Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

Accelerating polymer simulation by means of tree data-structures and a parsimonious Metropolis algorithm[☆]

Stefan Schnabel[*], Wolfhard Janke

Institut für Theoretische Physik, Universität Leipzig, IPF 231101, 04081 Leipzig, Germany

ARTICLE INFO

Article history:
Received 20 March 2020
Received in revised form 15 May 2020
Accepted 25 May 2020
Available online 30 May 2020

Keywords:
Self-avoiding walk
Polymer
Monte Carlo
Metropolis
05.10.Ln
05.40.Fb

ABSTRACT

We show how a Monte Carlo method for generating self-avoiding walks on lattice geometries which employs a binary-tree data-structure can be adapted for hard-sphere polymers with continuous degrees of freedom. Data suggests that the time per Monte Carlo move scales logarithmically with polymer size. Next we generalize the method to Lennard-Jones polymers with untruncated monomer–monomer interaction. To this end we propose a variant of the Metropolis algorithm and demonstrate that in combination with the tree data-structure logarithmic scaling can be preserved. We further show how the replica-exchange method can be adapted for the same purpose.

1. Introduction

The simplest model for a polymer chain realizing nothing but its linear geometry is provided by a random walk, e.g., a random path on a periodic lattice. While these objects can be treated very easily with analytical methods, they do not possess an abundance of realistic features and real polymers behave similarly to random walks only at the Θ-point. If, as a step towards more realistic representations, excluded volume interaction is to be included, the simplest model is the self-avoiding walk. It is similar to the random walk, but any lattice site may only be visited once. Different parts of the polymer are not allowed to occupy the same space. Now the problem becomes more difficult and since in three dimensions the Flory exponent – signifying the scaling behavior of geometric quantities such as the end-to-end distance or the radius of gyration – as well as corrections to scaling are not exactly known, numerical methods are applied.

Ideally, in order to determine expectation values of observables of interest, one would like to sum over all possible walks and algorithms that can efficiently generate them are, therefore, strongly desired. While simple recursive methods on the simple-cubic lattice require days to generate all walks up to 20 steps, much more sophisticated techniques have been applied in order to enumerate walks of 36 steps [1]. Monte Carlo (MC) computer simulation methods introduce a statistical uncertainty, but are able to investigate much larger systems and until not too long ago $N \approx 10^5$ was the state of the art. Recently, however, Clisby introduced an ingenious technique [2–4] for the investigation of self-avoiding walks by means of a binary-tree representation simulating walks of length $N \approx 3 \times 10^7$. One of the technique's main features is a variable resolution; if possible large parts of the walk are treated as single units that are only 'zoomed-into' if the situation at hand demands it. So far this technique has only been applied to walks on lattices. In the first part of this study we describe how it can be adapted to investigate more realistic off-lattice hard-sphere polymers with continuous degrees of freedom.

To make the self-avoiding walk model for polymers more realistic the interaction with a solvent has to be included. This is often done implicitly by the introduction of an attractive term in the Hamiltonian. The transition from a bad solvent with the polymer in a collapsed, globular state towards a good solvent with configurations resembling swollen coils then corresponds to a change from low temperature where the interaction has a strong influence to high temperature where it is rather unimportant. On lattices, this attractive interaction can simply be a constant negative energy contribution for monomer–monomer contacts. Studies of such models, e.g., [5–7], have confirmed predictions about the general nature of the collapse transition. However, details like the form of corrections to scaling still remain unclear. While it is expected that the investigation of larger systems will provide further insight, it is also conceivable that off-lattice models with an underlying continuous and isotropic geometry might reach the asymptotic regime already for smaller sizes. There, the most commonly used potential is the 12-6 Lennard-Jones

[☆] The review of this paper was arranged by Prof. D.P. Landau.

[*] Corresponding author.

E-mail addresses: stefan.schnabel@itp.uni-leipzig.de (S. Schnabel), wolfhard.janke@itp.uni-leipzig.de (W. Janke).

https://doi.org/10.1016/j.cpc.2020.107414

potential, where both the repulsive and the attractive component are polynomial with exponents 12 and 6, respectively. Since this potential exerts forces at all distances the recalculation of the energy of the polymer after a modification of the configuration is of complexity $O(N^2)$. In order to achieve better performance the interaction is often truncated ensuring that beyond a certain distance the forces become zero. We show how this can be avoided by combining the tree-like data structure with a novel implementation of the Metropolis algorithm that does not rely on the exact knowledge of the change of energy.

In this paper we discuss in Section 2 the models under consideration. We then introduce a variant of the Metropolis algorithm in Section 3 which permits the simulation of large polymers with untruncated interactions and explore in Section 4 the mathematical properties of the transformations that are used to affect conformational updates. In Section 5 our version of Clisby's binary-tree technique is described, its power is demonstrated in Section 6 by an application to hard-sphere polymers. This is followed by the application of the generalized method to Lennard-Jones polymers both with untruncated and truncated interactions in Section 7. There we also briefly discuss how the replica-exchange method can benefit from the same ideas. Our conclusions are presented in Section 8. Finally, in the Appendix we describe a procedure for different types of conformational updates.

2. Models

2.1. Hard-sphere polymer

We first consider a freely-jointed chain with the excluded volume interaction modeled by hard spheres. This is the continuum model that most closely resembles the self-avoiding walk on a lattice.

If the chain is allowed free movement then the first monomer x_1 can be placed at an arbitrary point and positions of monomers $x_2, \ldots, x_N$ are given by

$$\mathbf{x}_k = \mathbf{x}_1 + \sum_{i=1}^{k-1} \mathbf{b}_i, \tag{1}$$

where the bond vectors have a constant length $|\mathbf{b}_i| = b$ and the distance between any two monomers has a lower bound to satisfy the constraint

$$|\mathbf{x}_k - \mathbf{x}_l| \geq D \cdot b. \tag{2}$$

Here, D is the diameter of the hard spheres measured in units of b. All values from $D = 0$ (the ideal chain) to $D = 1$ (tangent hard-sphere or pearl-necklace model) are possible.

2.2. Lennard-Jones polymer

For the second model we abandon the hard spheres in favor of a 12-6 Lennard-Jones interaction

$$U(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] \tag{3}$$

acting between all pairs of monomers. The attractive force models implicit hydrophobic interactions such that low (high) temperatures with dense (extended) configurations correspond to bad (good) solvent. For the simulations presented in this paper we chose the bead diameter $\sigma = b/2^{1/6}$ such that $U(r)$ is minimal for $r_{\min} = b$. The complete Hamiltonian reads

$$\mathcal{H} = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} U(\mathbf{x}_i - \mathbf{x}_j) \tag{4}$$

and its calculation requires[1] $(N-1)(N-2)/2$ evaluations of U. Therefore, $\mathcal{H}$ is of quadratic complexity: $\mathcal{H} \in \mathcal{O}(N^2)$. Using standard techniques this limits the size of the systems that can be treated. For instance, the most recent study we found [8] reached $N = 217$. Therefore, such models are rarely investigated. Instead, the problem is typically avoided by a shift towards slightly different models. These use truncated potentials that do not create forces beyond a certain cut-off distance r_c which typically equals a few multiples of σ. However, as we will show it is possible to achieve fast simulation with the full potential.

