Simulated Annealing with Coarse Graining and Distributed Computing

Andreas Pedersen, Jean-Claude Berthet, and Hannes Jónsson

Science Institute, VR-III, University of Iceland, 107 Reykjavík, Iceland

Abstract. EON is a software package that uses distributed computing, systematic coarse graining and bookkeeping of minima and first order saddle points too speed up adaptive kinetic Monte Carlo simulations. It can be used to optimize continuously differentiable functions of a large number of variables. The approach is based on finding minima of the cost function by traversing low-lying, first-order saddle points from one minimum to another. A sequence of minima is thus generated in a path through regions of low values of the cost function with the possibility of 'temperature' controlled acceptance of higher lying saddle points. Searches of first order saddle points are carried out using distributed computing and the minimum-mode following method. Coarse graining which involves merging local minima into composite states and the recognition of previous search paths and saddle points are used to accelerate the exploration of the cost function. In addition to obtaining an estimate of the global minimum, a simulation using this approach gives information about the shape of the cost function in the regions explored. Example applications to the simulated annealing of a cluster of water molecules on a platinum metal surface and grain boundary in copper are presented.

Keywords: optimization, distributed computing, coarse graining, water cluster, grain boundary.

1 Introduction

Annealing has for a long time been used to improve the atomic scale structure of materials by eliminating strain and defects. Glassblowers, for example, use annealing to prevent new glass structures from cracking. When a material is annealed, its temperature is brought to an elevated level to accelerate thermally activated mobility of defects. However, the elevated temperature can also increase the population of defects. By cooling slowly enough from the elevated temperature, the defects become less stable while still being mobile enough to get annihilated. The result of this treatment is a lowering of the energy as the arrangement of the atoms is optimized. If the cooling is slow enough, all defects will be eliminated and the global energy minimum reached. This, however, may require impossibly long time.

Optimization of functions of many variables is often carried out using computer simulated annealing algorithms that mimmic roughly the annealing of

K. Jónasson (Ed.): PARA 2010, Part II, LNCS 7134, pp. 34–44, 2012.

materials. The 1983 article by Kirkpatrick, Gelatt and Vecchi [1] where such an approach was promoted and applied to circuit design now has over 12000 citations. The cost function to be minimized is taken to give the 'energy' of the system. A Monte Carlo algorithm based on random numbers is used to simulate an annealing process were changes in the arguments of the cost function are accepted or rejected in accordance with a fictitious 'temperature'. The reason for introducing temperature is to introduce and control the probability of accepting increases in the cost function since they may be an essential intermediate step to ultimately reach lower values.

The Adaptive kinetic Monte Carlo algorithm (AKMC) [2] can be used to accelerate simulations of the time evolution in materials without the need for a priori information about possible transitions in the system. This is unlike the regular KMC algorithm where the mechanism and rate of possible transitions is needed as input [3]. It is also different from the Metropolis Monte Carlo algorithm [4] in that the changes made to the variables represent likely transition mechanisms rather than just random moves and one can estimate the 'time' evolved at each iteration. The AKMC method has been applied successfully to several different problems, for example the annealing of grain boundaries in metals [5], reactions at surfaces [6] and crystal surface annealing during growth [7]. In AKMC, the possible transitions are found by locating first-order saddle points on the energy surface that are in the vicinity of currently known local minima. The probability of the possible transitions decreases exponentially as a function of the height of the saddle point over the current minimum. A random number is used to pick the next transition according to the relative rates of possible transitions. Several additional features in the implementation have been developed to speed up the simulation such as (1) distributed computing using the BOINC communication framework [18] were hundreds of computers connected by internet can be used simultaneously for the saddle point searches; (2) systematic coarse graining were local minima separated by low-lying saddle points are grouped together to form composite states; (3) bookkeeping of previously found saddle points and search paths to suggest new, short searches and terminate searches that are likely to converge to known saddle points. The software package EON [8] has been written to carry out such simulations. While it has so far been applied to studies of atomic scale systems, it can in principle be used for optimization of any cost function where first derivatives are continuous and can be evaluated readily. Since the algorithm follows regions where the cost function is relatively small, this approach may be particularly useful when the calculation of high values of the cost function can be problematic because of, for example, convergence problems.

