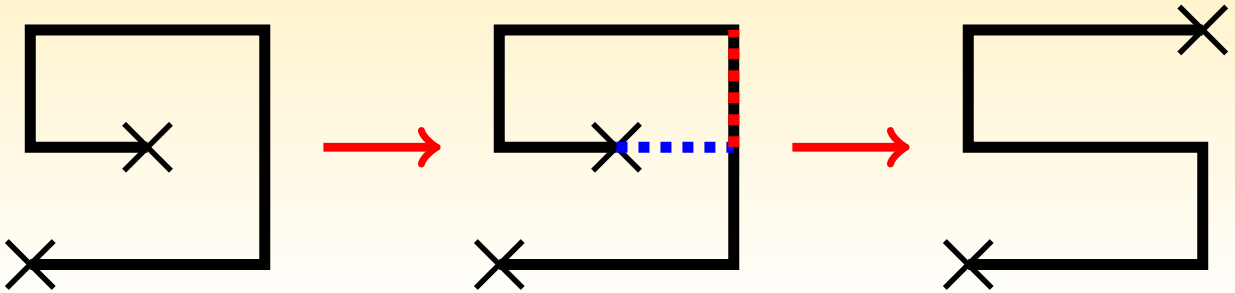
Backbite moves for sampling Hamiltonian paths.



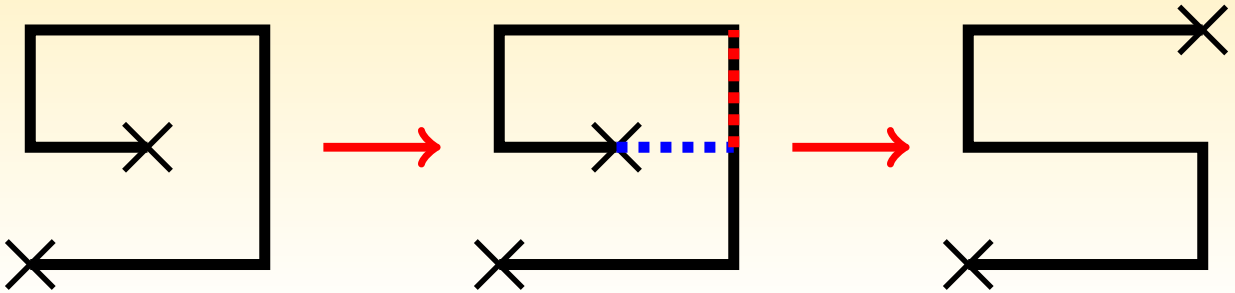
Backbite moves for sampling Hamiltonian paths.

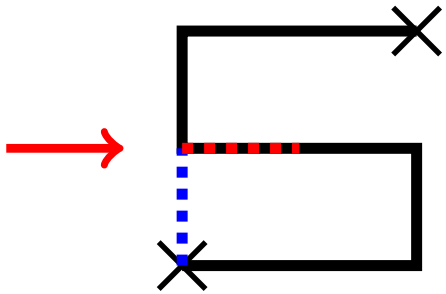


Backbite moves for sampling Hamiltonian paths.

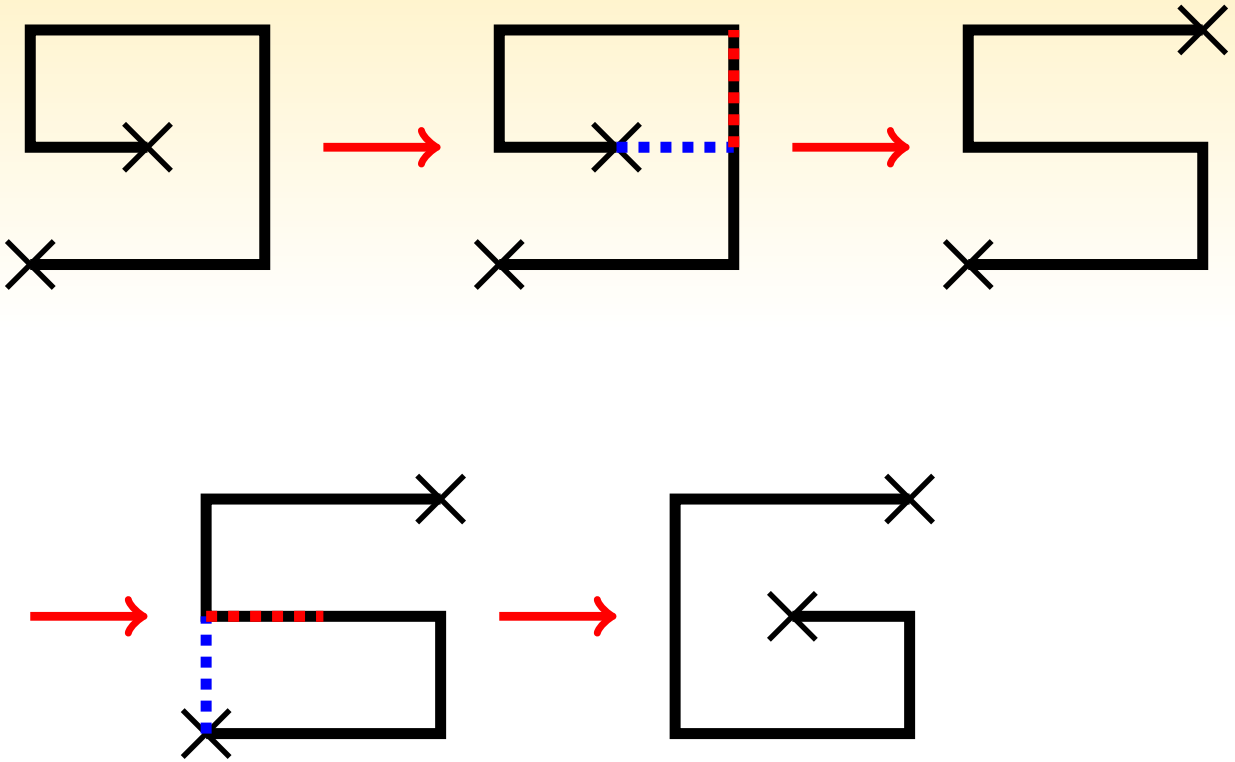


Backbite moves for sampling Hamiltonian paths.