Note that for both models we have N monomers connected by $N-1$ bonds as opposed to the notation of N steps and $N+1$ occupied sites that is often used in the context of random walks.

3. A parsimonious Metropolis algorithm

Since the dawn of the information age the Metropolis algorithm [9] has been the workhorse of computational statistical physics. Detailed balance,

$$P(\mu)W(\mu,\nu) = P(\nu)W(\nu,\mu), \tag{5}$$

provides a solution to the Master equation. Here, μ, ν are states of the system, P occupation probabilities, and $W(\mu,\nu)$ is the probability to occupy ν in the next step if μ is occupied now. To be precise, the probability for a transition between two states in a Monte Carlo simulation is often the product of the probability of such an update being proposed and of it being accepted. Usually the probabilities for proposals are symmetric such that

$$P(\mu)W_{\text{accept}}(\mu,\nu) = P(\nu)W_{\text{accept}}(\nu,\mu), \tag{6}$$

which in turn is solved by

$$W_{\text{accept}}(\mu,\nu) = \min\left(1, \frac{P(\nu)}{P(\mu)}\right). \tag{7}$$

If a canonical distribution at the inverse temperature $\beta = (k_{\mathrm{B}}T)^{-1}$ with $P(\mu) \propto \exp(-\beta E(\mu))$ is aimed at, this becomes

$$W_{\text{accept}}(\mu,\nu) = \min\left(1, e^{-\beta \Delta E}\right), \tag{8}$$

with $\Delta E = E(\nu) - E(\mu)$. The usual procedure is to calculate ΔE, determine this probability, and to draw a uniformly distributed random number $\xi \in [0, 1)$. The update is accepted if $\xi < W_{\text{accept}}(\mu,\nu)$. We intend to reverse this sequence. The last condition is equivalent to

$$\xi < e^{-\beta \Delta E} \tag{9}$$

and consequently to

$$-\frac{\ln \xi}{\beta} > \Delta E, \tag{10}$$

assuming that $\beta > 0$. Hence, it is possible to draw ξ first and then to estimate ΔE with increasing precision until it can be decided whether (10) is fulfilled or violated. In cases where such an estimate can be done with less computational effort than a complete calculation, the simulation should run faster than with the standard technique, while with both methods the same updates are accepted or rejected and the trajectories through the configuration space are, therefore, identical.

Albeit being potentially extremely beneficial, such an approach has – at least to our knowledge – not been pursued in the past. This may be due to two reasons. On the one hand it is only relevant for Hamiltonians with long-range interactions which severely limits the number of candidate systems. On the

[1] Since the bond length is fixed, contributions of adjacent monomers are constant.

other hand, the state of the system must be represented in the computer in such a way that estimates of the interaction of large parts of the system can be done very fast which requires the use of suitable data structures.

4. Monte Carlo update moves as transformations

During the simulation we modify the polymer configuration using two well established update moves. For the hard-sphere polymer the pivot move [10] is sufficient while the simulation of the Lennard-Jones polymer greatly benefits from the application of a single-bond-rotation update [11]. As long as one is not interested in dense configurations, more sophisticated strategies are not needed [12]. For both moves we choose a $k \in \{1, \ldots, N-1\}$ and a random axis through $\mathbf{x}_k$ perpendicular to $\mathbf{b}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$. The rotation angle is drawn such that as a result the new position $\mathbf{x}'_{k+1}$ is at a random position on the sphere of radius $b = |\mathbf{b_k}|$ around $\mathbf{x}_k$. For the pivot update we apply the rotation that is given by this axis and the angle to all monomers $\mathbf{x}_{k+1}, \ldots, \mathbf{x}_N$. For the single-bond-rotation $\mathbf{x}_{k+1}$ is modified in the same way, but all bonds $\mathbf{b}_i$ with $i \neq k$ are kept unchanged such that $\mathbf{x}'_i|_{i>k+1} = \mathbf{x}_i + \mathbf{x}'_{k+1} - \mathbf{x}_{k+1}$. In both cases $N-k$ monomer positions are altered such that $k(N-k)-1$ monomer–monomer distances change (while the distance between $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ remains b). Therefore, basic implementations that recalculate all new distances are of complexity $O(N^2)$. More advanced techniques that search only in the neighborhood of the monomers and that, therefore, only work for short-range interactions are able to reduce this to $O(N)$.

In general the rotation of a monomer position $\mathbf{x}$ around an axis that passes through the point $\mathbf{x}_p$ is given by

$$\mathbf{x}' = \mathcal{R}(\mathbf{x} - \mathbf{x}_p) + \mathbf{x}_p = \mathcal{R}\mathbf{x} - \mathcal{R}\mathbf{x}_p + \mathbf{x}_p, \tag{11}$$

where $\mathcal{R}$ is the rotation matrix. Hence

$$\mathbf{x}' = \mathcal{R}\mathbf{x} + \mathbf{c}, \tag{12}$$

with

$$\mathbf{c} = \mathbf{x}_p - \mathcal{R}\mathbf{x}_p. \tag{13}$$

It is easy to see that reflections on arbitrary planes as well as simple shifts by some vector can also be expressed in this form.

Applying two transformations consecutively,

$$\begin{aligned} \mathbf{x}'' &= \mathcal{R}_2\mathbf{x}' + \mathbf{c}_2, & (14)\\ &= \mathcal{R}_2\left(\mathcal{R}_1\mathbf{x} + \mathbf{c}_1\right) + \mathbf{c}_2, & (15)\\ &= \left(\mathcal{R}_2\mathcal{R}_1\right)\mathbf{x} + \left(\mathcal{R}_2\mathbf{c}_1 + \mathbf{c}_2\right), & (16) \end{aligned}$$

we obtain the same structure as in (12) which means that it is easily possible to 'multiply' two transformations before applying them to the monomer:

$$\begin{aligned} &\mathcal{T}_1 \equiv \{\mathcal{R}_1, \mathbf{c}_1\}, \quad \mathcal{T}_2 \equiv \{\mathcal{R}_2, \mathbf{c}_2\}, & (17)\\ &\mathcal{T}_1 \bullet \mathbf{x} := \mathcal{R}_1\mathbf{x} + \mathbf{c}_1, & (18)\\ &\mathcal{T}_2 \bullet (\mathcal{T}_1 \bullet \mathbf{x}) = (\mathcal{T}_2 \circ \mathcal{T}_1) \bullet \mathbf{x}, & (19) \end{aligned}$$

with

$$\mathcal{T}_2 \circ \mathcal{T}_1 := \{\mathcal{R}_2\mathcal{R}_1, \mathcal{R}_2\mathbf{c}_1 + \mathbf{c}_2\}. \tag{20}$$

It is also easy to see that this operation is associative

$$(\mathcal{T}_3 \circ \mathcal{T}_2) \circ \mathcal{T}_1 = \mathcal{T}_3 \circ (\mathcal{T}_2 \circ \mathcal{T}_1). \tag{21}$$

For the notation used in this paper rotations are realized as matrices. However, we found that an implementation using quaternions is faster. For large chains ($N \geq 2^{16}$) for which the overhead from other aspects of the algorithm is negligible that speed-up is about 20% independently of D.