In this article we describe briefly the algorithms on which the EON software is based and discuss two applications: (1) ordering of water molecules in a cluster on a Pt surface, and (2) atomic ordering at a grain boundary in Cu. We conclude with some remarks on the applicability of this software to other, more general optimization problems.

2 The Cost Function

The cost function is assumed to be a continuously differentiable function of N variables

$$f : \mathbb{R}^N \rightarrow \mathbb{R} \tag{1}$$

In typical applications of AKMC such as the ones described below, N is on the order of 10^3. The cost function defines a surface in high dimensional space and the goal is to find the global minimum, or a representative minimum if there are several roughly equally low minima. When navigating on the surface, the extremal points where $\nabla f = 0$ and the function value is low are of particular interest, namely local minima and first order saddle points. At a minimum, the Hessian matrix, $\mathrm{H_{ij}} = \frac{\partial^2 f}{\partial x_i \partial x_j}$, has only positive eigenvalues but at a first order saddle point it has one negative eigenvalue. Hereafter, a saddle point will be taken to mean *first order* saddle point. An *optimal path* is such that at each point on the path the gradient is pointing along the path [9,10]

$$\nabla f - \nabla f \cdot \hat{\tau}\ \hat{\tau}\ =\ 0 \tag{2}$$

were $\hat{\tau}$ is the normalized tangent vector for the path at that point [11]. A first order saddle point is a maximum along an optimal path between two minima. We assume that the gradient ∇f of the cost function can be evaluated readily (recent developments in automatic differentiation [12] could prove valuable in this context), but second derivatives are not needed. The method used to find first order saddle points involves a minimization using a transformed gradient where the component along the minimum mode of the Hessian is reversed

$$\nabla f^{eff} = \nabla f - 2(\nabla f \cdot \hat{v}_\lambda)\hat{v}_\lambda \tag{3}$$

Here, $\hat{v}_\lambda$ is a normalized eigenvector corresponding to the minimum eigenvalue, λ, of the Hessian. This projection locally transforms the gradient in the vicinity of a first order saddle point to a gradient characteristic of the vicinity of a minimum and the conjugate gradient method (without line search [13,14]), for example, can be used to converge on the first order saddle point. The minimum mode vector is found using the dimer method [13] without having to evaluate the Hessian or any of the second derivatives. Given a local minimum, a small random change of the variables away from the minimum is first made and the minimum mode following (MMF) method is then used to climb up the cost function surface and converge on to a saddle point. For more detail about this method and its performance, see refs. [14].

Once a search trajectory exits the region in the neighborhood of a minimum where all eigenvalues are positive, the MMF search path is stable and deterministic, i.e. a given point outside the positive region will converge onto a certain saddle point. A basin of attraction can be defined as illustrated in Fig. 1.

3 Adaptive Kinetic Monte Carlo Algorithm

The kinetic Monte Carlo method (KMC) [3] is an iterative algorithm where a Markov chain of states is generated using a predefined rule for transitions and

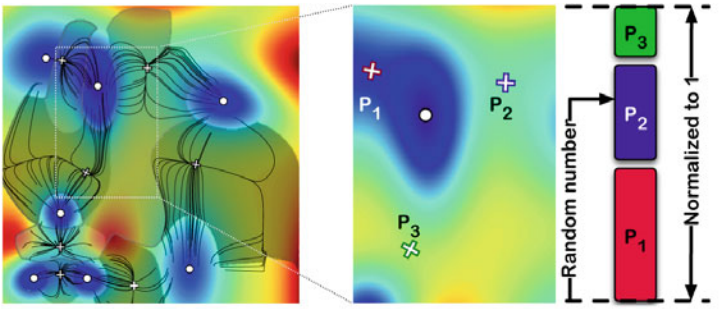

Fig. 1. Left panel: A model cost function and paths from multiple saddle point searches carried out with the MMF method. Minima are denoted by a white dot and saddle points by a white + with arms pointing along the eigenvectors of the Hessian. Shaded regions are basins of attraction of the saddle points. For each local minimum, several paths (solid lines) leading to saddle points are generated using the MMF method. The paths tend to merge as they approach a saddle point. Searches that lead to saddle points outside the figure are not shown. Right panel: Schematic illustration of an iteration in the AKMC method. For each saddle point, a rate can be evaluated from Eqn. 6. The normalized rate for each of the three possible transitions gives a transition probability, represented by the colored regions in the column, and a random number can be used to pick the next transition.

