SQERTSS: Dynamic Rank Based Throttling of Transition Probabilities in Kinetic Monte Carlo Simulations

Thomas Danielson[1], Jonathan Sutton,[3] Celine Hin[1,2], Aditya Savara[3]
Corresponding Author: Aditya Savara, savaraa@ornl.gov
[1]*Department of Materials Science and Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA, USA*
[2]*Department of Mechanical Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA, USA*
[3]*Chemical Sciences Division, Oak Ridge National Laboratory, Oak Ridge, TN, USA*

ABSTRACT

Lattice based Kinetic Monte Carlo (KMC) simulations offer a powerful simulation technique for investigating large reaction networks while retaining spatial configuration information, unlike ordinary differential equations. However, large chemical reaction networks can contain reaction processes with rates spanning multiple orders of magnitude. This can lead to the problem of “KMC stiffness” (similar to stiffness in differential equations), where the computational expense has the potential to be overwhelmed by very short time-steps during KMC simulations, with the simulation spending an inordinate amount of KMC steps / cpu-time simulating fast frivolous processes (FFPs) without progressing the system (reaction network). In order to achieve simulation times that are experimentally relevant or desired for predictions, a dynamic throttling algorithm involving separation of the processes into speed-ranks based on event frequencies has been designed and implemented with the intent of decreasing the probability of FFP events, and increasing the probability of slow process events -- allowing rate limiting events to become more likely to be observed in KMC simulations. This Staggered Quasi-Equilibrium Rank-based Throttling for Steady-state (SQERTSS) algorithm designed for use in achieving and simulating steady-state conditions in KMC simulations. As shown in this work, the SQERTSS algorithm also works for transient conditions: the correct configuration space and final state will still be achieved if the required assumptions are not violated, with the caveat that the sizes of the time-steps may be distorted during the transient period.

1. Introduction

Lattice based kinetic Monte Carlo (KMC) serves as a powerful and widely used computational technique for investigating the stochastic spatio-temporal evolution of atomic and molecular systems, e.g. catalytic chemical reactions on crystalline surfaces [1-11]. The technique transitions a system from an initial configuration to a subsequent configuration, where the subsequent configuration is selected using a random number and a probability weighted by the available transition rate constants (for chemical reactions, the transition rate constants are proportional to the reaction rate constants). By comparison to solutions of ordinary differential equations, lattice based KMC provides the advantage of being able to obtain localized, site-specific reaction information. However, the computational expense of lattice based KMC has the potential to become intractable when the event frequencies of the different reaction events (e.g. diffusion, dissociation, association, adsorption, desorption, etc.) span many orders of magnitude, in which case the KMC steps are dominated by events of fast frivolous processes (FFPs). As a result, using KMC to simulate experimentally relevant simulation times (~10^4 s) for complex chemical reaction networks poses a challenge, since the simulation may be prone to spending many steps simulating FFPs with very small time increments (i.e. similar to stiffness in ODEs). An FFP is defined here as a rapid process which does not significantly progress the reaction network towards a steady-state (e.g. H^+ hopping back and forth between neighboring O^{2-} atoms). One example from the literature where the local site-specific information may be required to capture the kinetics but FFPs would prevent the KMC simulation from being tractable is Fischer-Tropsch over Ru, for which there is evidence that there are both spatio-temporal effects in the kinetics as well as pre-equilibrium reaction steps.[12] The FFPs that are mediated in this work are a subset of the quasi-equilibrated (QE) processes present in a simulation: we call a process QE when the forward and reverse rates are equal or approximately equal, and thus what is referred to as QE in this work may not be reflective of a Boltzmann distribution across the states present in the system (we do not attempt to identify whether a QE process is actually at equilibrium -- what we refer to as QE does include processes that are at equilibrium). Additionally, various processes have significantly lower transition probabilities, making them rarely occurring events. These rare events may be important for transitioning the reaction network to a steady-state; however, the occurrence of rare events may require a substantial number of KMC steps to achieve even a single rare event occurrence (particularly when FFPs are present).