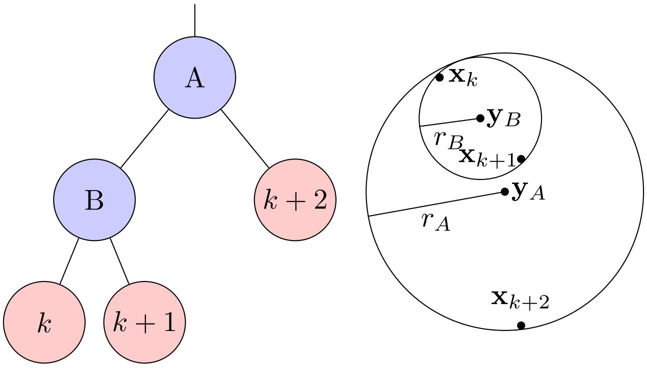

Fig. 1. Left: A sub-tree with three monomers represented by the leaf-nodes k, $k+1$, and $k+2$ and two internal nodes. Right: The contained geometric information: Monomer positions $\mathbf{x}_k$, $\mathbf{x}_{k+1}$, $\mathbf{x}_{k+2}$ and sphere parameters $\mathbf{y}_A$, r_A and $\mathbf{y}_B$, r_B.

5. Polymer configurations as binary trees

A few years ago Clisby [2,3] has introduced the binary tree as a fundamental data structure for the simulation of self-avoiding walks on lattices and has achieved most impressive results. Strongly inspired by his ground-breaking work we adapt this approach for off-lattice polymers.

All data is organized in a binary tree where the leaves represent individual monomers and any internal node, i.e., each node that is not a leaf, provides a coarse-grained representation of its children: Each node contains among other data the parameters for a sphere that comprises all monomers in the sub-tree to which it is root (Fig. 1). Such a representation serves two purposes. On the one hand it allows one to ensure that distinct parts of the polymer represented by different nodes do not overlap, since a sufficient (although not necessary) condition is that the respective spheres do not intersect. On the other hand modifications to the polymer can be applied at a level of low resolution to nodes high in the tree while the application to the more numerous smaller nodes below is postponed. The goal is to aggregate multiple modifications at high levels such that their later applications at low levels can be done collectively in a more efficient manner. To that end each internal node is able to store a transformation of the shape defined in (17) that applies to all nodes in its sub-tree, i.e., the nodes constituting its collective offspring, except the node itself. Consider for instance the tree in Fig. 2(a). Although the monomer position stored in the lower left node is $\mathbf{x}_1$, the actual position of the first monomer is given by $\mathcal{T}_A \bullet (\mathcal{T}_B \bullet \mathbf{x}_1)$, and the position of the center of sphere that belongs to node B is actually $\mathcal{T}_A \bullet \mathbf{y}_B$. Only when a need to access a certain position in a node arises the respective transformations in the 'ancestor'-nodes are applied. This is either done separately outside the tree or by pushing down transformations as depicted in Fig. 2(b,c).

Let us summarize which data has to be stored in a single node:

- links relevant for the geometry of the tree, i.e., links to the parent-node and the two children,
- parameters for a sphere that contains all monomers in the sub-tree to which the node is root,
- the data for a transformation that is to be applied to all nodes in the node's sub-tree, but not to the node itself,
- additional information, e.g., index of the node or size of the sub-tree.

Since the underlying data structure is a binary tree it is natural although not required to chose system sizes that are powers of two.

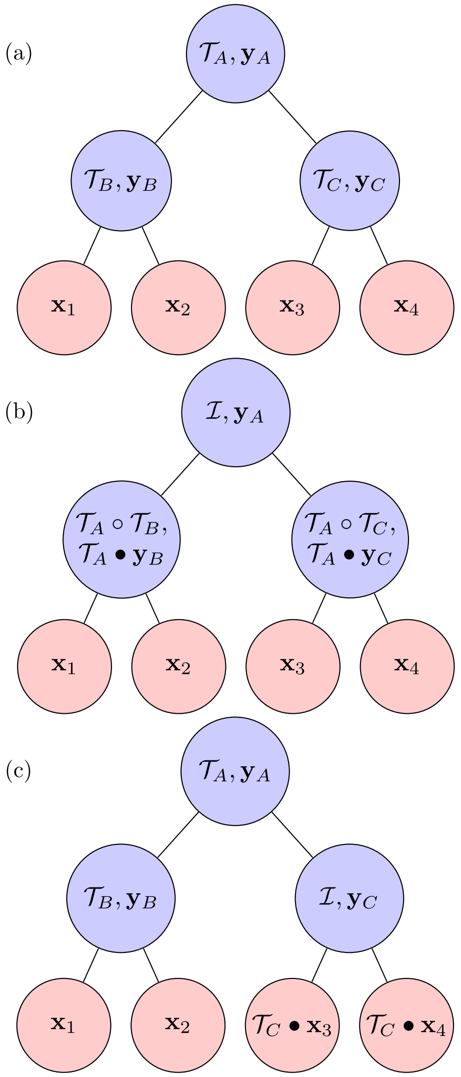
(a)
$\mathcal{T}_A, \mathbf{y}_A$
$\mathcal{T}_B, \mathbf{y}_B$
$\mathcal{T}_C, \mathbf{y}_C$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\mathbf{x}_4$
(b)
$\mathcal{I}, \mathbf{y}_A$
$\mathcal{T}_A \circ \mathcal{T}_B, \mathcal{T}_A \bullet \mathbf{y}_B$
$\mathcal{T}_A \circ \mathcal{T}_C, \mathcal{T}_A \bullet \mathbf{y}_C$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\mathbf{x}_4$
(c)
$\mathcal{T}_A, \mathbf{y}_A$
$\mathcal{T}_B, \mathbf{y}_B$
$\mathcal{I}, \mathbf{y}_C$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathcal{T}_C \bullet \mathbf{x}_3$
$\mathcal{T}_C \bullet \mathbf{x}_4$

Fig. 2. (a) A binary tree for a polymer with four monomers with transformations in the internal nodes. (b) The tree in (a) with the transformation $\mathcal{T}_A$ pushed down. (c) The tree in (a) with the transformation $\mathcal{T}_C$ pushed down. Here, $\mathcal{I}$ stands for the identity or absence of a transformation.