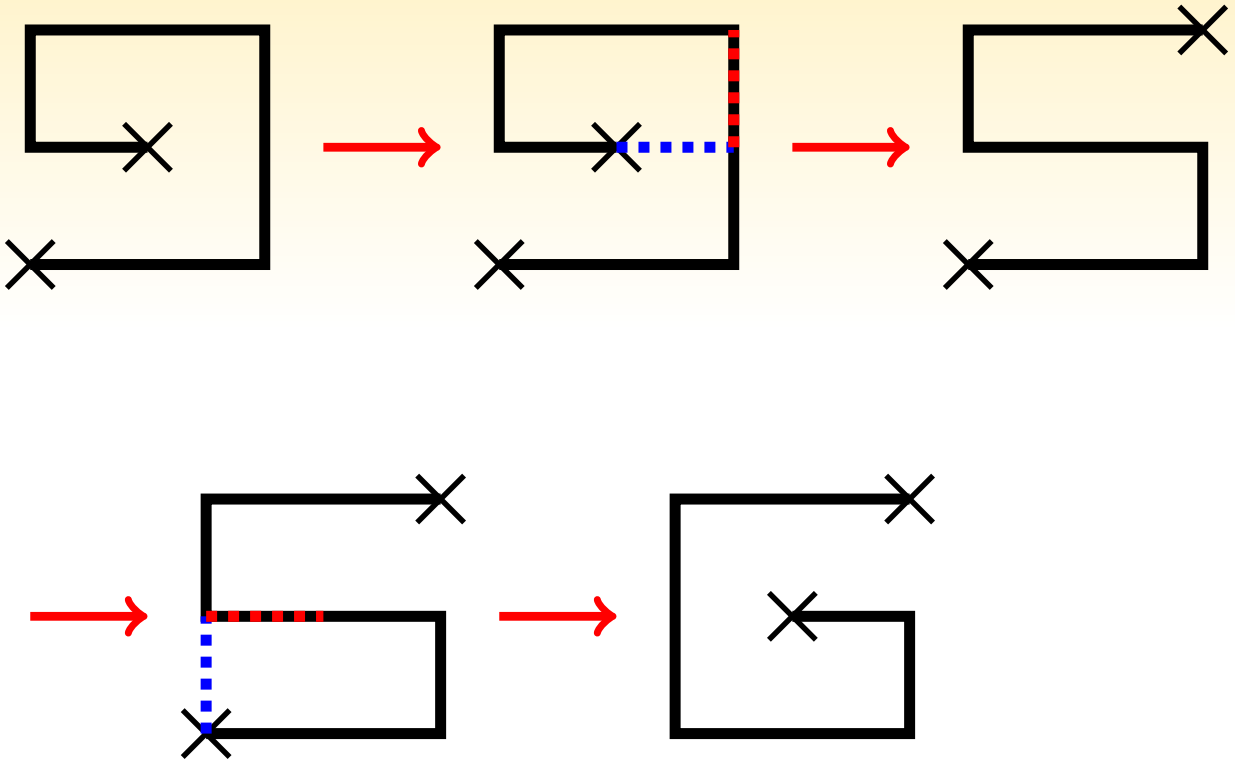




Backbite moves for sampling Hamiltonian paths.



Backbite moves for sampling Hamiltonian paths.



Each time we make a backbite move we create a loop, delete the edge which completes the loop, and *reverse* the orientation of the remaining edges of the loop.

Backbite move

- For the simple cubic lattice, loops are of mean size $O(N)$.

Backbite move

- For the simple cubic lattice, loops are of mean size $O(N)$.
- This suggests time $O(N)$ to perform a backbite move.

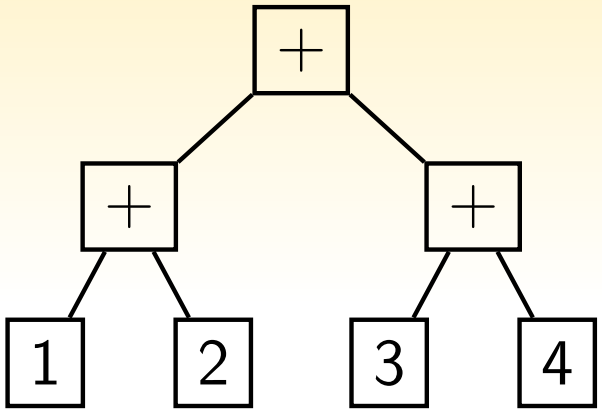
Backbite move

- For the simple cubic lattice, loops are of mean size $O(N)$.
- This suggests time $O(N)$ to perform a backbite move.
- However, can use binary tree data structure with *time reversal* as our symmetry operation.

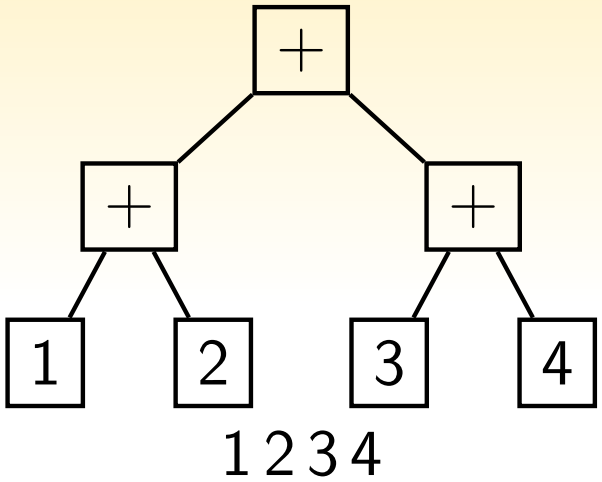
Backbite move

- For the simple cubic lattice, loops are of mean size $O(N)$.
- This suggests time $O(N)$ to perform a backbite move.
- However, can use binary tree data structure with *time reversal* as our symmetry operation.
- Only trick required is bookkeeping for determining neighbours.