their rate constants

$$C^{KMC} = \{f_0, f_1, f_2, \dots f_\eta\}. \tag{4}$$

The mechanism and probability of transitions needs to be read in as input before the simulation starts. In the adaptive kinetic Monte Carlo method (AKMC) [2] this input is not required but at each state, which is a minimum of the cost function, multiple low lying saddle points are found by MMF searches starting from slightly different starting points (as illustrated in Fig. 1) where the displacements are drawn from a Gaussian distribution . Each saddle point or rather the optimal path going through the saddle point, represents a transition mechanism and a possible new state in the Markov chain. After several low lying saddle points have been found, a random number in the interval $[0, 1]$ is used to pick one of the possible transitions according to the relative rates. By repeating this procedure, a chain of minima and associated saddle points is generated

$$C^{AKMC} = \left\{f_0^{min}, \{f_1^{sp}, f_1^{min}\}, \{f_2^{sp}, f_2^{min}\}, \dots \{f_\eta^{sp}, f_\eta^{min}\}\right\}. \tag{5}$$

The rate constant for a transition from $i-1$ to i via saddle point i is an exponential function of the value of the cost function at the saddle point, consistent with Boltzmann distribution of thermal energy

$$k_i \;=\; A\, \exp\left[-\frac{f_i^{sp} - f_{i-1}^{min}}{T}\right] \tag{6}$$

where T is the temperature and A is some prefactor which can in the simplest approach be taken to be the same for all transitions. In transition state theory

(TST) of thermal transitions on an energy surface [15], the rate constant is estimated from the probability of a thermal fluctuation that brings the system to a transition state separating the initial state from other states multiplied by the rate of crossing the transition state. After making second order Taylor expansions of the cost function about the saddle point and the minimum, the expression for the prefactor becomes [16]

$$A^{HTST} = \frac{\prod_j^D \nu_j^{min}}{\prod_j^{D-1} \nu_j^{sp}} \tag{7}$$

where ν_j^{min} is the frequency associated with eigenmode j at the minimum and ν_j^{sp} at the saddle point. The mode with negative eigenvalue at the saddle point is not included. This expression does require the evaluation of the Hessian at the minimum and at the saddle point. It assigns higher probability for transitions where the 'valley' widens when going from the minimum to the saddle point and lower probability if the valley narrows. But, typically the value is similar for transitions in a given system and AKMC simulations are sometimes carried out using just a fixed prefactor, thereby avoiding the evaluation of second derivatives all together. The saddle point searches are continued until a predefined criterion has been reached, for example that the lowest saddle point has been found a certain number of times, or that no new saddle point in the relevant range (defined by the temperature) has been found in the last n searches where n is some predefined number.

From all the successful saddle point searches from the current minimum, i, in the Markov chain (some searches may not converge, and some may converge to a saddle point that is not directly connected by an optimal path to the current minimum), a table of possible transitions and their normalized probability, P_i is constructed. This is illustrated in Fig. 1. A random number is then used to pick one of the possible transitions and the system advanced to a new state, found by sliding down forward along the optimal path from the saddle point. From the Poisson distribution of residence time, the expectation value of the time evolved in each iteration can be estimated to be $\tau = 1/\sum_i k_i$ where the sum extends over all possible transitions from this state. Random sampling from this distribution is made by picking another random number, χ, in the interval $[0, 1]$ and estimating the time increment as $\tau = -\log \chi / \sum_i k_i$.

3.1 Distributed Computing

The most computationally demanding part of an AKMC simulation is the search for saddle points. But, each search is independent of the others, requires only the location of the current local minimum and an initial search direction as input. For large systems each search can take several minutes. The saddle point searches can, therefore, be distributed to several computers connected only by internet. The EON software is based on such a distributed computing approach [8]. In the first version, the communication middleware Mithral was used [17] but a more recent version of EON is based on BOINC. In the later implementation,

the client software is executed as a remote procedure and the communication uses *http*. BOINC offers stability of the central server as the core parts are a SQL database and an Apache server. The client and server are decoupled which is particularly useful when debugging since the client calculation can be run as stand-alone calculation and ordinary debugging tools applied.