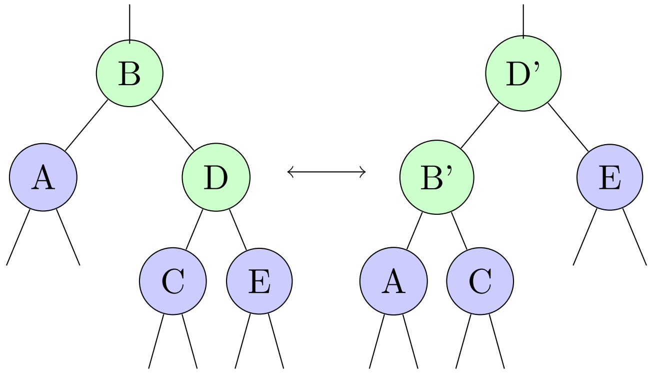

Fig. 3. Tree-rotations are used to move nodes up or down. Only the nodes B and D are affected.

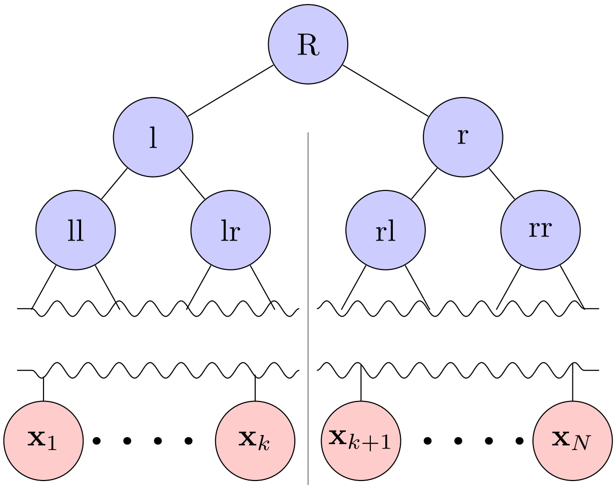

Fig. 4. The binary tree during an update.

6. Simulating hard-sphere polymers

In order to perform an update of the polymer it is convenient to rearrange the tree. Consider Fig. 3, where versions of a section of a tree are displayed. Transitions between them are called tree-rotations. With respect to the parts of the polymer they contain, during these operations only the nodes B and D are altered: Node B has the children A and D while node B′ links to A and C, while the children of D are C and E as opposed to B′ and E for D′. This means that during such an operation one has to recalculate the spheres in the nodes B′ and D′ if going from left-to-right in Fig. 3 or in D and B when moving right-to-left. First, however, it is important to take care of potentially stored transformations. The easiest way to do this is to push down the transformations in B and D (B′ and D′) such that both nodes do not hold a transformation when the actual tree-rotation is performed. Note that the horizontal order of the nodes is not affected. From left-to-right the nodes read A,B,C,D,E or A,B′,C,D′,E. Furthermore, each internal node, i.e., each node that does not represent a single monomer, is in such a horizontal order always placed between the same two leaves (monomers) while there is exactly one internal node between any two adjacent leaves. This is exploited when an update is to be performed. Assume that the following update of the polymer configuration is proposed[2]

$$\mathbf{x}'_i = \begin{cases} \mathbf{x}_i & \text{if} \quad i \le k \\ \mathcal{T}_U \bullet \mathbf{x}_i & \text{else.} \end{cases} \tag{22}$$

One can then identify the internal node that is (in horizontal order) positioned between the leaves corresponding to $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ and use tree-rotations to move up this node until it becomes the root-node, i.e., the node on top without a parent. Since the horizontal order is preserved and the root node is between $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$, it is clear that the leaves $\mathbf{x}_1, \ldots, \mathbf{x}_k$ are now in the left part, i.e., the sub-tree to which the left child of the root-node is root, and the leaves $\mathbf{x}_{k+1}, \ldots, \mathbf{x}_N$ are in the right part. This situation is depicted in Fig. 4. Once the tree is in this shape, one can test whether the original left part overlaps with the transformed right part. If the spheres of the children of the root node do not overlap, which is the case if the distance between their midpoints is larger than the sum of the radii,

$$|\mathbf{y}_l - \mathcal{T}_U \bullet \mathbf{y}_r| > r_l + r_r, \tag{23}$$

then there can be no overlap of any two individual monomers. Otherwise the resolution on one side has to be increased by

[2] In practice it is, of course, more efficient to modify the left part ($i = 1, \ldots, k-1$) if $k < N/2$. This is part of our implementation, but omitted here in favor of a simpler description.

stepping down one level in the tree. One has to either test that

$$\begin{aligned} &|\mathbf{y}_{ll} - \mathcal{T}_U \bullet \mathbf{y}_r| > r_{ll} + r_r \\ \text{and} \quad &|\mathbf{y}_{lr} - \mathcal{T}_U \bullet \mathbf{y}_r| > r_{lr} + r_r \end{aligned} \tag{24}$$

or that

$$\begin{aligned} &|\mathbf{y}_{l} - \mathcal{T}_U \bullet \mathbf{y}_{rl}| > r_{l} + r_{rl} \\ \text{and} \quad &|\mathbf{y}_{l} - \mathcal{T}_U \bullet \mathbf{y}_{rr}| > r_{l} + r_{rr}. \end{aligned} \tag{25}$$

Beforehand, either the transformation $\mathcal{T}_l$ or $\mathcal{T}_r$ has to be pushed down such that the actual positions $\mathbf{y}_{ll}, \mathbf{y}_{lr}$ or $\mathbf{y}_{rl}, \mathbf{y}_{rr}$ are used. This process is continued iteratively; whenever an individual inequality is violated, it has to be replaced by two conditions that are derived by splitting one of the participating nodes. Intuitively, one should split the larger one. For this model, it seems that splitting the node which contains more monomers is slightly more efficient ($\approx$ 1%) than splitting the node with the larger radius. The update is rejected if the process reaches a point where two nodes that are leaves, i.e., monomers, overlap. Otherwise, when the process terminates with all remaining inequalities fulfilled, the update is accepted and the transformation $\mathcal{T}_U$ is stored in the node r:

$$\mathcal{T}_r' = \mathcal{T}_U, \tag{26}$$

or is multiplied to the existing transformation, if this node still contains one:

$$\mathcal{T}_r' = \mathcal{T}_U \circ \mathcal{T}_r. \tag{27}$$

Finally, we retrace our steps and use the inverse tree-rotations as before, in order to rebalance the binary tree.