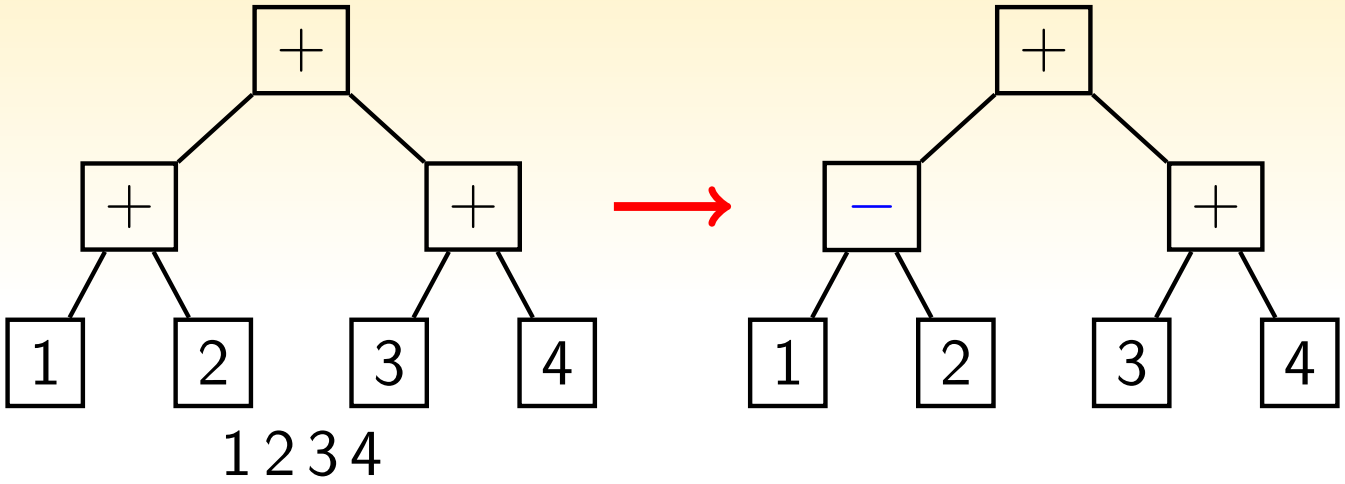
This time binary tree has time reversal symmetry elements in the nodes.



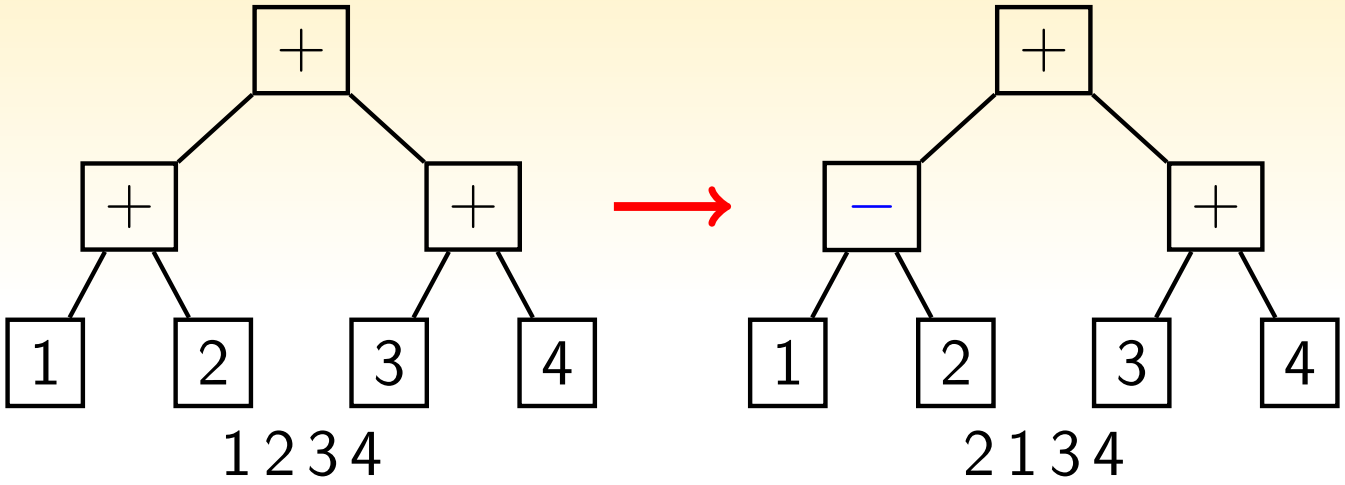
This time binary tree has time reversal symmetry elements in the nodes.



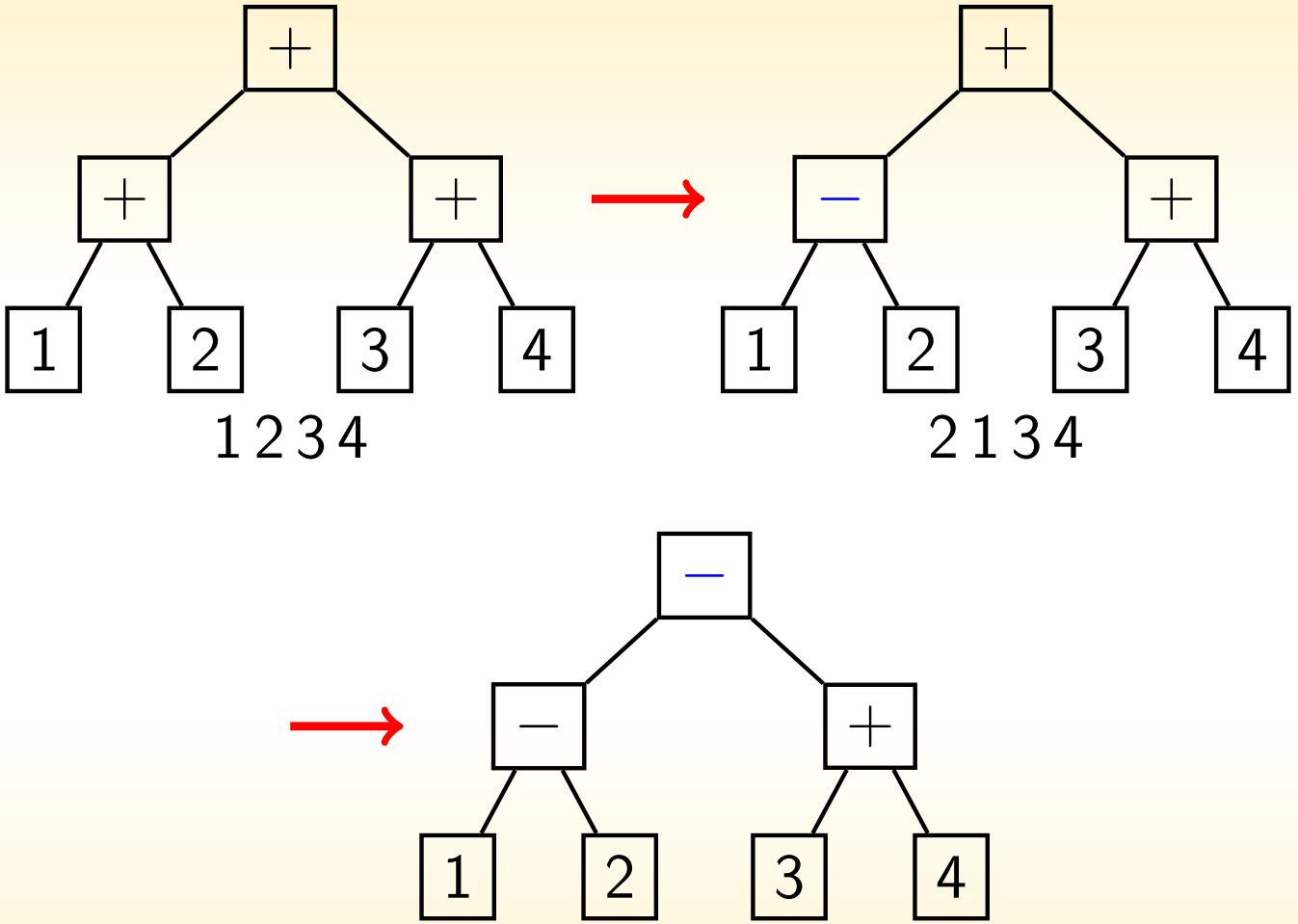
This time binary tree has time reversal symmetry elements in the nodes.



