Journal of Computational Physics **173,** 302–321 (2001)
doi:10.1006/jcph.2001.6877, available online at http://www.idealibrary.com on IDEAL®

Monte Carlo Algorithms for Complex Surface Reaction Mechanisms: Efficiency and Accuracy

J. S. Reese,* S. Raimondeau,* and D. G. Vlachos†

**Department of Chemical Engineering, University of Massachusetts, Amherst, Massachusetts 01003; and †Department of Chemical Engineering and Center of Catalytic Science and Technology (CCST), University of Delaware, Newark, Delaware 19716*
E-mail: vlachos@che.Udel.edu

Received February 9, 2001; revised June 29, 2001

A continuous-time Monte Carlo (CTMC) algorithm with lists of neighbors and local update (tree-type architecture) for simulating the dynamics and stationary pattern formation of complex surface reaction mechanisms is discussed. Two additional CTMC algorithms, often used in the literature, are also presented. The computational efficiency of these CTMC algorithms is compared to a null-event algorithm for the CO oxidation on a Pt(100) surface by direct numerical simulations. Furthermore, we have derived simple formulas for the real time advanced using the null-event algorithm and the CTMC with local update algorithm for the infinitely fast and finite CO oxidation kinetics as well as a unimolecular surface reaction. We have found that the proposed CTMC algorithm with classes and local update can be much faster than the traditional null-event algorithms by orders of magnitude, when stiffness occurs (rare event dynamics). In addition, we address the computational accuracy of Monte Carlo algorithms, due to limited resolution caused by finite lattice sizes, for key intermediate species in a complex reaction mechanism. It is shown that surface concentrations below the resolution of the lattice and corresponding reaction rates can accurately be calculated through the use of a time-weighted average of reaction rates.

INTRODUCTION

Surface reactions are an important component in the design and optimization of homogeneous–heterogeneous systems such as catalytic and chemical vapor deposition reactors. Traditionally, surface reactions have been modeled assuming a uniform distribution of each species on the surface. This idealized situation is often envisioned for infinite Fickian-type diffusion coefficients of all surface species. This mean field (MF) approximation does not take into account the stochastic nature of nucleation phenomena and spatial

0021-9991/01 $35.00

inhomogeneities associated with heterogeneous systems such as crystal defects and lateral adsorbate–adsorbate interactions. In addition, any nonlinear rate term in a MF conservation equation, such as the dissociative adsorption of a diatomic molecule on a surface or the reaction rate of a bimolecular surface reaction, has an explicit dependence on the local microenvironment of atoms resulting in spatial micropatterns. These spatial inhomogeneities render the MF approximation a poor representation of experimental findings. In order to overcome this problem, Monte Carlo (MC) techniques are used on a lattice representing the chemisorption sites of various species. MC simulations solve a master equation

$$\frac{dP_\alpha}{dt} = \sum_\beta [W_{\alpha\beta} P_\beta - W_{\beta\alpha} P_\alpha], \tag{1}$$

where P_α is the probability of the surface being in configuration α and $W_{\alpha\beta}$ is the transition probability per unit time of the surface going from configuration β to α. Given the large number of possible configurations, Eq. (1) cannot be solved analytically for any real system. As a result, methods such as MC are often implemented. This representation belongs to the general class of interacting Ising particle systems (IPS) of nonequilibrium statistical mechanics.

MC simulations are typically computationally intensive and thus, the implementation algorithm can have important consequences for computational efficiency. The first group of MC algorithms for surface reactions, introduced by Ziff and co-workers, used sites or pairs selected at random and assumed that an event may occur or not (a null-event algorithm). This algorithm is known as the ZGB method [1]. Following up on their pioneering work, many studies have employed variations of the original algorithm to include additional mechanistic steps [2–6]. Regarding specific reaction systems, this *null-event algorithm* has been used to model monomer–dimer reaction systems such as the CO oxidation by O_2, monomer–monomer surface reactions, dimer–dimer surface reactions, and more complicated reaction mechanisms such as the CO methanation and the catalytic reduction of NO by CO [7–18]. A similar null-event, real-time algorithm was also used for a unimolecular surface reaction [19, 20]. For an overview of many physical applications that have been modeled using MC, see the review papers by Evans [21], Zhdanov and Kasemo [22, 23], Kang and Weinberg [24], Jansen and Lukkien [25], Lombardo and Bell [26], and Nieminen and Jansen [27]. The latter reference has an overall review of ZGB algorithm modifications.

It turns out that the application of the null-event algorithms to more complicated reaction schemes is nontrivial, and time is often presented in MC trials or steps. An exception to this can be found for a unimolecular reaction [19, 20, 28]. However, the methods presented there, based on unimolecular events only, cannot be directly applied to more complex kinetic systems (e.g., the $A_2 + B_2$ reaction). Furthermore, real time is sometimes miscomputed [23, 29]. Modifications to the ZGB algorithm to reduce or eliminate null events have been proposed and implemented to model CO oxidation [30–35]. These algorithms are a derivative of the one proposed by Bortz *et al.*, often referred to as the continuous-time MC method (CTMC) or kinetic MC (KMC) method [36]. For spatially homogeneous reactions, related, real-time MC algorithms were introduced long ago by Gillespie [37, 38]. A variation of such an algorithm, discussed below, has proven to be computationally very efficient at low temperatures in growth, etching, and equilibration of materials [39] over the most commonly used Metropolis algorithm [40]. This last algorithm has also been used to model a unimolecular surface reaction coupled to a transport model in the adjacent gas-phase of a

catalyst [41] and (with a lumping technique to handle numerical stiffness caused by partial equilibrium situations) systems with detailed chemistry, such as hydrogen ignition [42].

In this paper, a general algorithmic framework is presented that is capable of handling arbitrarily complex, detailed surface reaction mechanisms based on the CTMC method, similar to the work of [30]. This feature is essential to replacing computational packages based on MF approximations, such as Surface CHEMKIN, in large-scale reactor models [43]. Because of the lack of real time in the ZGB algorithm, detailed comparison of the computational efficiency of various algorithms is not straightforward. By introducing real time in the ZGB algorithm and in general in null-event algorithms, a systematic comparison of various CTMC and null-event algorithms is conducted here for two examples, the CO oxidation on Pt and a unimolecular surface reaction. Numerical efficiencies are contrasted with analytic formulas, derived here for the first time, regarding the real time advanced by the null-event algorithm and the most efficient CTMC algorithm. Finally, issues related to accuracy and finite size effects for intermediates found at low concentrations are discussed, and an example from reactions in series is presented.

NULL-EVENT ALGORITHMS

Null-event algorithms depend to some extent on the specific reaction system studied. For the presentation here, we start with the CO oxidation on platinum and follow this by the unimolecular surface reaction $A \rightarrow B$.

Infinitely Fast CO Oxidation Kinetics

CO oxidation is a problem that has received considerable attention. The reaction mechanism considered is

$$\begin{aligned} \mathrm{CO(g)} + {}^* &\rightarrow \mathrm{CO}^* \\ \mathrm{O_2(g)} + 2^* &\rightarrow 2\mathrm{O}^* \\ \mathrm{CO}^* + \mathrm{O}^* &\rightarrow \mathrm{CO_2(g)}, \end{aligned}$$

where * denotes a vacant site or adsorbed species and g denotes a gaseous species. Gaseous CO adsorbs onto the surface occupying a single site. Oxygen adsorbs dissociatively onto the surface, requiring two adjacent sites. The surface reaction between a CO^* and an adjacent O^* is fast, and the product CO_2 desorbs readily from the surface.