Various formulations have been developed to address the issue of simulating long time scales, relative to the characteristic time-scales of processes being simulated, when using KMC. Some methods treat the occurrence of a FFP as a localized energy basin [13-18]. In “basin” methods, the probabilities of the system evolving to a state outside of the basin, and the respective escape time for different paths out of the energy basin, are calculated. In many cases, such an approach is considered to be exact in preserving the underlying kinetics of the system. An alternative approach is the τ-leap method [19-24]. Within the τ-leap approach, reactions are executed several times and the simulation time is advanced by coarser time increments than the standard microscopic time-step. While these approaches are suitable for simple systems (e.g. diffusion of point defects), their utility for investigating a complex chemical reaction network on a lattice is limited by needing an ability to predict the configurations which may occur outside of the energy basin. In general, it is not possible to predict which of these configurations would be accessible *a priori* (especially for rare configurations). Likewise, the computational expense of the basin hopping formulations scales increasingly with the number of processes and possible outcomes. Thus, for simulations containing many reaction processes and many possible configurations, an alternative method is needed, and must be able to simulate long time scales in an efficient manner. The approach we use below is chosen specifically for addressing this problem in the context of complex chemical reaction networks with FFPs, though it may be applicable to other systems where basin hopping is not feasible for the same reasons.

We introduce an alternative approach: Staggered Quasi-Equilibrium Rank-based Throttling for Steady-State (SQERTSS), which circumvents the complications of FFPs, rare events, and rare configurations by dynamically throttling the transition probabilities of reaction processes based on the local-in-time KMC event frequencies. In this work, throttling refers to speeding up or slowing down a particular process (in the context of KMC, this means increasing/decreasing the event frequency by changing the associated transition rate constant). The dynamic throttling of transition probabilities is similar to solving stiff ODEs by combining the Quasi Steady-State Approximation [25-27] and dynamic time-step methods[28] which assume a separation of time scales (classified by speed-ranking). In this work, we use the stiffness mediation in the context of an asymptotic approach to steady-state for various reactions on the time-scale of interest. By applying the SQERTSS approach, FFPs become less likely to occur and slower events more likely to occur, resulting in a faster approach to a steady-state of the reaction network while maintaining kinetically appropriate ratios between transition probabilities when computationally feasible. This approach also enables appropriate timescales to be simulated once steady-state is reached; in a way where the accuracy achieved is based on the computational resources available (a paragraph on this philosophy is provided in the supporting information). A similar method is net event KMC, [29] in which fast reversible processes are lumped together as one and the net rate is determined by the difference in the forward and reverse process. While net event KMC successfully improves the computational expense of KMC simulations while maintaining the relative probabilities of occurrence between forward and reverse processes, it does not maintain the staggering of the event frequencies between processes of different speed-ranks (described in sections 3 and 4); thus net event KMC (which is different from SQERTSS) can affect the paths accessed and consequently the chemical selectivity in complex reaction networks. The SQERTSS methodology introduced here is designed to retain the chemical selectivity for complex chemical reaction networks, even for many cases where net event KMC would not.

After describing the methodology, we present results where the algorithm has been applied to sample reaction networks with physically realistic parameters. We find that this algorithm behaves as anticipated, and is practical.

2. Computational Methods and Model Details

The reaction network has been investigated using lattice Kinetic Monte Carlo (KMC) as implemented in the program KMOS, which was developed by Hoffman and Reuter as a general purpose code for lattice kinetic Monte Carlo simulations encompassing multiple chemical reactions [30]. The current section will outline the Lattice KMC framework that has been used in this work, followed by a description of the reaction network that has been used for testing.

2.1. Lattice KMC Formulation

The KMC formulation will be discussed in the context of a catalytic chemical reaction on a crystal surface where there exists a set of well-defined unique reaction sites (e.g. atomic or molecular scale surface sites). The simulation cell of the crystal surface is repeated through space with the use of periodic boundary conditions, and each surface site has the potential of becoming occupied by chemical species. The surface, and the associated species on the surface, define a surface configuration at a particular point in time. The probability (and time-steps) for transitions between surface configurations are related to the transition rate constants associated with each possible reaction process. Kinetically, the stochastic evolution of the surface configuration through time follows a Markovian master equation:

$$\dot{\rho}_u(t) = \sum_v (w_{uv}\,\rho_v(t) - w_{vu}\rho_u(t)) \tag{1}$$

where, $\rho_u(t)$ is the probability for the system to be in configuration u at time t, and w_{vu} is the transition rate constant which represents the mean frequency ("rate") at which configuration u is expected to transition to configuration v in the absence of other possible transitions. The KMC algorithm generates an ensemble of possible trajectories that can propagate the surface to other configurations across time, such that the average over the entire ensemble of trajectories yields the probability densities of the underlying Markovian master equation.