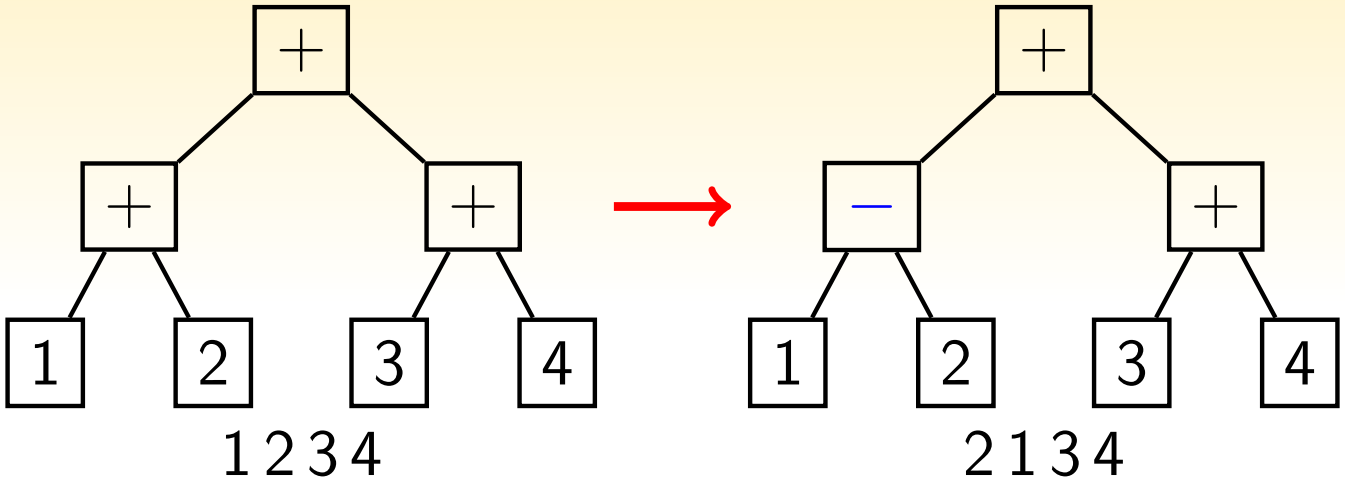
This time binary tree has time reversal symmetry elements in the nodes.

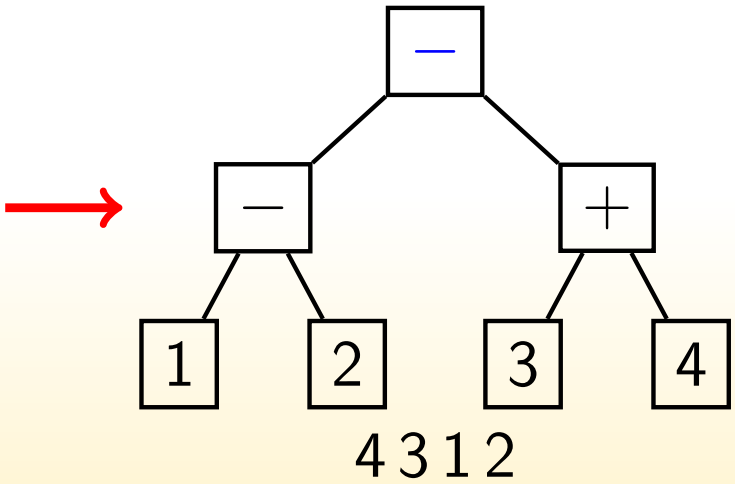


This time binary tree has time reversal symmetry elements in the nodes.

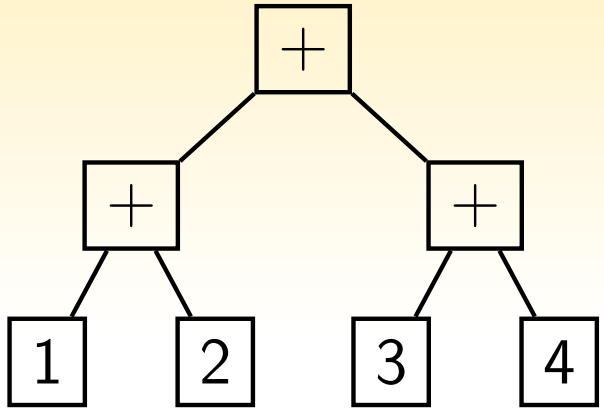


This time binary tree has time reversal symmetry elements in the nodes.