The catalyst surface is modeled using a square lattice representing the Pt(100) plane and periodic boundary conditions are employed (note that surface reconstruction is not considered in this work). The algorithm introduced by Ziff and co-workers [1] is as follows. A MC event consists of a microprocess (adsorption of CO or O_2) being randomly selected. The probability of picking the carbon monoxide microprocess is given by the gas-phase mole fraction of carbon monoxide y_{CO}. Similarly, the probability of picking oxygen adsorption is given by the mole fraction of oxygen in the gas-phase, which for a binary mixture (assuming $y_{\mathrm{CO_2}} = 0$) is $(1 - y_{\mathrm{CO}})$. The rate constant of the surface reaction is considered to be infinitely large. Upon selection of a CO adsorption microprocess, a site is selected at random. If the site is occupied, then adsorption is unsuccessful, and the event ends (a null event). If the site is not occupied, the CO molecule adsorbs, and the adjacent sites are randomly checked

for the presence of an O^*. If an O^* is found, the two species react, forming CO_2, which immediately desorbs. Upon selection of an oxygen adsorption event, two adjacent sites are randomly selected. If either site is occupied, the adsorption is unsuccessful and the event ends (a null event). Otherwise, the oxygen dissociatively adsorbs onto the surface, and the adjacent sites are checked for CO^*. If any adjacent site is occupied with CO^*, the two species react, forming CO_2, which immediately desorbs.

Finite CO Oxidation Kinetics

The null-event ZGB algorithm can be extended to finite kinetics. First a lattice site is randomly picked. If the site is empty, adsorption is attempted as discussed above, with a probability that depends on the adsorption rate constants. If the site is occupied, a reaction event is attempted. The execution of the surface reaction follows the same procedure as the dissociative adsorption of O_2; that is, once a site has been chosen, adjacent sites are randomly checked for the complementary surface species. When a CO^*–O^* pair is selected, the event is successful with a probability that depends on the reaction rate constant, k_r.

In our work, results from this algorithm have been compared to CTMC, and excellent agreement was found. In the presence of finite, fast kinetics, i.e., a large k_r compared to the adsorption rate constants, the normalization constant of all probabilities is very high. As a result, many unsuccessful adsorption events occur, resulting in poor efficiency of the null-event algorithm. We will return to this point in a later section, where analytical formulas for the real time advanced by various algorithms are discussed for various cases.

A Unimolecular Surface Reaction

Next we discuss an example of a unimolecular reaction of a single species A transforming into B according to the overall scheme

$$\begin{aligned} A + {}^* &\rightarrow A^* \\ A^* &\rightarrow A + {}^* \\ A^* &\rightarrow B + {}^*. \end{aligned}$$

Attractive adsorbate–adsorbate interactions are considered in the desorption step. The null-event algorithm has been outlined in [19, 20]. In brief, a random number is used to select a lattice site. When the selected site is empty, an adsorption event is attempted with probability proportional to the adsorption rate constant. When the occupied site (central atom) is filled, the existence of local neighbors is examined and the probability of a central atom for reaction and desorption is then computed and compared to a second random number to decide if the event will be executed or not.

Remarks on Null-Event Algorithms

Null-event algorithms have a major disadvantage: they possess events in which nothing occurs. In these algorithms, there is no *a priori* influence of the surface configuration on the selection of a microprocess. Instead, the effect of the surface configuration on the probabilities is implicitly accounted for through the fraction of null events that in turn affects the real time simulated by the algorithm. Since adsorption requires at least one

vacant site to be successful, the fraction of null events increases with decreasing coverage of empty sites, resulting in poor efficiency at low coverage of vacancies. We have found that this inefficiency of algorithms with null events can be a serious impediment when longer real times need to be reached, as happens, for example, when coupling with macroscopic transport equations is attempted in the adjacent fluid phase [41], i.e., within a multiscale computational framework. We discuss this issue more quantitatively below.

An obvious advantage of null-event algorithms is their ease of implementation related to the lack of bookkeeping of neighbor lists. As a result, for conditions where all events are nearly successful, these algorithms could be potentially quite efficient, given the fact that they entail fewer operations per MC event than CTMC algorithms (see also discussion below).

Real Time in Null-Event Algorithms

The original ZGB algorithm lacks real time, which is important for comparison to experimental data and other algorithms. (Here the efficiency of algorithms is mainly compared in terms of real time advanced rather than merely in terms of CPU seconds). Furthermore, computation of reaction rates needed in boundary conditions of larger scale models is impossible because of a lack of real time in null-event algorithms. A connection of MC events with real time was discussed in [44]. Here we propose calculating the average time step based on a single microprocess. By choosing the adsorption of CO, we obtain

$$\Delta t = \frac{1}{\hat{\Gamma}_{a,\mathrm{CO}}}, \tag{2}$$

where $\hat{\Gamma}_{a,\mathrm{CO}}$ is the transition probability of CO adsorption per unit time,

$$\hat{\Gamma}_{a,\mathrm{CO}} = y_{\mathrm{CO}}\Omega_{*}, \tag{3}$$

and Ω_{*} is the number of vacant sites. The time is updated only when a successful CO adsorption is completed, as suggested a few years ago for another surface reaction system [19, 20]. Time is introduced in Eq. (3) through the adsorption rate constant, which here is taken as one, but it can also be computed with real kinetic parameters (see Eq. (4)). A similar approach is followed for the unimolecular surface reaction.

We should note that the time step can also be computed using another microprocess such as the O_2 adsorption process, but this would require scanning the entire surface to compute the transition probability for this microprocess every successful O_2 adsorption event (see transition probabilities in the next section). This would then render the null-event algorithm very inefficient to run. In practice, the microprocess with the shortest time scale, and thus the one most frequently selected, should provide more accurate results regarding the time scales. Processes whose rate is a linear function of coverage, such as the CO adsorption above, should obviously be preferred, as scanning of the surface is not required. We discuss an alternative method, based on an average time step per event, in a subsequent section.

CONTINUOUS-TIME MONTE CARLO ALGORITHMS FOR SURFACE REACTIONS

First, we input the reaction mechanism along with the associated kinetic parameters (i.e., preexponentials and activation energies) in order to compute all transition probabilities of

the reaction mechanism studied. Given the surface temperature, the reaction rate constants are calculated for the adsorption (using the kinetic theory of ideal gases) and the desorption-reaction (using an Arrhenius expression) microprocesses, respectively, as

$$k_a = \frac{s_0 N_A}{C_T \sqrt{2\pi MRT}} \tag{4}$$

$$k_{d,r} = A_{d,r} \exp\left(\frac{-E_{d,r}}{RT}\right), \tag{5}$$

where s_0 is the sticking coefficient for a clean surface, M is the molecular weight of the gaseous adsorbing species, R is the ideal gas constant, N_A is Avogadro's number, C_T is the site density on the surface (sites per unit area), T is the temperature, A is the preexponential, and E is the activation energy.