Within the current work, the variable step-size method (VSSM) has been used for the propagation of the system through different surface configurations across time. The time-steps in VSSM emerge from a basis that the waiting times for n uncorrelated events that occur with transition rate, k, follow a Poisson distribution as:

$$p_n(k, \Delta t) = \frac{(w\Delta t)^n e^{-w\Delta t}}{n!} \tag{2}$$

From this, the waiting time between two events is given by the case that no events occur:

$$p_o(w, \Delta t) = e^{-w\Delta t} \tag{3}$$

Using a uniform random number, $r_t \in (0,1]$:

$$\Delta t = \frac{-\ln(r_t)}{w_{total}} \tag{4}$$

Where, w_{total} is the sum of all transition rate constants for all processes that can occur within the current configuration (for a more thorough derivation of VSSM see refs [30-34]). Thus, the step-size is variable due to the fact that the number of processes available to occur in each configuration is variable, and also due to the random number in Equation 4. Within VSSM, the event which occurs in a given KMC step is also selected using a separate random number, r_w, and the sum of all available process transition rates as:

$$\sum_v^{j-1} w_{vu} < r_w w_{tot,u} \leq \sum_v^{j} w_{vu} \tag{5}$$

The most relevant KMC output when investigating the rates of reactions in a simulation are the process event frequencies. The event frequency (i.e. the number of times an event happens per unit cell per second) for a given process can be calculated across a given time interval as:

$$TOF_a = \frac{M}{\Delta t} \tag{6}$$

where M is the number of times process a has occurred during the time interval Δt -- for example, this could correspond to the number of times a molecule is produced for a particular species such as CO or CH_3OH. Typically, an event frequency is calculated based on the observations from multiple (e.g., thousands of) KMC steps, and these groupings of steps can be called 'snapshots'.

To summarize, each step in a VSSM KMC simulation can be stated as follows:

1. Initialize lattice if at the first step, otherwise take lattice configuration from previous step.
2. Update list of available processes and their associated transition rates to calculate $k_{total} = \sum k$
3. Draw two random numbers on the interval (0,1]
4. Select an event that satisfies equation 5
5. Update lattice configuration according to the event which occurs
6. Update time according to equation 4

2.2 Reaction Model: Methanol Adsorption and Conversion on CeO_2 (111)

The following section will outline the reaction network that has been simulated using KMC in order to test the dynamic throttling algorithm. The reaction network is a simplified reaction network from one in the literature,[35, 36] that begins with methanol adsorption on a CeO_2 (111) surface and results in output fluxes of formaldehyde, H_2 and methanol through various reaction processes.

Under the conditions which will be simulated, there is initially only an input flux of methanol gas molecules encountering a defect free CeO_2 (111) surface. The CeO_2 (111) surface has three distinct site types: Ce, O and an intermediate bridge site for methanol adsorption processes. The simulation is conducted for methanol being continuously exposed to the surface at a pressure of 1 bar, although the pressure simply serves as a parameter for the adsorption transition rate constant and no gas phase is explicitly present. Since the surface is initially free of any adsorbates and methanol is the only input gas, the first process to occur in this simulation is methanol adsorption. The transition rate constants for adsorption processes can be calculated by:

$$w_{Ads} = \frac{P}{\sqrt{2\pi k_B T M}} e^{\frac{-E_A}{RT}} \tag{9}$$

where, P is the input pressure of the gas, k_B is the Boltzmann constant, T is the temperature, M is the molecular mass, E_A is the activation energy for adsorption and R is the gas constant. The surface sites geometry for methanol adsorption on the CeO_2 (111) surface can be described using Figure

1.[37] A methanol molecule will adsorb onto the surface and bridge between the site marked A and any one of the neighboring B sites (B1, B2 or B3) – these represent Ce and O sites, respectively. The probability of any of these three orientations occurring is equivalent, assuming that all three B sites are unoccupied. Here, the A sites are the lattice positions of surface layer cerium cations and the B sites are the lattice positions of surface layer lattice oxygen anions. Thus, when adsorption of methanol occurs, both a Ce site and an O site are occupied. Other intermediates will also adsorb on A sites, B sites, or bridge both sites.

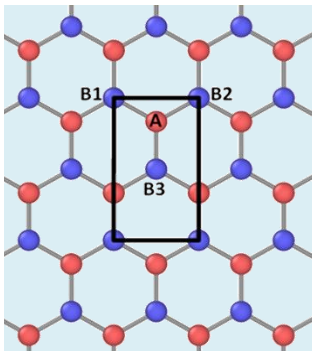


Figure 1. The CeO_2 (111) surface geometry used for KMC simulations. Red and blue correspond to Ce and O sites, respectively.