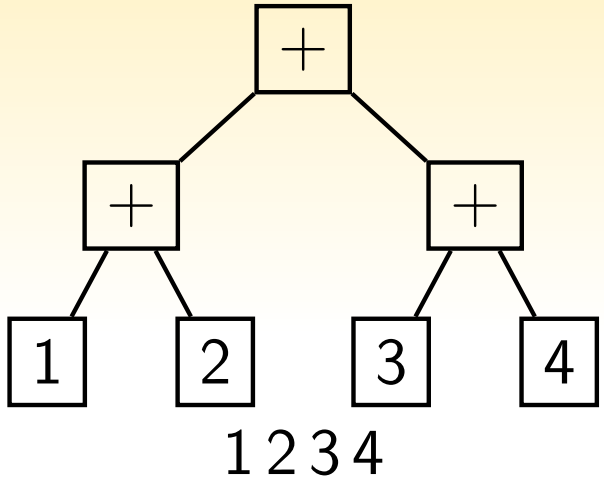




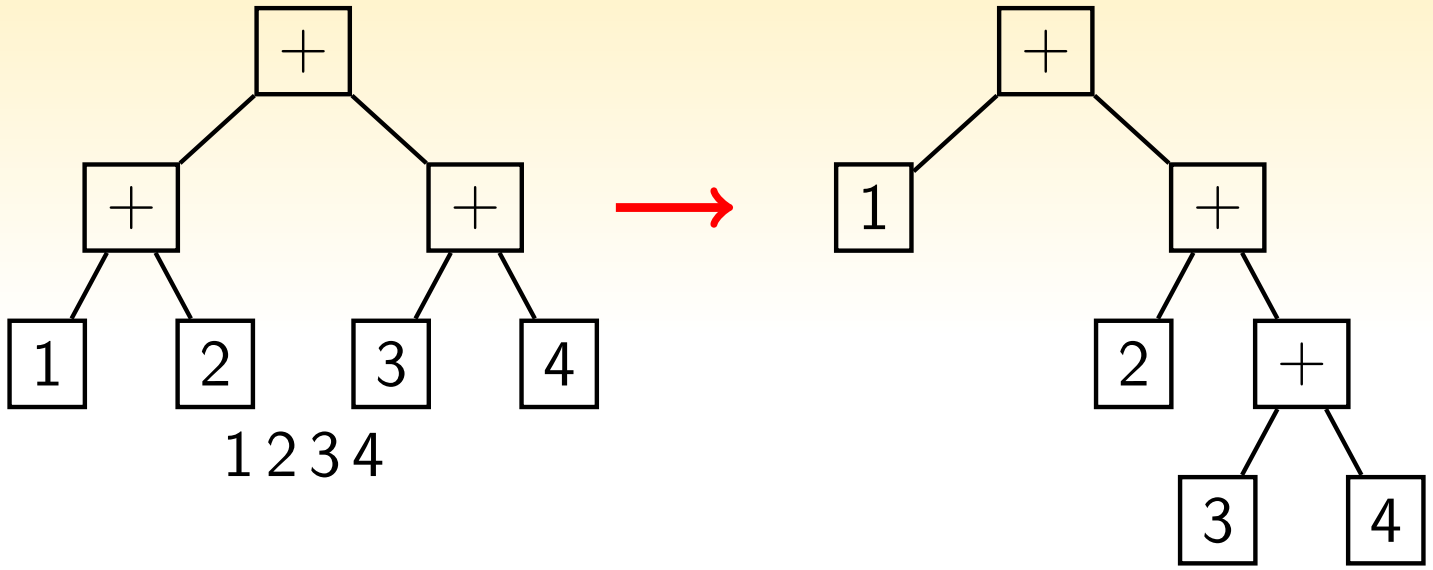
How do we reverse sequences of steps which don't align with the tree?



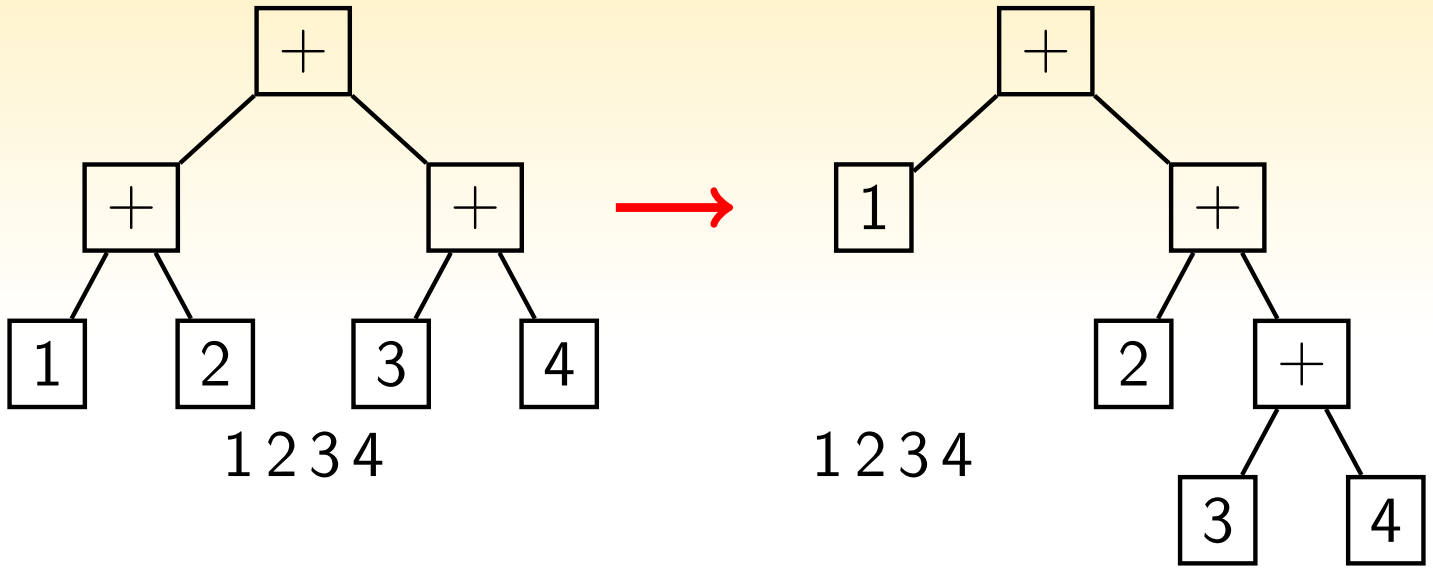
How do we reverse sequences of steps which don't align with the tree?