The algorithms discussed below are a modification of the one originally proposed by Bortz *et al.* for spin prediction of an Ising model, known as the CTMC algorithm or KMC method [36]. We present the general case of surface reaction mechanisms followed by specific examples. Figure 1 provides a schematic of our overall algorithm, which consists of microscopic processes, classes, and sites within a class (tree-type architecture). Table I indicates the possible microprocesses describing the surface kinetics and their respective transition probabilities. To calculate the probabilities P, given in Table I, the concept of classes is introduced. Each surface species (including vacancies) is set in a class according to its microenvironment and thus a class contains these sites, which have exactly the same microenvironment and thus equal transition probability. Furthermore, classes for pairs such

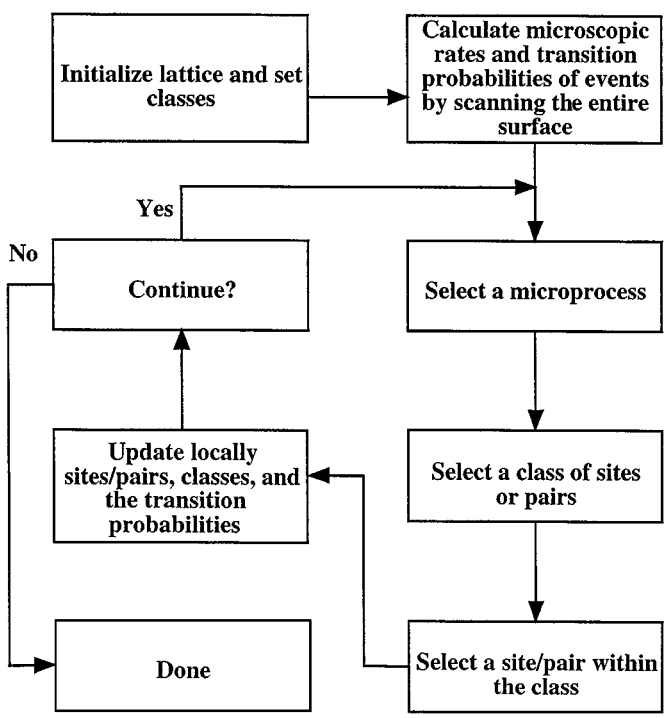

FIG. 1. Schematic of the general proposed CTMC algorithm with lists of neighbors and local update.

TABLE I
Six Common Langmuir–Hinshelwood Microprocesses for Catalytic Surface Reactions and Their Corresponding Transition Probabilities

Microprocesses	Transition probability
Unimolecular adsorption: $A(g) + {}^* \rightarrow A^*$	$\hat{\Gamma} = k[A] \cdot P_*$
Dissociative adsorption: $A_2(g) + 2^* \rightarrow 2A^*$	$\hat{\Gamma} = k[A_2] P_* \cdot P_{*/*}$
Bimolecular surface reaction: $A^* + B^* \rightarrow$ Products	$\hat{\Gamma} = k(P_{A^*} \cdot P_{B^*/A^*} + P_{B^*} \cdot P_{A^*/B^*})$
Unimolecular decomposition: $A^* + {}^* \rightarrow B^* + C^*$	$\hat{\Gamma} = k(P_{A^*} \cdot P_{*/A^*} + P_* \cdot P_{A^*/*})$
Unimolecular desorption: $A^* \rightarrow A(g) + {}^*$	$\hat{\Gamma} = k P_{A^*}$
Associative desorption: $2A^* \rightarrow A_2(g) + 2^*$	$\hat{\Gamma} = k P_{A^*} \cdot P_{A^*/A^*}$

Note. Here P_* is the probability of picking a vacant site, $[A]$ is the gas-phase partial pressure of species A, $P_{x/y}$ is the probability of picking a y site given that an x site was first selected, and k is the reaction rate constant in s^{-1}. The transition probability is on a per site basis per unit time.

as (*,*) for O_2 adsorption and (CO*,O*) for reaction are also considered. Using the transition probabilities for the different classes, the pair probability $P_{\{x,y\}}$ is computed as the product of the probability P_x of choosing site x and the conditional probability $P_{y/x}$ of choosing site y once site x has been picked. The probability of a class involving a single species (e.g., P_x), is just the fraction of sites occupied by this species (often referred to as coverage), i.e.,

$$P_x = \frac{\Omega_x}{\Omega_T}, \tag{6}$$

where Ω_x is the number of sites occupied by x and Ω_T is the total number of catalyst (lattice) sites. For transition probabilities involving two adjacent sites of identity x and y, the number of sites of identity x having i adjacent sites of identity y is denoted as Ω_{xyi} (hereafter termed as size of class xyi). The conditional probability $P_{y/x}$ is given by

$$P_{y/x} = \frac{\sum_{i=1}^{4} i\,(\Omega_{xyi})}{4\Omega_x}. \tag{7}$$

Note that the summation extends over all possible classes found (four on a square lattice assuming either on top or only hollow binding).

The total transition probability per unit time is then given by

$$\hat{\Gamma}_{\text{tot}} = \sum_{i=1}^{n_r} \hat{\Gamma}_i, \tag{8}$$

where n_r is the total number of microprocesses encountered and the transition probabilities $\hat{\Gamma}$ are defined in Table I. Subsequently, the jth microprocess is selected using a random number R_n if

$$\sum_{k=1}^{j-1} \hat{\Gamma}_k < R_n \hat{\Gamma}_{\text{tot}} < \sum_{k=1}^{j} \hat{\Gamma}_k. \tag{9}$$

The choice of a microprocess based on its transition probability ends the first level of the decision tree.

In the second level of the decision tree, a class belonging to the chosen microprocess is selected. For microprocesses requiring one site (e.g., unimolecular adsorption or desorption), there is a single class, except when the microprocess depends on the local microenvironment. Examples of the latter case include an adsorption sticking coefficient which depends on the local coverage and lateral interactions in the desorption step (A^*–A^* interactions only). The transition probability of this step is then corrected by multiplying the transition probability with the lateral interactions' term (demonstrated here for first-nearest neighbors only)

$$I_{xx} = \frac{\sum_{i=0}^{4} \Omega_{xxi} e^{-iw/RT}}{\Omega_x}, \tag{10}$$

where w is the interaction strength (positive for attractive interactions and negative for repulsive interactions). If the microprocess requires two sites, first a class of eligible pairs is selected. The classes are weighted in the same way as they are in the calculation of the reaction rates. This gives a probability of selecting class with m xy pairs as

$$C_{xym} = \frac{m\Omega_{xym}}{\sum_{i=1}^{4}[i\,(\Omega_{xyi} + \Omega_{yxi})]}, \tag{11}$$

and similarly for the yx class of the microprocess. The selection of a class ends the second level of the decision tree.

Following the selection of a class, a site or pair within that class is selected at random and the event is executed (third level of the decision tree). There are two variations of site selection, which exhibit quite different computational efficiencies, and for this reason, we discuss them next. As a first method, the site or pair selection is randomly done over the entire surface; i.e., sites or pairs are randomly selected until the chosen microprocess can be carried out [32–35]. This method resembles the ZGB algorithm with the exception of the elimination of null events. We call this algorithm the *CTMC method without lists.* The second method is to randomly select a site or pair from a list whose number equals the size Ω of the class. The coordinates of sites or pairs of each class are stored in multidimensional matrices, and so upon selection of one site or pair, the identification of the site or pair requires no further computational work [39, 42]. In the case that a class involving a pair is chosen, first a site is randomly selected and then the adjacent sites are checked for the second species by starting at a random neighbor of the first site and proceeding clockwise. When the second species is located, both sites are updated. We call this algorithm the *CTMC with lists.*

Once a site/pair has been selected, the microscopic process is executed, and the corresponding classes are updated. This update can be performed by rescanning the entire surface after every event (*global update*), which is a computationally expensive method. To save on computation time, a *local update* can be implemented, in which only sites affected by the event are updated within the radius of interactions, and the corresponding changes in the transition probabilities are computed to update $\hat{\Gamma}$. As a result, the computation time is practically independent of the lattice size. In this local update, the size of Ω of the altered classes and the stored coordinates of affected sites and/or pairs are also changed. In particular, by modifying Nicholson's idea [45], screening of vectors and of the entire lattice can be avoided. To achieve this, the location and coordinates of a new atom are being added to

the bottom of the list, whereas the location and coordinates of an atom being removed from the list are swapped with those of the last atom of the list.