Even though BOINC supports distributed computing using public computers, we typically only run EON clients on local resources in clusters dedicated to research where EON-clients get executed when nodes become idle. Currently, EON is being extended to work also with the ARC-middleware [19] which will enable execution of EON on the Nordic data grid [20] and it is, furthermore, being extended to run on Amazon's 'elastic cloud' [21].

A rough estimate of the performance of a distributed AKMC simulation using EON carried out on a cluster of 2.8 GHz Intel Pentium 4 computers with GB internet connection shows that if a calculation on the client takes ca. 5 min. then the server can on average keep 100 clients busy. When saddle point searches are too fast for the server to keep up, two or more searches can be sent to the client at a time, thus saving on the overhead in the distribution of the tasks.

3.2 Coarse Graining

Kinetic Monte Carlo simulations, including AKMC, often get stuck in a small subset of states. This happens, for instance, when two states are connected by a transition that is much faster than any transition away from the pair. Essentially, the problem arises when there is large disparity in the rate constants. To address this problem, a coarse graining algorithm has been developed and implemented in EON. The problem is two-fold: (1) to automatically identify such sets of states and group them into a *composite state*, and (2) devise an algorithm for escaping the composite state and estimating the time that would have been spent there in the absence of coarse graining.

The first issue is addressed by defining a reference value of the cost function for each state, f_i^{cg} which starts out being equal to the minimum value $f_i^{cg} = f_i^{min}$. Each time this minimum is entered and added to the Markov chain, the reference value is incremented by a small amount, ϵ

$$\epsilon \;=\; \epsilon_0 \, \frac{f_i^{sp} - f_i^{min}}{f_i^{sp} - f_{low}^{min}} \tag{8}$$

where f_{low}^{min} is the lowest minimum found in the simulation so far and ϵ_0 is some constant chosen for the simulation (the results of the simulation are unaffected by the choice of ϵ_0 over several orders of magnitude as illustrated in a test case below). When a transition occurs with $f_i^{sp} < f_{i-1}^{cg}$, then state i is merged with state $i-1$ to form a new composite state. If either of the states i or $i-1$ or both already belong to a composite state, then the composite states are extended by an additional state or two composite states are merged depending on the situation.

This process of merging states into composite states is illustrated in Fig. 2. The simulation starts at the state corresponding to the second lowest minimum

and there is a high saddle point separating it from the global minimum. Only after all the minima close to the initial minimum have been grouped into one large composite state, can the high saddle point be overcome and the global minimum is then found after only a few iterations. This example illustrates how EON could be used to find the global minimum of a cost function where multiple local minima are present and one or more high saddle point separate groups of minima. When a composite state has been formed from a set of minima, the time evolution within that set of states becomes irrelevant and only two things need to be determined: (1) From which one of the local minima will the system escape the composite state? (2) How long time will the system spend in the composite state before escaping. These questions can be answered rigorously using absorbing Markov chain theory [22].

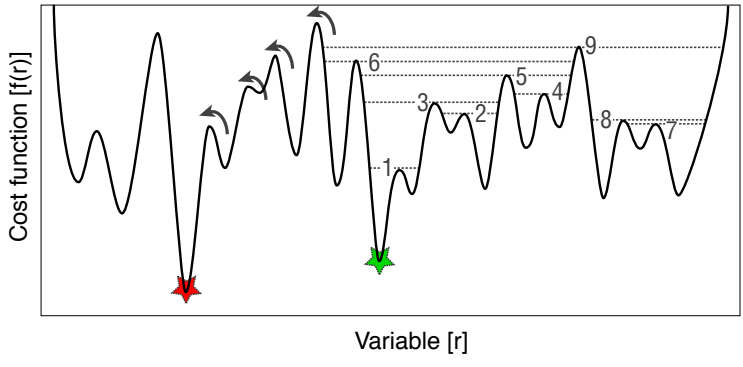

Fig. 2. Example of a cost function of one variable where the annealing simulation starts in a basin consisting of several local minima but is separated by a large barrier from the global minimum. Without coarse graining the simulation would be stuck for many iterations between the initial state (green star) and the adjacent shoulder state. These two are the first to get merged into a composite state (1). Then a new composite state (2) is formed, also by merging two minima. In the third merger the two composite states are combined into a composite state with four minima (3). Eventually, the large barrier is overcome when all 10 minima have been merged into one composite state (9). Then, the global minimum (red star) is quickly reached.