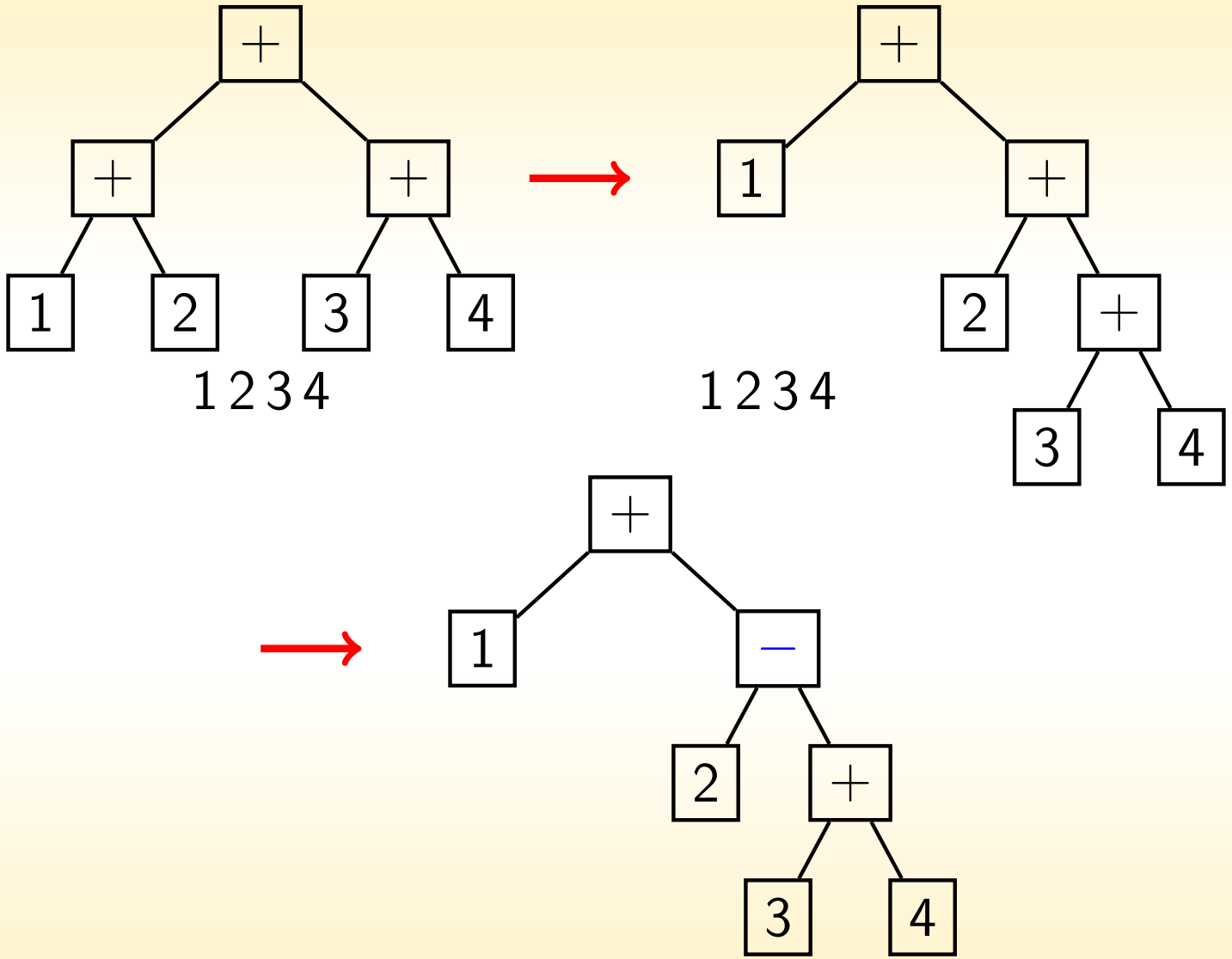
How do we reverse sequences of steps which don't align with the tree?



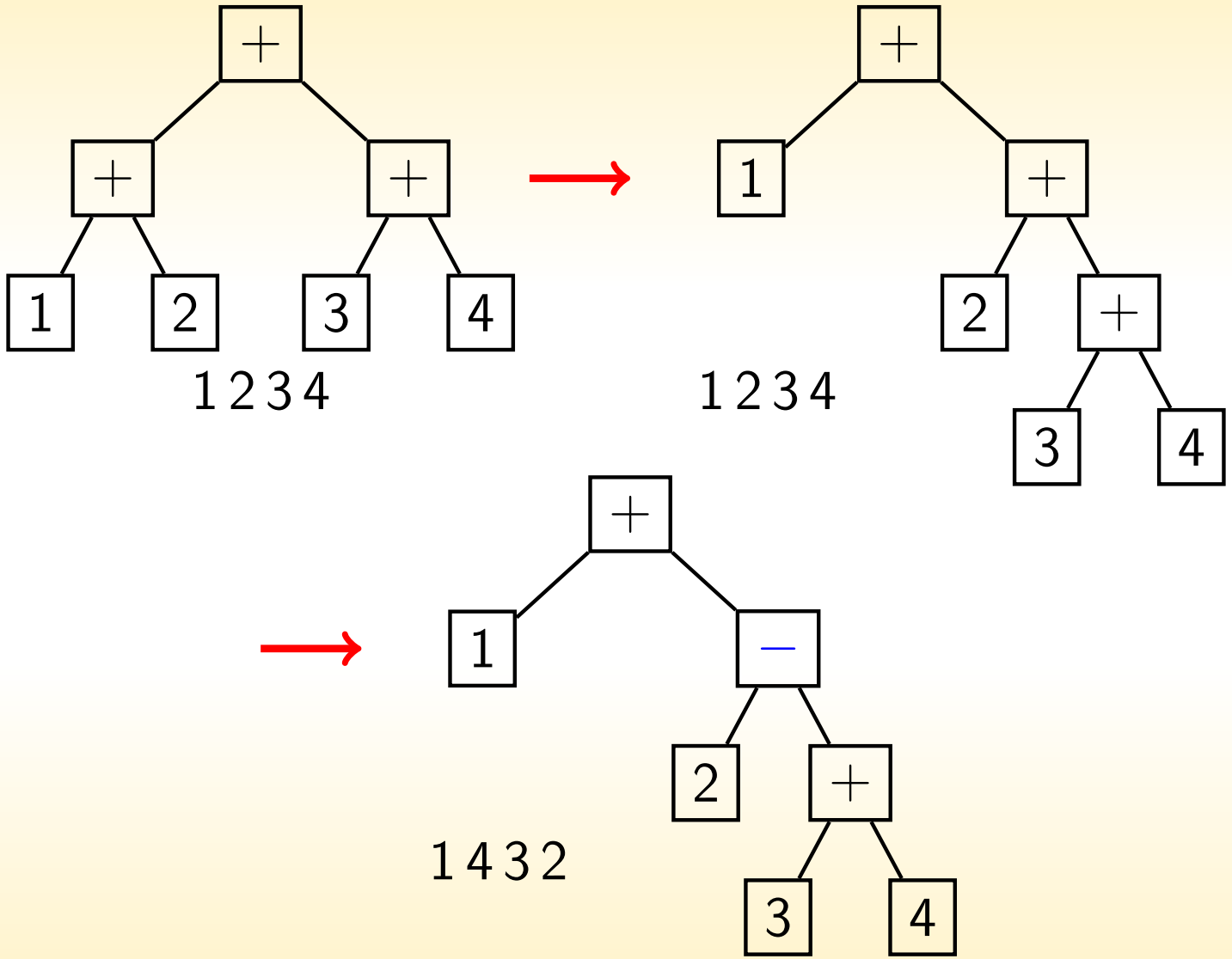
How do we reverse sequences of steps which don't align with the tree?