As a final step in the cycle, the average time elapsed during an event is computed by

$$\Delta t = \frac{1}{\Omega_T \hat{\Gamma}_{\text{tot}}}. \tag{12}$$

The computation proceeds until some criterion is met. Since probabilities are computed *a priori* of a MC event, each trial is in fact successful. We should note that surface diffusion can also be included in the above algorithm in a similar way by considering it as an additional microprocess.

For the specific example of CO oxidation, the rates for the three microprocesses are

CO adsorption:

$$\hat{\Gamma}_{a,\text{CO}} = k_a^{\text{CO}} P_t y_{\text{CO}} P_* \tag{13}$$

O_2 adsorption:

$$\hat{\Gamma}_{a,\text{O}_2} = k_a^{\text{O}_2} P_t y_{\text{O}_2} P_* P_{*/*} \tag{14}$$

Reaction:

$$\hat{\Gamma}_r = k_r \left(P_{\text{CO}^*} P_{\text{O}^*/\text{CO}^*} + P_{\text{O}^*} P_{\text{CO}^*/\text{O}^*} \right). \tag{15}$$

Here P_t is the total gas pressure. Since the surface reaction rate constant is taken to be infinite in the ZGB model, k_r is taken here as a large number. To be consistent with the ZGB algorithm, k_a for both reactants is set equal to one. Note that computation of real time in the null-event algorithm, based on Eq. (14) or Eq. (15), demands knowledge of conditional probabilities that require scanning of the entire lattice, a computationally inefficient process in a null-event algorithm where these probabilities are not computed.

For the class update, the CTMC method without lists can use either a local or a global update. However, to explore the computational differences between algorithms, the local update is used for the CTMC method without lists (otherwise the method becomes too CPU intensive). The computation time of the CTMC method with lists using global update is also compared to show the advantage of using the local update; thus, a total of four algorithms are considered below.

SELF-CONSISTENCY OF ALGORITHMS

First, the proposed local update CTMC algorithm has been compared to literature results to ensure numerical accuracy. Two monomer–monomer systems were considered [15, 16]. In both cases, neither adsorbate–adsorbate interactions nor surface diffusion was considered. In addition, the NO–CO reaction mechanism on a square lattice was compared to the results of [12]. As a final comparision, a dimer–dimer reaction mechanism was considered [18]. In all cases, the agreement with reported results in the literature was very good.

As an example, Fig. 2 shows a reaction isotherm for the CO oxidation by molecular oxygen. The gas-phase concentrations remain constant throughout a simulation and are uniform across the surface. The surface is taken as a uniform, Pt(100) plane, with no

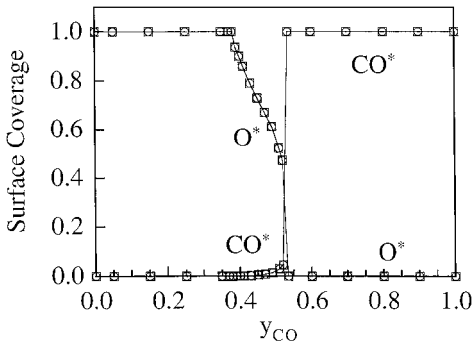

FIG. 2. Reaction isotherm of CO oxidation on a catalytic surface obtained with different algorithms for a lattice size of 160 × 160. Poisoning with O occurs for gas-phase concentrations $y_{CO} < \sim 0.39$ and with CO happens for $y_{CO} > \sim 0.52$. The region between these two limits remains reactive at steady state. The rate constants of adsorption of CO and O_2 are both set to 1. Solid lines: ZGB algorithm; squares: CTMC method with local update and lists; circles: CTMC method without lists. The CTMC with global update is not shown for clarity.

defects. As Ziff and co-workers (and many others) have shown, a reactive region exists for this reaction [1, 6]. The transition from an oxygen-poisoned surface to the reactive region is continuous (a second-order phase transition), whereas the transition from the reactive region to a CO-poisoned surface is discontinuous (a first-order phase transition). Figure 2 indicates that all algorithms predict the same reactive region.

Aside from steady state solutions, time-dependent situations can be a more severe test of different algorithms. As an example, Fig. 3 shows a transient response starting from an empty surface toward the steady state, within the reactive region. There is good agreement between the ZGB and the proposed CTMC algorithms (only the local update with lists is shown for clarity). This reaffirms the accuracy of the time integration using the null-event algorithm.

COMPUTATIONAL EFFICIENCY BY DIRECT NUMERICAL SIMULATIONS FOR A CASE STUDY

To investigate the computational efficiency of the four algorithms, the specific example of CO oxidation is used. All the simulations were performed on a 533-MHz DEC Alpha

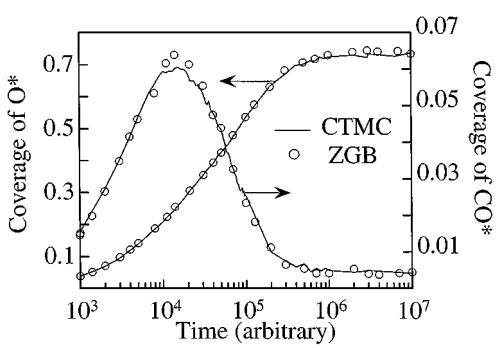

FIG. 3. Transient response of an initially clean surface exposed to a gas-phase concentration of $y_{CO} = 0.45$, which is in the reactive region. The simulation is performed on a 160 × 160 lattice. There is good agreement between the ZGB algorithm and the CTMC method with lists and local update, even for a single run.

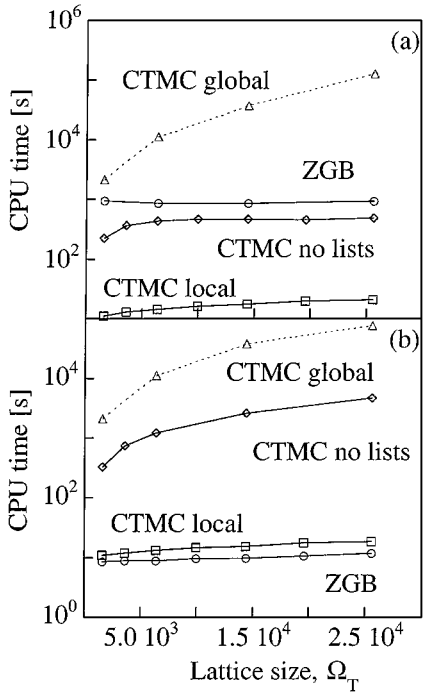

FIG. 4. Comparison of computation times of various algorithms for (a) finite, fast kinetics of $k_r = 100$ and (b) infinitely fast kinetics. For the latter case, the ZGB is the fastest algorithm but it becomes very slow when stiff problems are encountered. The efficiency of the CTMC algorithm with lists and global update depends logarithmically on the lattice size, whereas efficiency of the CTMC algorithm without lists depends on kinetics. The other parameters are those of the ZGB model.

workstation. Figures 4a and 4b plot the CPU time required by the four algorithms to reach the same real time versus lattice size in the case of finite fast kinetics ($k_r = 100$, $k_a P_t = 1$) and infinitely fast kinetics (note that a very large rate constant $O(10^{32})$ was taken in the CTMC algorithm). For these comparison, I/O and initialization are not included. Furthermore, owing to differences in programming between different codes, these numbers should be viewed as reasonable indicators rather than absolute values.