Test: Annealing in One-dimension. A problem where the coarse graining algorithm was tested on a cost function of one variable is illustrated in Figure 3. An AKMC simulation both with and without coarse graining was used to estimate the time it takes the system to go from a high initial state minimum to the global minimum, traversing a high barrier. The results are presented on Table 1 and show that the coarse graining does not affect the estimated time, even as the basic increment, ϵ_0, is varied over several orders of magnitude.

This simple test illustrates how the coarse graining speeds up the annealing simulation without affecting the results, in particular the estimated time. In this case, the acceleration is up to 100 fold. But, more importantly, the coarse

graining can make a simulation doable while it is impossible without the coarse graining (for example, this same objective function but a significantly lower temperature). Fig. 3 shows how the number of iterations needed in a KMC simulation increases for the simple one-dimensional test case as the temperature is lowered. Without the coarse graining, the number of iterations needed to reach the global minimum is about three orders of magnitude larger at a temperature of 0.007 than at 0.1, but with the coarse graining there is only less than a factor of 2 increase.

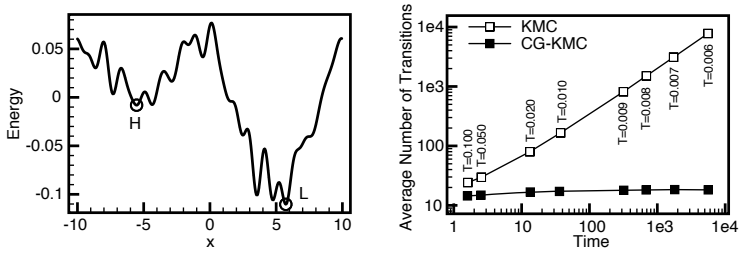

Fig. 3. A model cost function of one variable used to test the coarse graining algorithm. Starting in state H, the AKMC simulation with and without coarse graining was used to estimate the time needed to reach the global minimum, L. The results are given in Table 1 and show that the coarse graining does not affect the estimated time, even as the basic increment, ϵ_0, is varied over many orders of magnitude. Right panel: Average number of transitions (iterations) needed in the simulations versus the estimated mean time needed to go from point H to point L for different 'temperatures'. While the regular KMC needs many more iterations at low temperature, the coarse grained simulation only requires slightly more.

Table 1. The elapsed time and number of transitions required to move from local minimum H to global minimum L in Fig. 3, averaged over 1000 runs at a 'temperature' of 0.09, with no coarse graining (no CG) and with using various values of the basic increment, ϵ_0 in the coarse graining algorithm. The prefactor was set to $A = 10$.

$\log \epsilon_0$	Time	Ave. Trans.
no CG	708 ± 23	1547 ± 51
-5	670 ± 20	1287 ± 31
-4	684 ± 21	405 ± 6
-3	683 ± 18	71 ± 1
-2	698 ± 13	15 ± 0

3.3 Other Tricks to Improve Efficiency

Each minimum visited is stored, i.e. the value of the variables and the value of the cost function. When a minimum is revisited, new saddle point searches are not carried out unless the reference energy has increased beyond the region where

good sampling of saddle points has been obtained previously. Also, since search paths tend to merge together close to the saddle points, as shown in Fig. 1 intermediate points along search paths in the basin of attraction (where one eigenvalue is negative) are stored and used to terminate later searches that come close to a previous search path. Only the values of variables that change most along the path are stored. We refer to this as the 'skipping-path' method [23]. Around each of the stored intermediate points, a spherical region with small radius is defined and any later search that comes within one of these regions gets terminated and the intermediate points of that search added to the list. When thorough sampling of saddle points is carried out, this can save substantial computational effort [23].