How do we reverse sequences of steps which don't align with the tree?



How do we reverse sequences of steps which don't align with the tree?



Results

- $\mathbb{Z}^2$:

Results

- $\mathbb{Z}^2$:
  - Moves have mean size $O(N^{\approx 0.75})$.

Results

- $\mathbb{Z}^2$:
  - Moves have mean size $O(N^{\approx 0.75})$.
  - $L = 8192$, $N = 67\,108\,864$: 360 000 steps, versus 6.77 tree rotations.

Results

- $\mathbb{Z}^2$:
  - Moves have mean size $O(N^{\approx 0.75})$.
  - $L = 8192$, $N = 67\,108\,864$: 360 000 steps, versus 6.77 tree rotations.
- $\mathbb{Z}^3$:

Results

- $\mathbb{Z}^2$:
  - Moves have mean size $O(N^{\approx 0.75})$.
  - $L = 8192$, $N = 67\,108\,864$: 360 000 steps, versus 6.77 tree rotations.
- $\mathbb{Z}^3$:
  - Moves have mean size $O(N)$.

Results

- $\mathbb{Z}^2$:
  - Moves have mean size $O(N^{\approx 0.75})$.
  - $L = 8192$, $N = 67\,108\,864$: 360 000 steps, versus 6.77 tree rotations.
- $\mathbb{Z}^3$:
  - Moves have mean size $O(N)$.
  - $L = 512$, $N = 134\,217\,728$: 46 million steps, versus 20.6 tree rotations.

Hydrodynamic radius

- Calculating R_{e}^2 easy, just keep track of ends. R_{g}^2 almost as easy.

Hydrodynamic radius

- Calculating R_{e}^2 easy, just keep track of ends. R_{g}^2 almost as easy.
- Why not R_{h}?

$$R_{\mathrm{h}}^{-1} = \frac{1}{N^2} \sum_{i \neq j} \frac{1}{r_{ij}}$$

$$\langle R_{\mathrm{h}} \rangle \sim D_{\mathrm{h}} N^{\nu} \left(1 + \frac{a}{N^{1-\nu}} + \frac{b}{N^{0.53}} + \cdots \right)$$

Hydrodynamic radius

- Calculating R_{e}^2 easy, just keep track of ends. R_{g}^2 almost as easy.
- Why not R_{h}?

$$R_{\mathrm{h}}^{-1} = \frac{1}{N^2} \sum_{i \neq j} \frac{1}{r_{ij}}$$

$$\langle R_{\mathrm{h}} \rangle \sim D_{\mathrm{h}} N^{\nu} \left(1 + \frac{a}{N^{1-\nu}} + \frac{b}{N^{0.53}} + \cdots \right)$$

  - Relevant to experiments.

Hydrodynamic radius

- Calculating R_{e}^2 easy, just keep track of ends. R_{g}^2 almost as easy.
- Why not R_{h}?

$$R_{\mathrm{h}}^{-1} = \frac{1}{N^2} \sum_{i \neq j} \frac{1}{r_{ij}}$$

$$\langle R_{\mathrm{h}} \rangle \sim D_{\mathrm{h}} N^{\nu} \left(1 + \frac{a}{N^{1-\nu}} + \frac{b}{N^{0.53}} + \cdots \right)$$

  - Relevant to experiments.
  - Strong corrections to scaling, large N should make a big difference.

Hydrodynamic radius

- Can't rapidly calculate R_{h}, as it depends non-linearly on all $O(N^2)$ interparticle distances, c.f. $O(\log N)$ to perform a pivot.

Hydrodynamic radius

- Can't rapidly calculate R_{h}, as it depends non-linearly on all $O(N^2)$ interparticle distances, c.f. $O(\log N)$ to perform a pivot.
- So estimate R_{h} instead!

Hydrodynamic radius

- Can't rapidly calculate R_{h}, as it depends non-linearly on all $O(N^2)$ interparticle distances, c.f. $O(\log N)$ to perform a pivot.
- So estimate R_{h} instead!
- MCMC: Time average(O) = Ensemble average(O).

Hydrodynamic radius

- Can't rapidly calculate R_{h}, as it depends non-linearly on all $O(N^2)$ interparticle distances, c.f. $O(\log N)$ to perform a pivot.
- So estimate R_{h} instead!
- MCMC: Time average(O) = Ensemble average(O).
- Key insight: for an unbiased estimator $E(O)$, Time average$(E(O))$ = Ensemble average(O).

Hydrodynamic radius

- Can't rapidly calculate R_{h}, as it depends non-linearly on all $O(N^2)$ interparticle distances, c.f. $O(\log N)$ to perform a pivot.
- So estimate R_{h} instead!
- MCMC: Time average(O) = Ensemble average(O).
- Key insight: for an unbiased estimator $E(O)$, Time average$(E(O))$ = Ensemble average(O).
- Our estimator is a weighted version of $1/r_{ij}$, monomers i and j chosen at random.