First we discuss scaling laws. Figure 4 shows that in all cases the null-event algorithm and the CTMC with local update algorithm have a weak dependence on the lattice size as no scanning of the lattice is required during a MC simulation (except for an initial scanning). In null-event algorithms, the CPU time required to advance the real time by a certain amount is largely dependent on the percentage of null events performed, which is independent of lattice size. However, the fraction of null events strongly depends upon the value of rate constants, as shown in Fig. 4 and discussed later in this paper. On the other hand, the CTMC with global update algorithm exhibits a logarithmic dependence (when plotted on a log–log scale, a straight line is obtained) on the lattice size due to the scanning of the surface at each event required to update the classes and the associated coordinates for the computation of the transition probabilities. Finally, the CTMC without lists algorithm, which randomly selects sites until the MC event is successful, exhibits an interesting scaling behavior. Specifically, in the case of finite kinetics, the CPU time increases with increasing lattice size and reaches a plateau, whereas in the case of infinitely fast kinetics, it exhibits a logarithmic dependence

on lattice size. As was reported by Jansen *et al.*, for these type of algorithms the lattice size dependence changes from $O(\ln(\Omega_T^{1/2}))$ to $O(\Omega_T)$ when the surface reaction is fast, due to a low probability of finding suitable sites for reaction [34].

Figure 4 indicates that the CPU times of the CTMC algorithm with local and global update remain unaffected by the reaction rate constant as this does not directly affect the execution time of a MC event. In contrast, the performance of the null-event algorithm increases dramatically from finite to infinitely fast kinetics primarily because of the large normalization constant. Despite the absence of null events in the CTMC algorithms, the null-event algorithm is the fastest in the infinitely fast reaction case. However, the null-event algorithm is much slower than the local update with lists CTMC algorithm for finite, fast kinetics as shown in Fig. 4. While null events are eliminated in CTMC algorithms, the execution time, related to the number of operations involved per MC event, is generally higher than that for the null-event algorithm. A detailed analysis of computer speed and memory of three different MC algorithms was given in [46], and so here we only briefly touch on this subject. Without taking into account the initialization of classes and transition probabilities in the CTMC algorithm and neglecting the number of additions and subtractions, the CTMC with lists and with local update code involves about five times more operations and access to many vectors and matrices. Therefore, it is clear that the CTMC with local update code is more demanding to run for the same number of MC events but has the benefit of eliminating null events. Finally, we should remark that while CTMC algorithms provide real time in a transparent way, their efficiency becomes a computational impediment, especially for large lattices, unless lists of neighbors and local update are employed. It also becomes clear why scanning of the lattice for computing the real time in null-event algorithms with nonlinear rates resulting from bimolecular steps, i.e., O_2 adsorption, or lateral interactions, mentioned above, slows computations considerably.

ANALYTIC FORMULAS FOR REAL TIME ADVANCED BY MONTE CARLO ALGORITHMS

The null-event algorithm is very straightforward to implement compared to the CTMC with local update and it is faster per MC event. Therefore, it would be interesting to derive simple criteria allowing one to choose *a priori* between the null-event algorithm and the CTMC with local update. In order to analytically compare their numerical efficiencies, the ratio of the real time advanced by each code for a given number of MC events can be derived at steady state as

$$\frac{t_{\text{null}}}{t_{\text{CTMC}}} = \frac{\tau_{\text{succ}}^{\text{ads}} P_{\text{succ}}^{\text{ads}}}{\tau_{\text{CTMC}}}, \tag{16}$$

where $\tau_{\text{null}}^{\text{ads}}$ and τ_{CTMC} correspond to the average real time advanced per MC event in the null-event algorithm and the CTMC algorithm, respectively, and $P_{\text{succ}}^{\text{ads}}$ is the probability of having a successful adsorption event. τ_{CTMC} is given by Eq. (12) and $\tau_{\text{null}}^{\text{ads}} = 1/\hat{\Gamma}_{a,A}$. The task is then to compute the temporal averaged probabilities, assuming that the time scales are normally distributed with a small standard deviation. This assumption has been verified via direct numerical simulations. Note that here the time advanced in null-event algorithms follows [19, 20] and differs from the conventional way of incrementing the

time step by the inverse of the lattice size after each event. The latter method is shown below to give equivalent results only for the ZGB algorithm. Next we discuss selected examples.

A Unimolecular Surface Reaction

We start with the unimolecular reaction as the first example. The probability of having a successful event, which corresponds to the sum of the probabilities of each microprocess, is

$$P_{\text{succ}} = \frac{k_a P_t y_A}{k_{\max}} P_* + \frac{k_r}{k_{\max}}(1 - P_*) + \frac{k_d}{k_{\max}} \frac{\sum_{i=0}^{4} \Omega_{AAi} e^{-iw/RT}}{\Omega_T}, \tag{17}$$

where $k_{\max}$ is the normalization constant of all probabilities. From the steady state mass balance we can deduce that $P_{\text{succ}}^{\text{ads}} = P_{\text{succ}}^{\text{des}} + P_{\text{succ}}^{\text{rxn}} = P_{\text{succ}}/2$. Combining all this information, we obtain expressions for the average real time of each algorithm,

$$P_{\text{succ}}^{\text{ads}} \tau_{\text{succ}}^{\text{ads}} = \frac{k_a P_t y_A P_* + k_r(1 - P_*) + k_d \frac{\sum_{i=0}^{4} \Omega_{AAi} e^{-iw/RT}}{\Omega_T}}{2k_{\max} k_a P_t y_A \Omega_*} = \frac{1}{k_{\max} \Omega_T} \tag{18}$$

$$\tau_{\text{CTMC}} = \frac{1}{k_a P_t y_A \Omega_* + k_r(\Omega_T - \Omega_*) + k_d \sum_{i=0}^{4} \Omega_{AAi} e^{-iw/RT}}, \tag{19}$$

leading to the following general expression for the ratio of the real times:

$$\frac{t_{\text{null}}}{t_{\text{CTMC}}} = \frac{\left(k_a P_t y_A P_* + k_r(1 - P_*) + k_d \frac{\sum_{i=1}^{4} \Omega_{AAi} e^{-iw/RT}}{\Omega_T}\right)^2}{2k_{\max} k_a P_t y_A P_*} = \frac{2k_a P_t y_A P_*}{k_{\max}}. \tag{20}$$

Note that in Eqs. (18)–(20), P_* and Ω_{AAi} are temporal averages. Very good agreement (up to a few percent deviation) is observed between the ratio obtained from direct numerical simulations and Eq. (20) for all simulations conducted.

To assess the computational efficiency issue, limiting cases were next studied. Indeed, if one of the rate constants is much larger than the other two (in the limit of $w = 0$), Eq. (20) can be simplified. For example, if $k_a P_t y_A \gg k_{d,r}$ then the surface will be almost completely covered by A^* at steady state. Using the mass balance, one can express the vacant site coverage to obtain the following simple expression (assuming $k_{\max} = k_a P_t y_A$):

$$\frac{t_{\text{null}}}{t_{\text{CTMC}}} = 2P_* = \frac{2(k_r + k_d)}{k_a P_t y_A}. \tag{21}$$

Similarly when $k_{d,r} \gg k_a P_t y_A$ one can write (assuming $k_{\max} = k_{d,r}$)

$$\frac{t_{\text{null}}}{t_{\text{CTMC}}} = \frac{2k_a P_t y_A}{k_{d,r}}. \tag{22}$$

Equations (21) and (22) compare well to Eq. (20) and, thus, to the numerical simulations. We can therefore reach a simple conclusion that the null-event algorithm becomes very inefficient, by several orders of magnitude, when there is disparity in rate constants. This

general conclusion drawn for a unimolecular reaction seems, however, to contradict the results for CO oxidation shown in Fig. 4, where an infinitely fast reaction was modeled. This issue will be clarified below.