In some systems the transitions are 'local' in that they only involve appreciable change in some small subset of the variables. Then, one transition may not affect strongly transitions involving another set of variables. An example of that from the atomic scale simulations is a rearrangement of atoms in one part of the simulated system affecting only insignificantly atoms in another part of the system. The stored saddle points can, therefore, be good guesses for new saddle point searches where only a few iterations using the MMF method are needed to re-converge on saddle points. This can save substantial amount of computational effort [24].

4 Applications

Two applications of simulations with EON are presented here briefly, both involving the optimization of the structure of atomic scale systems. In the first, a cluster consisting of 7 water molecules adsorbed on a Pt(100) surface is annealed to find the optimal structure. The interaction potential function was developed by Zhu and Philpott [25] but the interaction between the water molecules and the surface has been scaled down to match recent density functional theory calculations [26]. The initial configuration is generated by adding two molecules to the optimal configuration of a cluster consisting of five molecules, which is a pentagonal ring. The structure is shown in Fig. 4. A shallow minimum slightly higher in energy has one of the molecules rotated in such a way that a hydrogen atom gets pointed away from the surface. The AKMC simulation jumps between those two states several times until eventually the two are joined in a composite state. A significant barrier is involved in breaking up the pentagonal ring to form a rectangular core of 6 molecules with one molecule weakly bound to the edge, but this leads to substantial lowering of the energy. After that, the AKMC simulation flips the under-coordinated edge molecule and this is repeated until the two configurations get merged in a composite state.

The second example comes from a study of a twist and tilt grain boundary in copper. The results of an extensive study have been presented elsewhere [5] but here we present for the first time the improvement obtained by using coarse graining. A potential function of the EMT form is used here to describe the intermolecular interactions [27]. The simulated system consists of 1309 atoms

but the two outermost layers parallel to the grain boundary are kept frozen so the cost function (the energy of the system) is a function of 3567 variables, three coordinates for each movable atom. The goal is to gain information about the atomic arrangement at the grain boundary. Away from the grain boundary the atoms are in a FCC crystal arrangement. One of the questions addressed in these studies is how wide a region around the grain boundary atoms are anomalously coordinated and what arrangement of atoms is most characteristic for the grain boundary. Since the atoms that are not in an FCC crystal arrangement are the focus here and are most likely to be involved in transitions, the initial random displacement is generated in a spherical region including about 30 atoms centered on a randomly picked, non-FCC atom. Typically, 1 to 10 atoms are displaced by more than half an Ångström in a single transition [5].

The excess energy due to the non-FCC arrangement of atoms at the grain boundary, the interface energy, is shown as a function of iterations in the simulation for a temperature of 300 K in Fig. 4. At first, the coarse graining was turned off and while some transitions occurred and annealing took place, the simulation was not making any progress for a long time. The energy in the last 200 iterations is shown far to the left in the figure. The system is going back and forth between a few states with interface energy between 81.8 and 82.0 meV/Å^2. Then, coarse graining is turned on (iteration labeled '0') and a composite state is formed which enables the simulation to get to higher energy states, up to 82.4