Hydrodynamic radius

- Can't rapidly calculate R_{h}, as it depends non-linearly on all $O(N^2)$ interparticle distances, c.f. $O(\log N)$ to perform a pivot.
- So estimate R_{h} instead!
- MCMC: Time average(O) = Ensemble average(O).
- Key insight: for an unbiased estimator $E(O)$, Time average$(E(O))$ = Ensemble average(O).
- Our estimator is a weighted version of $1/r_{ij}$, monomers i and j chosen at random.
- 1.591 ± 0.007, (Dünweg et al., 2002), vs 1.58040 ± 0.00002.

Counting SAW

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

$$c_N \sim A\ N^{\gamma-1}\mu^N\,[1 + \text{corrections}]$$

Counting SAW

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

$$c_N \sim A\ N^{\gamma-1}\mu^N\,[1+\text{corrections}]$$

- We wish to count c_N, and estimate μ.

Counting SAW

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

$$c_N \sim A\ N^{\gamma-1}\mu^N\,[1+\text{corrections}]$$

- We wish to count c_N, and estimate μ.
- Can estimate $c_N/(c_{N/2})^2$ via pivot algorithm - probability that when two SAW are concatenated the result is self-avoiding.

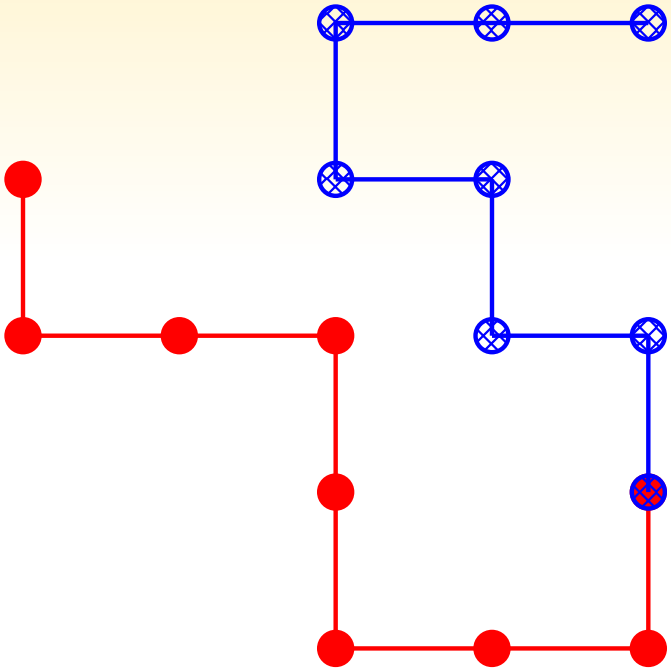
Counting SAW

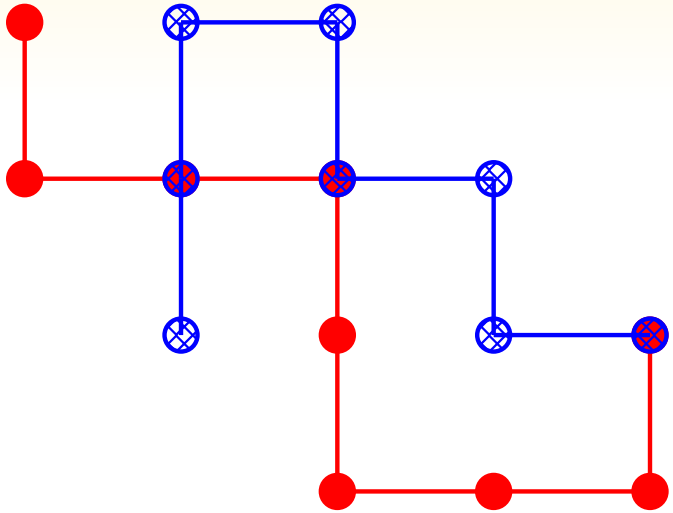
- Number of SAW of length N, c_N, tells us about how many conformations are available to SAW of a particular length:

$$c_N \sim A\ N^{\gamma-1}\mu^N\,[1+\text{corrections}]$$

- We wish to count c_N, and estimate μ.
- Can estimate $c_N/(c_{N/2})^2$ via pivot algorithm - probability that when two SAW are concatenated the result is self-avoiding.
- Then use

$$c_N = \frac{c_N}{c_{N/2}^2}\left(\frac{c_{N/2}}{c_{N/4}^2}\right)^2 \cdots c_k^{N/k}.$$





Counting SAW

- Sites of the walk which are close to the concatenation point have a much greater influence on the probability, than sites which are far away.

Counting SAW

- Sites of the walk which are close to the concatenation point have a much greater influence on the probability, than sites which are far away.
- Pivot sites chosen uniformly: $\widetilde{\tau}_{\mathrm{int}} = \Omega(N)$ (due to trapped configurations).

Counting SAW

- Sites of the walk which are close to the concatenation point have a much greater influence on the probability, than sites which are far away.
- Pivot sites chosen uniformly: $\widetilde{\tau}_{\mathrm{int}} = \Omega(N)$ (due to trapped configurations).
- Instead, choose distance from joint uniformly from all distance scales.

Counting SAW

- Sites of the walk which are close to the concatenation point have a much greater influence on the probability, than sites which are far away.
- Pivot sites chosen uniformly: $\widetilde{\tau}_{\mathrm{int}} = \Omega(N)$ (due to trapped configurations).
- Instead, choose distance from joint uniformly from all distance scales.
- $\Rightarrow \mu = 4.684\,039\,931(27)$ (C., 2013)

Counting SAW

- Sites of the walk which are close to the concatenation point have a much greater influence on the probability, than sites which are far away.
- Pivot sites chosen uniformly: $\widetilde{\tau}_{\mathrm{int}} = \Omega(N)$ (due to trapped configurations).
- Instead, choose distance from joint uniformly from all distance scales.
- $\Rightarrow \mu = 4.684\,039\,931(27)$ (C., 2013)
- This idea can be applied to other systems with additional length scales: confined polymers, bridges, star polymers, perhaps θ-polymers.

Conclusion

- Many opportunities to look for fast algorithms for polymer systems.

Conclusion

- Many opportunities to look for fast algorithms for polymer systems.
- Confined polymers, novel observables, dense polymers, θ-polymers, bridges, co-polymers, $\cdots$?

Conclusion

- Many opportunities to look for fast algorithms for polymer systems.
- Confined polymers, novel observables, dense polymers, θ-polymers, bridges, co-polymers, $\cdots$?
- Pivot algorithm only part of the story.