There are some differences to Clisby's technique beyond the mere transition from lattice to continuum. In particular, in our case the transformation in a given node applies to all nodes in the sub-tree to which it is root while in Ref. [3] transitions apply only to the sub-tree to which the right child of the node containing the transformation is root. This means that in our case for every monomer there are potentially $\log_2 N$ transformations that need to be applied, while for the original version this number is smaller on average, e.g., there are no transformations that apply to the first monomer $\mathbf{x}_1$ in any case. On the other hand, since in our version we push down transformations, about half of all nodes do not hold transformations at all such that the number that actually applies is smaller then the maximal value. In order to facilitate the tree-rotations we push down the transformations so that the relevant nodes are empty. This is not easily done in Clisby's version. Instead new transformations that keep the polymer configurations unchanged are determined which – at least in one direction – requires the inversion of one of the transformations. Since this is more complicated in continuum than on the lattice we chose this modification. On the other hand, with the original strategy it is in principle possible to omit the nodes containing individual monomers which would reduce the required memory by half.[3] Achieving a similar improvement with our method would only be possible by using structurally different nodes that do not store transformations, radii, or sizes for leaves, which would render the code a bit more complicated. We have not compared both methods and do not claim that one performs better than the other.

Testing our implementation for $D = 0.5$ with the pivot update we find that the desired efficiency is indeed achieved and that simulations with sizes up to $N \approx 10^7$ are possible. In Fig. 5 we show the mean time $t(N)$ that is required for a single pivot update. Although we cannot conclusively decide how this function behaves in the thermodynamic limit, it seems plausible that its derivative (inset of Fig. 5) for large N becomes proportional to N^{-1} which would imply that $t(N)$ scales like $\log N$. Updates are accepted with satisfying probability which is, however, decreasing with system size. While 80% of all updates are accepted for $N \approx 10$ this decreases to 20% for $N = 2^{23} \approx 10^7$. We find that the rate of acceptance for pivot updates attempted at the center of the polymer is reasonably well described by N^κ with $\kappa \approx -0.09$.

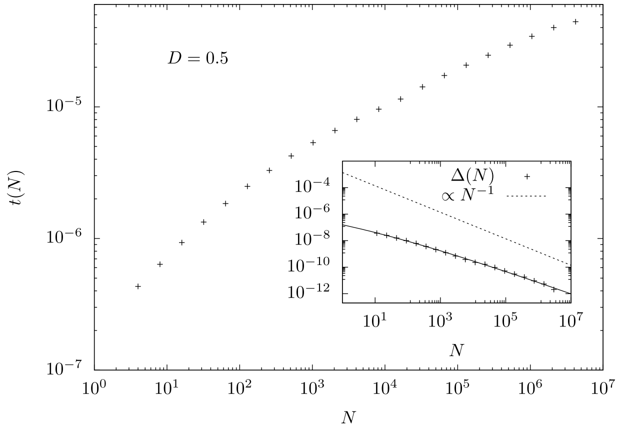

Fig. 5. The average time t (in seconds) required for a single pivot update for a hard-sphere polymer as a function of N. Inset: An estimate for the derivative $dt/dN \approx \delta(N) = \left[t(N\sqrt{2}) - t(N/\sqrt{2})\right]/\left(N/\sqrt{2}\right)$ with a fitted function of the form $\Delta(N) = \exp[c_1 - \ln N + c_2/(\ln N - c_3)]$ as guide for the eye and the suggested asymptotic relation e^{c_1}/N.

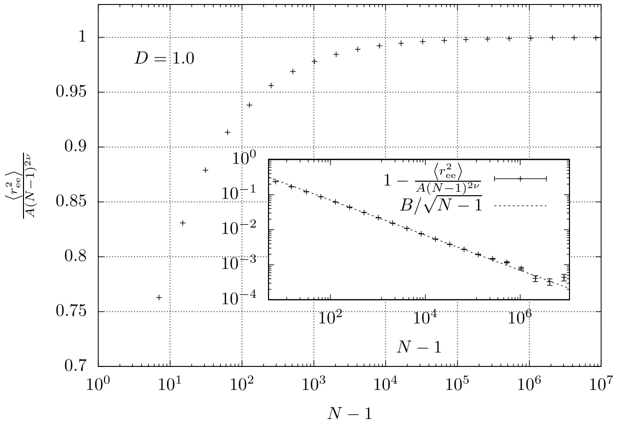

Fig. 6. Scaling of the average squared end-to-end distance. Inset: Over several orders of magnitude in length $N-1$ the first correction to scaling is very well described by a single term with exponent -0.5. The error bars in the inset are obtained with the Jackknife method. They are too small to be depicted in the main frame.

We performed simulations for $N = 2^3, \ldots, 2^{23}$ starting with a straight configuration. The first $64N$ Monte Carlo time steps were dedicated to the equilibration of the system. Afterwards, the number of steps depended on the system size. For instance, we performed $2 \times 10^9 N$ steps for $N \le 2^9$, $4 \times 10^8 N$ steps for $N = 2^{10}$, $4 \times 10^6 N$ steps for $N = 2^{15}$, $3 \times 10^4 N$ steps for $N = 2^{20}$, and $2500N$ steps for $N = 2^{23}$. Thereby, one time step consists of one pivot and one bond-rotation update. Statistical errors were estimated using the Jackknife method. When looking at the data shown in Fig. 6, we find that results agree with expectations. We acknowledge that analyzing the behavior of quantities like the squared end-to-end distance $\langle r_{\rm ee}^2 \rangle$ as a function of the number of bonds (or steps of a walk) $N-1$ is an intricate business and that the exponent of the first correction is not known. We will revisit it in detail in the future. Here, we are content with noticing

[3] N. Clisby, private communication.

that in the case $D = 1.0$ using the latest and most precise value for the Flory exponent available from a recent lattice study [4], $\nu = 0.5875970(4)$, we find that

$$\left\langle r_{\mathrm{ee}}^2 \right\rangle = A(N-1)^{2\nu}\left(1 - \frac{B}{\sqrt{N-1}}\right), \tag{28}$$

with $A = 1.77254$ and $B = 0.701$ obtained from a fit for $N > 10^3$, provides a very good description of the scaling. This demonstrates very convincingly the expected universality of lattice and continuum self-avoiding walks.