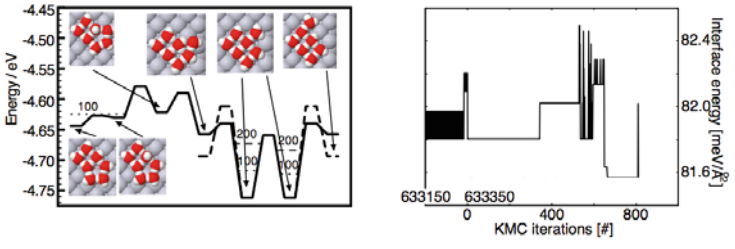

Fig. 4. Left: Simulation of a cluster of seven water molecules on a Pt(100) surface. The horizontal dashed lines show the reference level, f^{cg}, after 100 and 200 transitions. First, the two states on the far left are merged into a composite state. Then, the pentagonal ring breaks up and a rectangular hexamer core is formed with one water molecule at the edge. The difference between the two deep minima to the right is just the orientation of the low coordinated molecule. This kind of rotational flipping which does not represent a change in the energy or the structure is a significant problem in many simulations when coarse graining is not used. Right: Interface energy in a twist and tilt grain boundary between copper crystal grains as a function of iterations in an AKMC simulation. After 633350 iterations without coarse graining where the simulation was caught in a subset of 11 states and little progress was made (marked as '0' on the horizontal axis), the coarse graining algorithm is turned on. Then, a composite state of these minima was formed and the simulation reached higher energy states which eventually lead to a significant annealing event. By applying coarse graining, only 700 additional iterations were required to reach this annealing event while it took 300.000 additional iterations without coarse graining.

meV/Å^2 and eventually find a transition that reduces the energy significantly, an annealing event. It took less than 700 iterations from the time the coarse graining was turned on until the annealing event was observed. The same kind of annealing event was also observed by continuing the simulation without coarse graining, but then 300.000 iterations were needed.

Acknowledgments. The EON software is being developed in collaboration with the research group of Prof. G. Henkelman at the University of Texas at Austin. This work was supported by the Icelandic Research Fund, the University of Iceland research fund, and EC Marie Curie network 'Hydrogen'.

References

1. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Science 220, 671 (1983)
2. Henkelman, G., Jónsson, H.: J. Chem. Phys. 115, 9657 (2001)
3. Glauber, R.J.: J. Math. Phys. 4, 294 (1963); Martin, P.A.: J. Stat. Phys. 16, 149 (1977); Bortz, A.B., et al.: J. Comput. Phys. 17, 10 (1975)
4. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: J. Chem. Phys. 21, 1087 (1953)
5. Pedersen, A., Henkelman, G., Schiøtz, J., Jónsson, H.: New Journal of Physics 11, 073034 (2009)
6. Xu, L., Henkelman, G.: J. Chem. Phys. 131, 244520 (2009)
7. Henkelman, G., Jónsson, H.: Physical Review Letters 90, 116101 (2003)
8. Pedersen, A., Jónsson, H.: Math. and Comput. in Simulations 80, 1487 (2010)
9. Jónsson, H., Mills, G., Jacobsen, K.W.: Classical and Quantum Dynamics in Condensed Phase Simulations. In: Berne, B.J., Ciccotti, G., Coker, D.F. (eds.), ch.16, p. 385. World Scientific (1998)
10. Henkelman, G., Uberuaga, B., Jónsson, H.: J. Chem. Phys. 113, 9901 (2000)
11. Henkelman, G., Jónsson, H.: J. Chem. Phys. 113, 9978 (2000)
12. `http://www.autodiff.org`
13. Henkelman, G., Jónsson, H.: J. Chem. Phys. 111, 7010 (1999)
14. Olsen, R.A., Kroes, G.J., Henkelman, G., Arnaldsson, A., Jónsson, H.: J. Chem. Phys. 121, 9776 (2004)
15. Wigner, E.: Tr. Far. Soc. 34, 29 (1938); Eyring, H.: J. Chem. Phys. 3, 107 (1935)
16. Vineyard, G.H.: J. Phys. Chem. Solids 3, 121 (1957)
17. `http://www.mithral.com/projects/3rdparty/`
18. `http://boinc.berkeley.edu/`
19. `http://www.nordugrid.org/`
20. `http://www.ndgf.org/`
21. `http://aws.amazon.com/ec2/`
22. Novotny, M.A.: A tutorial on advanced dynamic Monte Carlo methods for systems with discrete state spaces (2001), URL arXiv.org:cond-mat/0109182
23. Pedersen, A., Hafstein, S.F., Jónsson, H.: SIAM Journal of Scientific Computing 33, 633 (2011)
24. Xu, L., Henkelman, G.: J. Chem. Phys. 129, 114104 (2008)
25. Zhu, S.-B., Philpott, M.R.: J. Chem. Phys. 100, 6961 (1994)
26. Árnadóttir, L., Stuve, E., Jónsson, H.: Surf. Sci. 604, 1978 (2010); Surf. Sci. (in press, 2011)
27. Jacobsen, K.W., Stoltze, P., Nørskov, J.K.: Surf. Sci. 366, 394 (1996)