The reaction pathway has been adapted from one in the literature[35, 36], and tailored to result in steady-state output fluxes of methanol, H_2 and formaldehyde. Reaction processes within the simplified reaction network are provided in Table 1 and include adsorption, desorption, and ionic dissociation and association. The transition rate constants for desorption, associative and dissociative processes are given by:

$$w = Ae^{\frac{-E_A}{RT}} \tag{10}$$

Where, A is the pre-exponential factor, E_A is the activation energy for the reaction process, R is the gas constant and T is the temperature. The simulation size consists of 400 unit cells, corresponding to a surface with dimensions of 38.25 x 66.26 Angstroms. The number of KMC steps per snapshot, and number of snapshots per simulation, are specified for each simulation within the Results section.

Table 1 Reaction processes within the sample reaction network. Reactions containing a "↔" indicate that there is both a forward and reverse reaction process listed and transition probabilties are listed in the same order. Reactions containing a "→" indicate that only one direction exists.

Reaction Number	Reaction Name	Chemical Equation	Forward Transition Rate Constant (s^{-1})	Reverse Transition Rate Constant (s^{-1})
1	H_2 Production	$2(H^{*}$ -- Ce^{4+} -- $O^{2-})$ → $H_2(g)$ + 2 Ce^{4+} -- O^{2-}	1.48×10^{9}	-
2	H* production/reduction	Ce^{3+} -- O^{2-} -- H^{+} ↔	9.79×10^{6}	3.63×10^{10}

		H^* -- Ce^{4+} -- O^{2-}		
3	H^+ Hopping between Ce^{3+} and Ce^{4+}	Ce^{4+} -- O^{2-} -- H^+ + Ce^{3+} -- O^{2-} $\leftrightarrow$ Ce^{4+} -- O^{2-} + Ce^{3+} -- O^{2-} -- H^+	$2.01x10^{12}$	$2.01x10^{12}$
4	H_2O Molecular Desorption	H_2O -- Ce^{4+} -- O^{2-} $\rightarrow$ H_2O + Ce^{4+} -- O^{2-}	$3.9x10^{10}$	-
5	H_2O ionic association/dissociation	H_2O -- Ce^{4+} -- O^{2-} + Ce^{4+} -- O^{2-} $\leftrightarrow$ OH^- -- Ce^{4+} -- O^{2-} + Ce^{4+} -- O^{2-} -- H^+	$4.53x10^{12}$	$1.28x10^{8}$
6	OH^- hopping in/out of vacancy	OH^- -- Ce^{4+} -- O^{2-} + Ce^{3+} -- $[]^0$ $\leftrightarrow$ Ce^{4+} -- O^{2-} + Ce^{3+} -- O^{2-} -- H^+	$4.53x10^{12}$	$1.95x10^{6}$
7	H_2O hopping in/out of vacancy	H_2O -- Ce^{4+} -- O^{2-} + Ce^{3+} -- $[]^0$ $\leftrightarrow$ H_2O -- Ce^{3+} -- $[]^0$ + Ce^{4+} -- O^{2-}	$4.53x10^{12}$	$6.25x10^{11}$
8	Methanol Adsorption/Desorption	$CH_3OH(g)$ + 2 Ce^{4+} -- O^{2-} $\leftrightarrow$ Ce^{4+} -- O^{2-} -- CH_3OH -- Ce^{4+} -- O^{2-}	$5.08x10^{6}$	$1.46x10^{9}$
9	CH_3OH ionic dissociation/association	Ce^{4+} -- O^{2-} -- CH_3OH -- Ce^{4+} -- O^{2-} $\leftrightarrow$ CH_3O^- -- Ce^{4+} -- O^{2-} + Ce^{4+} -- O^{2-} -- H^+	$1.74x10^{6}$	$6.37x10^{8}$
10	H abstraction from ionic CH_3O	CH_3O^- -- Ce^{4+} -- O^{2-} + 2 Ce^{4+} -- O^{2-} $\rightarrow$ CH_2O -- Ce^{4+} -- O^{2-} + Ce^{3+} -- O^{2-} + H^* -- Ce^{4+} -- O^{2-}	$3.29x10^{6}$	-
11	H abstraction from ionic CH_3O on a vacancy	(CH_3O^-) – Ce^{3+} -- $[\ ]^0$ + 2 Ce^{4+} -- O^{2-} $\rightarrow$ (CH_2O) -- Ce^{3+} -- $[\ ]^0$ + H^* -- Ce^{4+} -- O^{2-} + Ce^{3+} -- O^{2-}	$4.53x10^{9}$	-
12	Ionic CH_3O migration	Ce^{4+} -- O^{2-} + (CH_3O^-) -- Ce^{3+} -- $[\ ]^0$ $\leftrightarrow$ CH_3O^- -- Ce^{4+} -- O^{2-} + Ce^{3+} -- $[\ ]^0$	$6.89x10^{7}$	$9.28x10^{11}$
13	CH_2O Desorption	Ce^{4+} -- O^{2-} -- CH_2O $\rightarrow$ CH_2O + Ce^{4+} -- O^{2-}	$6.78x10^{9}$	-
14	CH_2O Desorption from Vacancy	CH_2O – Ce^{3+} -- $[\ ]^0$ $\rightarrow$ CH_2O + Ce^{3+} -- $[\ ]^0$	$5.88x10^{7}$	-