7. Simulating Lennard-Jones polymers

7.1. Untruncated interactions

For applying this idea to polymers with Lennard-Jones interaction the binary tree as used for the hard-sphere polymer again provides a well-suited data structure. Using trees in order to hierarchically estimate the interactions of N-body problems is not a new approach. Similar strategies have been used to simulate the (approximate) dynamics of gravitational systems as early as 1986 [13]. In these studies a more or less homogeneous three-dimensional system is separated into cubic cells which are organized in an octree, where each internal node has eight children. The linear nature of the polymer simplifies the situation considerably. Again, distinct groups of monomers $A = \{\mathbf{x}_k, \ldots, \mathbf{x}_{k+s_A-1}\}$ and $B = \{\mathbf{x}_l, \ldots, \mathbf{x}_{l+s_B-1}\}$, with $k \geq 1, l \geq k + s_A, N \geq l + s_B - 1$, are represented by spheres

$$\begin{aligned} |\mathbf{x}_i - \mathbf{y}_A| &< r_A, \quad i = k, \ldots, k + s_A - 1, \\ |\mathbf{x}_j - \mathbf{y}_B| &< r_B, \quad j = l, \ldots, l + s_B - 1. \end{aligned} \tag{29}$$

The interaction between two such groups can be estimated if the distance between the spheres exceeds the minimum position of the Lennard-Jones potential

$$|\mathbf{y}_A - \mathbf{y}_B| - r_A - r_B > r_{\min} = 2^{1/6}\sigma. \tag{30}$$

Due to the monotonicity of the interaction $U(r)$ for $r > r_{\min}$ the energy

$$E_{AB} = \sum_{i=k}^{k+s_A-1} \sum_{j=l}^{l+s_B-1} U(|\mathbf{x}_i - \mathbf{x}_j|) \tag{31}$$

has a lower bound

$$E_{AB} \geq s_A s_B U(|\mathbf{y}_A - \mathbf{y}_B| - r_A - r_B) = E_{AB}^{\min} \tag{32}$$

and an upper bound

$$E_{AB} \leq s_A s_B U(|\mathbf{y}_A - \mathbf{y}_B| + r_A + r_B) = E_{AB}^{\max}. \tag{33}$$

The energy is minimal if all monomers are concentrated at the point closest to the opposite sphere and maximal if they are at the farthest point.[4] If the estimate is not precise enough, it can be refined by splitting one of the contributing nodes $A \to \{A_{\mathrm{l}}, A_{\mathrm{r}}\}$ or $B \to \{B_{\mathrm{l}}, B_{\mathrm{r}}\}$ such that for instance

$$\begin{aligned} E_{AB}^{\min} &= E_{A_{\mathrm{l}}B}^{\min} + E_{A_{\mathrm{r}}B}^{\min} \quad \text{and} \\ E_{AB}^{\max} &= E_{A_{\mathrm{l}}B}^{\max} + E_{A_{\mathrm{r}}B}^{\max} \end{aligned} \tag{34}$$

provide an improved estimate. This can be done in a recursive fashion similar to the process applied for the hard-sphere polymer. One of the interaction partners also has to be split up if the two spheres are too close to each other in order to allow for an estimation in the first place. For the Lennard-Jones polymer, single monomers are represented by spheres with zero radius, consequently the estimate becomes exact calculation,

$$E_{AB}^{\min} = E_{AB}^{\max} = U(|\mathbf{x}_k - \mathbf{x}_k|), \tag{35}$$

if $s_A = s_B = 1$, and it can be evaluated also for distances below the potential's minimum distance.

If the binary tree is prepared as was done previously (Fig. 4) with the root node possessing the children l and r and if we intend to modify the right part using the transformation $\mathcal{T}_{\mathrm{U}}$ such that symbolically $r \to r' = \mathcal{T}_{\mathrm{U}} r$ then

$$\Delta E \in [E_{lr'}^{\min} - E_{lr}^{\max}, E_{lr'}^{\max} - E_{lr}^{\min}]. \tag{36}$$

Hence, the update is accepted if

$$E_{lr'}^{\max} - E_{lr}^{\min} < -\frac{\ln \xi}{\beta} \tag{37}$$

and rejected if

$$E_{lr'}^{\min} - E_{lr}^{\max} > -\frac{\ln \xi}{\beta}. \tag{38}$$

The interactions between l and r as well as l and $r' = \mathcal{T}_U r$ have to be evaluated and since in almost all cases initially the spheres will be too close or the estimates too rough, almost always interactions between nodes at lower levels will have to be included. Of course, we hope to avoid to consider too many interactions between small groups of monomers such that a decision is reached while interactions are evaluated at a low spatial resolution. This raises the question which particular interaction's estimate should be refined at any given point in order to improve the sum such that the overall process is efficient, i.e., terminates early. Similarly to the hard-sphere polymer it is possible to set up a recursive process that proceeds to smaller nodes until a particular condition is met. While previously non-intersection of the spheres was the only choice, now it is not so straightforward. Clearly the precision of an estimate of a node–node interaction E_{AB} can be derived by just calculating the difference $E_{AB}^{\max} - E_{AB}^{\min}$, but since this scales with the product $s_A s_B$ of the number of monomers of the two groups, it is useful to normalize

$$\alpha_{AB} := \frac{E_{AB}^{\max} - E_{AB}^{\min}}{s_A s_B}, \tag{39}$$

thus measuring the precision per monomer–monomer interaction. Now, we can define a target value α_c and descend to smaller nodes until only interactions that have smaller values α remain. If the result is not sufficiently precise for reaching a decision according to (37), (38) the process is repeated with a lower target value, e.g., $\alpha_c/2$. This approach has the advantage that it can easily be implemented using a recursive function and does not require additional data organization, since only the information about the particular interaction at hand is required.

An alternative, perhaps more intuitive and – as it turns out – more efficient strategy is to select the node–node interaction E_{AB} that possesses the largest absolute uncertainty $E_{AB}^{\max} - E_{AB}^{\min}$ and split its larger node. However, since for large polymers there can be many millions of interactions, finding the most uncertain one is not entirely trivial. Note that the node–node interactions form two binary trees themselves. The roots are the interactions between the nodes l, r and l, r'. Inner nodes in these trees represent estimates of interactions that at some point have been found to be too uncertain or impossible to make, due to close or intersecting spheres, and the current estimate of the total interaction energy is

[4] More sophisticated estimates are possible. For example, if one keeps track of the center of gravity of each group and if the distance between the spheres exceeds the inflection point of the Lennard-Jones potential, it can be shown that the energy is maximal if all monomers are concentrated in the centers of gravity of their respective groups. This energy is, therefore, also an upper bound. However, we found that the additional computations lead to a slower simulation in spite of the reduced depth resulting from the improved estimates.

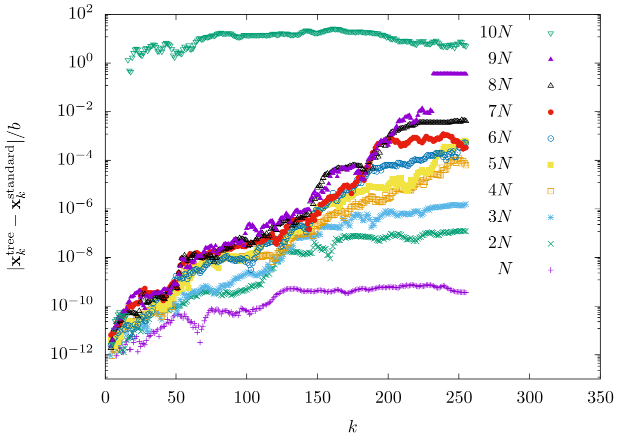

Fig. 7. Difference in monomer positions for different methods (see text). Simplified algorithms that always modify the right part of the polymer were employed, hence the continued agreement for small k.