Another case is when attractive interactions play a role in the efficiency of the two algorithms. Indeed, in the case where $k_d \gg k_a P_t y_a$ but $k_d e^{-4w/RT} \ll k_a P_t y_A$ and the reaction is slow, the surface becomes almost completely covered and we have the following simplification of Eq. (20):

$$\frac{t_{\text{null}}}{t_{\text{CTMC}}} = 2e^{-4w/RT}. \tag{23}$$

Equation (23) implies that the stronger the attractive interaction, the less efficient the null-event algorithm becomes.

Front Tracking for Moving Boundaries

We should note that the CTMC algorithm with lists and local update represents a front tracking technique; i.e., successful events occur only in the neighborhood of the moving interface. As an example, a moving boundary problem was analyzed by starting with a half-covered and a half-empty surface when a phase transition occurs, leading to a fully covered surface in the case of a unimolecular reaction. To reach steady state, the null-event algorithm needs two orders of magnitude more MC events than the CTMC with list and local update. This result is close to the steady state efficiency predicted by Eq. (20) for the specific parameters chosen.

Infinitely Fast CO Oxidation Kinetics

Following the approach outlined above, a similar analytic formula can be derived for the catalytic oxidation of CO described by Eqs. (13)–(15), using the mass balance at steady state $P_{\text{succ}}^{\text{ads}_{\text{CO}}} = 2P_{\text{succ}}^{\text{ads}_{\text{O}_2}} = P_{\text{succ}}^{\text{rxn}}$:

$$P_{\text{succ}}^{\text{ads}_{\text{CO}}} \tau_{\text{succ}}^{\text{ads}_{\text{CO}}} = \frac{2}{3} \frac{y_{\text{CO}} P_* + y_{\text{O}_2} P_* P_{*/*}}{y_{\text{CO}} \Omega_*} = \frac{1}{\Omega_T} \tag{24}$$

$$\tau_{\text{CTMC}} = \frac{2}{5} \frac{1}{y_{\text{CO}} \Omega_*}. \tag{25}$$

We then obtain the simple relation

$$\frac{t_{\text{null}}}{t_{\text{CTMC}}} = \frac{5 y_{\text{CO}} P_*}{2}. \tag{26}$$

For this specific case, the real time advanced by the original ZGB algorithm (see Eq. (24)) depends only on the lattice size. By comparison to Eq. (18), it is clear though that this result is problem specific (true only when $k_{\max} = 1$). It is our experience that $k_{\max}$ is not always easy to predict for complex kinetics with either long-range interactions or multicomponent interactions manifested in density functional theory simulations. For this reason, we propose that a $k_{\max}$ adaptive scheme could be used for optimal code performance while taking into account rare events with very large $k_{\max}$. Furthermore, at a given CO mole fraction, the efficiency of the two codes depends merely on the coverage of empty sites, which in this

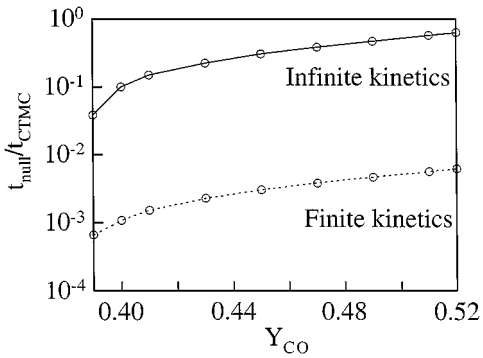

FIG. 5. Ratio of real times advanced by the null event algorithm and the local update with lists CTMC algorithm vs mole fraction of CO in the reactive regime for (a) finite, fast kinetics ($k_r = 100$) and (b) infinitely fast kinetics. The other parameters are those of the ZGB model.

example is bounded to relatively large numbers. Figure 5 shows the ratio of times within the reactive region computed using Eq. (26) and the coverage of vacancies obtained from direct numerical simulations. In this zone, the real time advanced by the null-event algorithm can be up to about one order of magnitude smaller than that of the local update with lists CTMC algorithm. This outperformance of the CTMC algorithm is compensated by its lower speed per MC event, as the results in Fig. 4 indicate. As noted above, it is interesting to note that despite the stiffness of this example (infinitely fast kinetics), the null-event algorithm performs very well. This is rationalized next.

Finite CO Oxidation Kinetics

The above derivation can be easily extended to take into account finite reaction kinetics in the CO oxidation. The ratio of times between the null-event algorithm and the CTMC with local update algorithm then becomes

$$\frac{t_{\text{null}}}{t_{\text{CTMC}}} = \frac{5 y_{\text{CO}} P_*}{2(k_r + 2)}. \tag{27}$$

Equation (27) indicates that for finite, fast kinetics, the null-event algorithm becomes computationally very demanding compared to the infinitely fast kinetics as k_r becomes large. This is in fact shown for a set of parameters in Fig. 5 and is consistent with the observations made in Fig. 4. The apparent paradox in computational efficiency of the null-event algorithm shown in Fig. 4 appears then to be caused by the specific (optimal) implementation of the original ZGB algorithm, where an adsorption step is often immediately followed by a surface reaction step. This implementation (a hybrid null-event algorithm) removes the stiffness of the equations. A similar situation arises when an infinite Fickian diffusion is modeled by randomly redistributing all atoms on the lattice in each MC event.

The cases studied above show that the null-event algorithm becomes very inefficient when the problem is stiff, i.e., when widely varying time scales are involved. Analytic formulas, such as Eqs. (20), (25), and (27), allow a quick study of the efficiency of the null-event algorithm over the CTMC approach. Indeed, even though the null-event algorithm is very straightforward to implement, it is not the best choice for stiff problems (rare event dynamics).

COMPUTATIONAL ACCURACY

Another topic of interest is the accuracy of MC algorithms in predicting (i) converged steady state solutions and (ii) low concentrations of surface species such as radicals. For the latter case, although radical species, such as OH^*, exist at very low surface concentrations, they are crucial intermediates in the overall mechanism (see H_2 oxidation chemistry for an example [47]). Given the small size of a lattice, it is not readily apparent whether such low concentrations and corresponding reaction rates can be accurately captured by time averages of MC simulations.

To exploit these issues, a simple system of three reacting surface species is considered. The mechanism and corresponding rate constants are as follows:

$$A + * \underset{k_{-1}}{\overset{k_1}{\leftrightarrow}} A^*$$

$$B + * \underset{k_{-2}}{\overset{k_2}{\leftrightarrow}} B^*$$

$$C + * \underset{k_{-3}}{\overset{k_3}{\leftrightarrow}} C^*$$

$$A^* \xrightarrow{k_4} B^*$$

$$B^* \xrightarrow{k_5} C^*.$$

Since the rate expressions for this system are all linear (they all involve a single lattice site), the results obtain from MC simulations can be compared to the exact values obtained using a MF model (a model system whose behavior is analytically predictable). This enables us to test the accuracy in computing surface coverages and reaction rates by MC simulations and to discuss implications for nonlinear kinetics of practical interest.