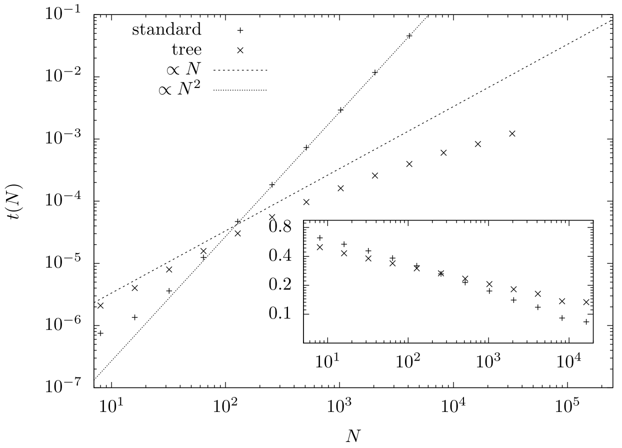

Fig. 8. Required times (in seconds) for the combination of a pivot and a bond-rotation update for the standard method and our algorithm at $k_{\rm B}T = 4\epsilon$ for a Lennard-Jones polymer of length N with untruncated interaction. Inset: Acceptance rate as function of N for the pivot (+) and bond-rotation (×) updates.

obtained by summation over the leaves. A tree is grown by adding two new interaction nodes to a former leaf, thus replacing its contribution in the total sum. These trees can be implemented as actual data structures and used as search trees in order to easily identify the leaf with the largest energy difference. This might be achieved by adding a link (pointer) to every node that points at the leaf with the largest energy difference in the sub-tree to which this node is root. Leafs point to themselves and whenever a leaf is modified, only the leaf itself and its direct ancestors have to be potentially modified by comparing the links in their children. If large polymers are considered with the number of monomers being in the range of several thousands, the trees can reach sizes of multiple millions of nodes. In order to limit the size of required memory we choose to limit the size of the tree and in those rare cases where the limiting size is reached we refrain from growing it further and proceed to improve the estimates in the leaves using the aforementioned recursive function and target precisions.

Comparisons with a standard algorithm that calculates $\varDelta E$ accessing monomer coordinates directly have been done. Both methods were initialized with the same configuration and individual runs for a polymer of size $N = 256$ at $k_{\rm B}T = 4\epsilon$ performed. This temperature is slightly above the Θ-point which we estimate to be $k_{\rm B}T_{\Theta} \approx 3.9\epsilon$ based on data not shown in this study. The difference in monomer positions is shown in Fig. 7. Individual curves show the situation after $N, 2N, \ldots, 10N$ combinations of one pivot and one bond-rotation update. Although both methods should in principle create the same trajectory in state space, associativity as presented in (21) does not hold in a computer simulation. It can make a tiny difference whether multiple transformations are sequentially applied to the monomer's coordinates or whether they are combined beforehand via the multiplication operation. Although the order of multiplication is mathematically irrelevant there are rounding errors that lead to different results. One could argue that since the new method performs fewer multiplications, it does less rounding and is, therefore, likely to be more stable. However, this is far from certain and further investigations are required for a definite answer. Initially, the differences are very small and do not affect whether an update is accepted or rejected, but after a short while the trajectories diverge. For the trajectories in Fig. 7 we find that true differences, i.e., a difference in the sequence of rejections and acceptances of updates, occurs for the first time after $9N$ moves. After $10N$ steps the two configurations no longer bear any resemblance.

Simulations for different sizes at the same temperature allow for comparisons and scaling of running times (Fig. 8). It is of little surprise that the standard technique that requires the calculation of all modified monomer–monomer distances soon approaches quadratic complexity. Our method is now substantially slower than for the hard-sphere polymer, but again it can be suspected that for large systems logarithmic scaling is realized. The acceptance rate is now more strongly affected by the polymer length and seems to decay in polynomial order with a larger albeit still favorably small exponent. We estimate $\propto N^{-0.28}$ for the pivot and $\propto N^{-0.18}$ for the bond-rotation update.

Once a function that estimates the interaction between two nodes in the tree has been set up, it can also be used to obtain an estimate for the energy of the entire polymer. If we define the internal energy of a node recursively as the sum of the interaction between its children and the internal energy of its children with individual monomers (leaves) possessing zero internal energy,

$$\begin{aligned}\mathcal{H}_{k,\ldots,k+s-1} &:= \mathcal{H}_{k,\ldots,k+\frac{s}{2}-1} + \mathcal{H}_{k+\frac{s}{2},\ldots,k+s-1} \\ &\quad + \sum_{i=k}^{k+\frac{s}{2}-1} \sum_{j=k+\frac{s}{2}}^{k+s-1} U(|\mathbf{x}_i - \mathbf{x}_j|), \\ \mathcal{H}_l &:= 0, \end{aligned} \tag{40}$$

then the energy of the polymer is given by the energy of the root node

$$\mathcal{H} = \mathcal{H}_{1,\ldots,N}. \tag{41}$$

Since there are $N - 1$ internal nodes, we need $N - 1$ estimates of node–node interactions each of which might recursively require multiple additional estimates. However, there is a good chance that for non-collapsed states and a target precision not too small this can be done with complexity $\mathcal{O}(N \log N)$ or faster. Once we can estimate the energy of one configuration the estimation of the difference between energies of two configurations is straightforward which allows for an implementation of the replica-exchange algorithm [14]. The exchange probability for swaps between two walkers at inverse temperatures $\beta_1 > \beta_2$ is given by

$$P_{\rm swap}^{\rm accept} = \min\left(1, e^{(\beta_1-\beta_2)(E_1-E_2)}\right), \tag{42}$$

so that the condition for accepting a replica-exchange update with reduced information reads

$$E_1 - E_2 > \frac{\ln \xi}{\beta_1 - \beta_2}. \tag{43}$$

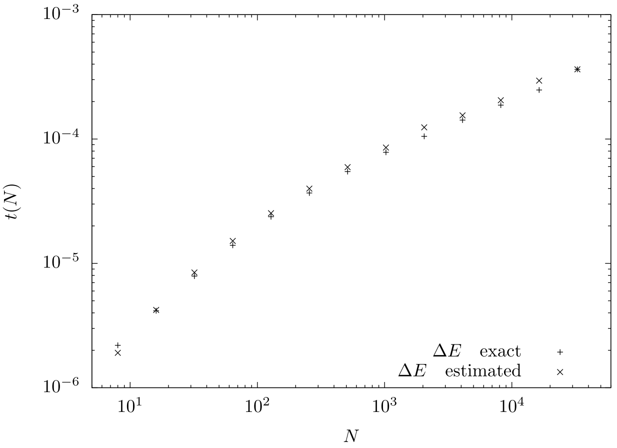

Fig. 9. Time t (in seconds) required for a combination of pivot and bond-rotation update as function of polymer size N for the two versions of the Metropolis algorithm. The Lennard-Jones potential is truncated at 3σ and $k_BT = 3.5\epsilon$.

7.2. Truncated interactions

Finally let us consider the case with a truncated potential which is the traditional technique applied when dealing with large systems. The potential is set to zero beyond a certain cutoff-distance r_c and the remaining part is shifted in order to avoid a discontinuity:

$$\tilde{U}(r) = \begin{cases} U(r) - U(r_c) & \text{if} \quad r < r_c \\ 0 & \text{otherwise.} \end{cases} \tag{44}$$

There are two intuitive ways of simulating such a system using the binary-tree structure. During an update one could proceed similarly to the hard-sphere case and establish which pairs of monomers are closer than the cutoff distance. Calculating their energies allows for a precise determination of ΔE and the standard Metropolis algorithm can be applied. Or the algorithm using reduced information could simply be used with the truncated potential. We would expect that the former is more efficient at low temperatures, since the estimation of interactions at small distances in dense configurations is less precise and many refinements might be necessary. At conditions near the collapse ($k_BT = 3.5\epsilon, r_c = 3\sigma$), however, we find that both methods perform similarly well (Fig. 9). It is worth noting that in comparison to the untruncated case even for the largest systems considered ($N = 2^{15}$) the introduction of the cutoff has only led to a threefold speedup.

8. Conclusion and outlook

In this study we have shown how the binary-tree method developed by Clisby for the simulation of self-avoiding walks on a lattice can be adapted to hard-sphere polymers with continuous degrees of freedom. It turns out that system sizes of $N \approx 10^7$ are not beyond the capabilities of these techniques. Although we cannot be certain at this point, it seems that for very large systems the computational complexity of an individual Monte Carlo move scales like $\log N$ which is a considerable improvement over standard techniques which are of complexity $O(N)$ or worse.

We introduced a version of the Metropolis algorithm that does not rely on exact knowledge of the change in energy and reaches decisions based on sufficiently precise estimates. We applied this method to a Lennard-Jones polymer without any interaction-range cutoff close to the collapse transition and find that again a scaling of $\log N$ for single steps seems to be the asymptotic behavior. Again, this is substantially better than the $O(N^2)$ scaling of the established methods. It turns out that self-interacting polymers up to length $N \approx 10^4$ can be investigated easily. The same idea in combination with the fact that estimates of the total energy can be obtained much faster than the exact value can be used to implement an improved version of the replica-exchange algorithm for these systems.

Although an interaction-range cutoff of the potential as a means to enable a simulation in the first place is no longer required, polymers using such a truncated interaction can, of course, be simulated using these methods. We find that for the system sizes considered here simulations are only modestly faster when a cutoff to the Lennard-Jones potential is used.

This work is intended to serve mainly as a proof-of-concept with a focus on performance. Along the way we demonstrated the universality of the scaling behavior of lattice and continuum self-avoiding walks by conducting simulations of the latter with up to $N = 2^{23} \approx 10^7$ steps. Although relatively simple models were considered in this study, it should be possible to introduce extensions like flexible spring-like bonds, bending stiffness or fixed bond angles, or multiple types of monomers with little effort.

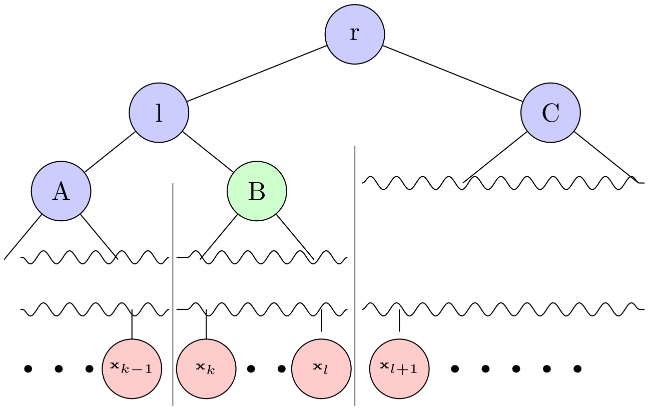

Fig. A.1. The binary tree during an update of the monomers $\mathbf{x}_k, \ldots, \mathbf{x}_l$ which are at the highest level represented by node B. Node l and r limit this section on the left and right and node–node interactions (or overlaps) have to be considered between A,B and B,C.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We thank Nathan Clisby for helpful discussions. The project was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) through the Collaborative Research Centre under Grant No. 189853844-SFB/TRR 102 (project B04).

Appendix. Modifying central parts of the polymer

As we have shown, for the Lennard-Jones polymer the acceptance rates decline when the system becomes large. It is, therefore, desirable to introduce additional updates of a more local nature that are not affected in this manner. This can be a crank-shaft move, where a part of the polymer is rotated around an axis passing through the limiting monomers or – if a model with flexible bonds is used – the shift of one or more adjacent monomers by a constant vector. The procedure is similar to the discussed update, with the distinction that now we have two internal nodes that pose the boundaries of the section that is to be moved. The tree is rearranged in a way that first one of them becomes root and in a second phase the other becomes a child of the new root node (Fig. A.1). With the tree in this shape the three relevant parts of the polymer are represented by single nodes and interaction between them can be evaluated recursively. Once the update is accepted it is again possible to apply the respective transformation at the highest level to a single node before rebalancing the tree.

References

[1] R.D. Schram, G.T. Barkema, R.H. Bisseling, J. Stat. Mech. (2011) P06019.
[2] N. Clisby, Phys. Rev. Lett. 104 (2010) 055702.
[3] N. Clisby, J. Stat. Phys. 140 (2010) 349.
[4] N. Clisby, B. Dünweg, Phys. Rev. E 94 (2016) 052102.
[5] P. Grassberger, R. Hegger, J. Chem. Phys. 102 (1995) 6881.
[6] F. Rampf, K. Binder, W. Paul, J. Polym. Sci. B 44 (2006) 2542.
[7] T. Vogel, M. Bachmann, W. Janke, Phys. Rev. E 76 (2007) 061803.
[8] A.M. Rubio, J.J. Freire, C.W. Yong, J.H.R. Clarke, J. Chem. Phys. 111 (1999) 1302.
[9] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, J. Chem. Phys. 21 (1953) 1087.
[10] N. Madras, A.D. Sokal, J. Stat. Phys. 50 (1988) 109.
[11] M. Bachmann, H. Arkın, W. Janke, Phys. Rev. E 71 (2005) 031906.
[12] S. Schnabel, M. Bachmann, W. Janke, J. Comput. Phys. 230 (2011) 4454.
[13] J. Barnes, P. Hut, Nature 324 (1986) 446.
[14] K. Hukushima, K. Nemoto, J. Phys. Soc. Japan 65 (1996) 1604.