As a first example, the transient response of an initially clean surface was explored. The kinetic parameters are chosen so that consumption of B and desorption of C are relatively fast, resulting in low coverages. Figure 6 shows that MC results from a single run are inaccurate at low surface concentrations, in this case, at short times. Interestingly enough,

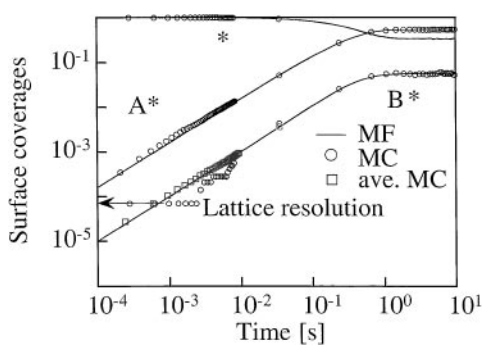

FIG. 6. Transient response of a clean surface toward steady state for a linear reaction mechanism and a lattice size of 120 × 120. The small lattice limits the accuracy of a single MC simulation for surface coverages below the lattice resolution. Circles correspond to MC data, squares correspond to temporal averaged (over 100 independent runs) MC data for the surface coverage of B^*, and solid lines correspond to the MF model. The parameters for this simulation are $k_1 = 2$, $k_{-1} = 0.01$, $k_2 = 1$, $k_{-2} = 0.02$, $k_3 = 3$, $k_{-3} = 10$, $k_4 = 1$, $k_5 = 10$, $y_A = 0.8$, $y_B = 0.1$, and $y_C = 0.1$.

deviations at short times of a single run do not propagate to longer times, as usually happens with integration of deterministic ordinary differential equations. Using a single simulation, concentrations above the lattice resolution approach the MF values. However, the deviations below the resolution limits of the small lattice require substantially more runs ($O(10^2)$) for the average to approach the MF value as shown with the squares in Fig. 6.

The effect of limited lattice size has been further exploited for a second set of conditions, where all rate constants are fixed except k_5, which ranges from 10^{-2} to 10^6. By increasing k_5, the surface concentration of B^* decreases, and at high values of the reaction rate constant, species B^* is in quasi-steady state and its concentration becomes very low (the number of B^* molecules basically varies between 0 and 1 in various surface snapshots). Under such conditions, the rate of reaction also varies between zero and a finite value, showing large fluctuations. We have previously shown that stochastic effects can cause significant deviations in results of small size systems from continuum conservation equations [42]. We further explore this issue for lattice MC next, when the species concentrations fall below the resolution threshold.

One method of calculating rates from MC simulations is to evaluate an arithmetic mean of instantaneous transition probabilities, given by

$$\langle \hat{\Gamma}_i \rangle = \frac{\sum_{k=1}^{n} \hat{\Gamma}_i(k)}{n}, \tag{28}$$

where n is the number of snapshots used. However, under conditions where large fluctuations in rates occur between consecutive events, the transition probabilities and, thus Δt, vary significantly. It is therefore expected that Eq. (28) will not provide accurate estimates of rates. Another method of calculating rates is to use a time-weighted average of the transition probabilities, given by

$$\langle \hat{\Gamma}_i \rangle = \frac{\sum_{k=1}^{n} \frac{\int_{\tau_k} \hat{\Gamma}_i \, dt}{\tau_k}}{n} = \frac{\sum_{k=1}^{n} \frac{M_i(k)}{\tau_k}}{n}, \tag{29}$$

where $M_i(k)$ is the number of events of microprocess of type i occurring over time τ_k (easily counted in a simulation) and n is now the number of intervals over which an average rate is computed. If the time step Δt is constant, then Eq. (29) reduces to Eq. (28). Figure 7a indicates that as k_5 increases, Eq. (28) becomes inaccurate, whereas Eq. (29) remains accurate. This fact can be exploited to calculate the surface coverage of intermediate species at low coverages. Using the rate calculated from Eq. (29), the surface coverage of B^* can be determined from

$$\langle \theta_B \rangle = \frac{\langle \hat{\Gamma}_5 \rangle}{k_5} \tag{30a}$$

compared to a simple arithmetic mean

$$\langle \theta_B \rangle = \frac{\sum_{k=1}^{n} \theta_B(k)}{n}. \tag{30b}$$

Figure 7b shows a comparison of the different methods for calculating the low concentration of the intermediate species. Using the method proposed above (Eq. (30a)), the concentration of B^* can be determined to a good degree of certainty, even from a single run.

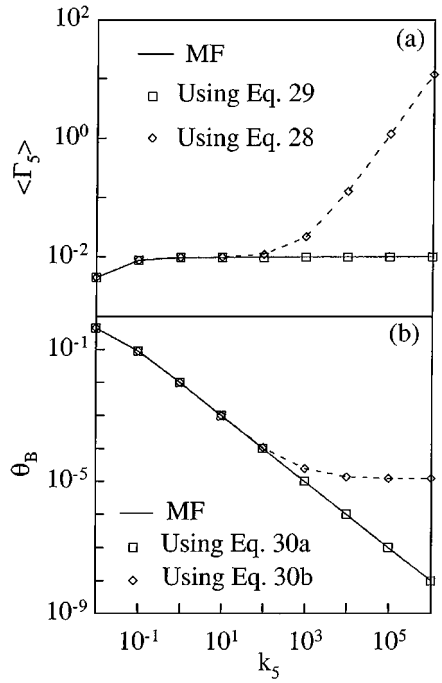

FIG. 7. Comparison of average transition probabilities (a) and coverage of intermediate species (b) versus the reaction rate constant k_5. The parameters for this simulation are $k_1 = 2$, $k_{-1} = 0.01$, $k_2 = 0.01$, $k_{-2} = 0.002$, $k_3 = 100$, $k_{-3} = 1$, $k_4 = 0.01$, $y_A = 1$, $y_B = 0.0$, and $y_C = 0.0$. The time-weighted average transition probabilities provide a more accurate method of calculation compared to the arithmetic mean when large fluctuations in transition probabilities occur between events. Despite a resolution threshold caused by finite lattice sizes, the concentrations of key intermediates can accurately be computed (see text).

As another example, the CO oxidation problem was considered. For the adsorption rates of CO and O_2, the agreement between Eq. (28) and Eq. (29) is good (0.001% for CO adsorption and 0.8% for oxygen adsorption). However, for the surface reaction rate, Eq. (28) resulted in large errors due to the huge reaction rate constant of the surface reaction.

For the example of CO oxidation, the convergence of steady state equations was also examined. At steady state, the rates involving a surface species multiplied by its stoichiometric coefficients (the residual) should be equal to zero. For CO^*, the normalized residual is

$$\frac{|\langle\hat{\Gamma}_{CO}\rangle - \langle\hat{\Gamma}_R\rangle|}{\langle\hat{\Gamma}_{CO}\rangle} \times 100\%. \tag{31}$$

The normalized residual for oxygen is

$$\frac{\left|2\langle\hat{\Gamma}_{O_2}\rangle - \langle\hat{\Gamma}_R\rangle\right|}{\langle\hat{\Gamma}_{O_2}\rangle} \times 100\%. \tag{32}$$

Using the rates obtained from Eq. (29), the steady state residuals are less than 1%. Such numbers are typically larger than the corresponding counterparts of continuum type diffusion–reaction solvers based on Newton's technique. While better convergence is possible, prohibitively large lattices or parallelization may be required to achieve this.

CONCLUSIONS

We have presented a general, efficient CTMC algorithm, applicable to complex surface reaction mechanisms, and discussed two additional, frequently used ones. Using the example of the CO oxidation, the computational efficiency of various algorithms was compared for the same integration time rather than the same number of MC events. Simple formulas were also derived to compare real times between the null-event algorithm and the CTMC with local update algorithm for a unimolecular reaction and the CO oxidation reaction. Finally, accuracy issues related to lattice size resolution were addressed. The main conclusions are summarized as follows:

- Using a single microprocess, such as CO adsorption, real time can be introduced into the ZGB algorithm and null-event algorithms in general.
- Null-event algorithms can become extremely inefficient compared to the CTMC algorithm with lists and local update when stiff problems are encountered.
- These two algorithms show a slight dependence on the lattice size compared to the CTMC with global update or the CTMC without lists algorithm. This is an interesting feature if larger lattice size simulations are required to obtain accurate solutions.
- Using time-weighted averages, the reaction rates and low surface coverages (below the resolution of the lattice size) can be computed accurately; on the other hand, simple arithmetic mean estimations can lead to wrong results in coverage using a single run. This behavior indicates that application of CTMC algorithms to realistic reaction schemes, where concentrations of intermediates are low, is feasible.

The computational efficiency of the proposed algorithm makes it an attractive alternative to other available algorithms. Aside from its speed, it can also be readily adapted to arbitrarily complex reaction mechanisms. This versatility can make it a more desirable choice than null-event algorithms.

ACKNOWLEDGMENTS

This work was supported in part by National Science Foundation Career Award under Grants CTS-9702615 and CTS-9904242, and by NETI. Insightful comments by Professor R. M. Ziff are also acknowledged.

REFERENCES

1. R. M. Ziff, E. Gulari, and Y. Barshad, *Phys. Rev. Lett.* **56**(24), 2553 (1986).
2. E. V. Albano, *Phys. Rev. E* **57**(6), 6840 (1998).
3. J. Cortés, E. Valencia, and P. Araya, *J. Chem. Phys.* **109**(13), 5607 (1998).
4. V. P. Zhdanov and B. Kasemo, *Phys. Rev. B* **55**(7), 4105 (1997).
5. F. Bagnoli, B. Sente, M. Dumont, and R. Dagonnier, *J. Chem. Phys.* **94**(1), 777 (1991).
6. C. A. Voigt and R. M. Ziff, *Phys. Rev. E* **56**(6), R6241 (1997).
7. D. ben-Avraham, D. Considine, P. Meakin, S. Redner, and H. Takayasu, *J. Phys. A: Math. Gen.* **23**, 4297 (1990).
8. J. W. Evans and T. R. Ray, *Phys. Rev. E* **50**(6), 4302 (1994).
9. J. W. Evans and M. S. Miesch, *Surf. Sci.* **245**, 401 (1991).
10. J. W. Evans, *J. Chem. Phys.* **98**(1), 2463 (1993).
11. C. A. Voigt and R. M. Ziff, *J. Chem. Phys.* **107**(8), 7397 (1997).
12. J. Cortés and E. Valencia, *Chem. Phys.* **229**, 265 (1998).

13. X. Y. Guo, B. Zhong, and S. Y. Peng, *Chem. Phys. Lett.* **233**, 580 (1995).
14. K. Yaldram and M. A. Khan, *J. Catal.* **131**, 369 (1991).
15. L. Cortés, H. Puschmann, and E. Valencia, *J. Chem. Phys.* **106**(4), 1467 (1997).
16. E. V. Albano, *Phys. Rev. Lett.* **69**(4), 656 (1992).
17. F. J. Williams, C. M. Aldao, A. Palermo, and R. M. Lambert, *Surf. Sci.* **412/413**, 174 (1998).
18. E. V. Albano, *J. Phys. A: Math. Gen.* **25**, 2557 (1992).
19. D. G. Vlachos, L. D. Schmidt, and R. Aris, *J. Chem. Phys.* **93**, 8306 (1990).
20. D. G. Vlachos, L. D. Schmidt, and R. Aris, *Surf. Sci.* **249**, 248 (1991).
21. J. W. Evans, *Rev. Mod. Phys.* **65**(4), 1281 (1993).
22. V. P. Zhdanov and B. Kasemo, *Surf. Sci. Rep.* **20**, 111 (1994).
23. V. Zhdanov and B. Kasemo, *Surf. Sci. Rep.* **39**(2–4), 29 (2000).
24. H. C. Kang and W. H. Weinberg, *Chem. Rev.* **95**(3), 667 (1995).
25. A. Jansen and J. Lukkien, *Catal. Today* **53**, 259 (1999).
26. S. J. Lombardo and A. T. Bell, *Surf. Sci. Rep.* **13**(1–2), 1 (1991).
27. R. Nieminen and A. Jansen, *Appl. Catal. A: Gen.* **160**, 99 (1997).
28. K. A. Fichthorn and W. H. Weinberg, *J. Chem. Phys.* **95**(2), 1090 (1991).
29. F. Qin, L. Tagliabule, L. Pivesan, and E. Wolf, *Chem. Eng. Sci.* **53**(5), 919 (1998).
30. P. Dufour, M. Dumont, V. Chabart, and J. Lion, *Comput. Chem.* **13**(1), 25 (1989).
31. I. Jensen, H. Fogedby, and R. Dickman, *Phys. Rev. A* **41**(6), 3411 (1990).
32. A. P. J. Jansen, *Phys. Rev. B* **52**(7), 5400 (1995).
33. A. P. J. Jansen, *Comput. Phys. Commun.* **86**, 1 (1995).
34. A. P. J. Jansen and R. M. Nieminen, *J. Chem. Phys.* **106**(5), 2038 (1997).
35. R. Kissel-Osterrieder, F. Behrendt, and J. Warnatz, in *Twenty-Seventh Symposium (International) on Combustion* (The Combustion Institute, Pittsburg, 1998), p. 2267.
36. A. B. Bortz, M. H. Kalos, and J. L. Lebowitz, *J. Comput. Phys.* **17**, 10 (1975).
37. D. T. Gillespie, *J. Comput. Phys.* **22**, 403 (1976).
38. D. T. Gillespie, *J. Phys. Chem.* **81**, 2340 (1977).
39. D. G. Vlachos, L. D. Schmidt, and R. Aris, *Phys. Rev. B* **47**, 4896 (1993).
40. N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, *J. Chem. Phys.* **21**, 1087 (1953).
41. D. G. Vlachos, *AIChE J.* **43**(11), 3031 (1997).
42. D. G. Vlachos, *Chem. Eng. Sci.* **53**(1), 157 (1998).
43. M. E. Coltrin, R. J. Kee, and F. M. Rupley, *Surface Chemkin (Version 4.0): A FORTRAN Package for Analyzing Heterogeneous Chemical Kinetics at a Solid-Surface-Gas-Phase Interface* (Sandia National Labs, Albuquerque, NM, 1990).
44. D. Sholl and R. Skodje, *Surf. Sci.* **334**, 295 (1995).
45. D. Nicholson, *CCP5 Quart.* **11**, 19 (1984).
46. J. Lukkien, J. Segers, P. Hilbers, R. Gelten, and A. Jansen, *Phys. Rev. E* **58**(2), 2598 (1998).
47. P. A. Bui, D. G. Vlachos, and P. R. Westmoreland, *Ind. Eng. Chem. Res.* **36**(7), 2558 (1997).