3. **Throttling Algorithm**

3.1 Conceptual Description of the Algorithm

The SQERTSS algorithm is designed to decrease the occurrence of fast frivolous processes and increase the occurrence of slower processes. The intention is to enable computationally tractable simulations that a) reach the reaction network's most likely steady-state from a given starting configuration, b) can simulate experimentally relevant simulation time-scales, and c) do not significantly alter the underlying kinetics of the system. The algorithm involves first classifying the processes as fast frivolous processes (FFPs), slow processes (SPs), and identifying the fastest rate limiting process (FRP), followed by "compressing" the transition rate constants of these processes to be nearer to each other with as little disruption of the underlying kinetics and system dynamics as possible.

Before delving into the details of the throttling algorithm, it is useful to define some important terms. A snapshot is a grouping of sequential KMC steps (typically thousands of individual events or more), over which quantities such as event frequencies are averaged to evaluate the state of the system. An event frequency (EF, with units of **events** $*$ **s^{-1}** $*$ **#unit cells^{-1}**) is the rate for a given process during a given snapshot of the KMC simulation. Typically, the number of steps in a snapshot will be significantly larger than the number of sites in a simulation, and this algorithm would not be needed for systems that can be simulated using snapshots smaller than the number of sites in a simulation. For each snapshot, we rely on three quantities related to the event frequencies. The observed event frequency (oEF) is the most recent EF observed for a particular process during simulation, and is based on the number of events which occurred divided by the simulation time transpired during that snapshot. The unthrottled event frequency (uEF) of a process is a back-calculated value for what the EF would have been during the previous snapshot in the absence of throttling. The predicted throttled event frequency (ptEF) is a prediction for what the EF will be during the next (to be executed) snapshot with a given level of throttling applied. A throttling factor (TF) is the coefficient by which the transition rate constant of a given process is being multiplied to achieve the throttling level desired. How much throttling is applied is based on the event frequency range, EF^{Range}, which is defined as *the EF associated with the fastest speed-rank* divided by *the EF associated with the slowest speed-rank*. Speed-ranks are explained below. During a simulation, the algorithm 'compresses' the EF^{Range} by throttling processes. We are now in a position to explain the details of some of the more complex subtasks in the algorithm. The subtasks in the algorithm can be broadly divided into two groups: (1) the speed-ranking and classification of processes and (2) the calculation of the throttling factors. Below, we describe each of these subtasks in turn.

3.2 Process Ranking and Classification

A central concept of the SQERTSS throttling algorithm is classifying the processes into different types, based on paired speed-ranking of the processes. In paired speed-ranking, the forward and reverse process pairs are assigned speed-ranks (one rank for each pair) based on the EF associated with the faster of the two processes in each pair. Here, the uEFs are used, as their rankings reflect the natural ranking of the processes (i.e., the rankings in the absence of throttling). Each uEF is back-calculated based on the throttling factors of the just-executed snapshot, using uEF = oEF/$TF_{m\text{-}1}$, where m-1 is the number of the just-executed snapshot. The throttling is also applied according to these paired ranks such that the ratios between the forward and reverse processes are maintained for each reaction (i.e., the transition rate constants for the pair forward and reverse processes are multiplied by the same throttling factor). This maintains the thermodynamics for each process pair during throttling. During speed-ranking, first process rank is associated with the fastest process pair, the second process rank is associated with second fastest process pair, and so on. The concept of paired speed-ranks is illustrated in Figure 2, with the corresponding event frequencies ranked in Table 2.

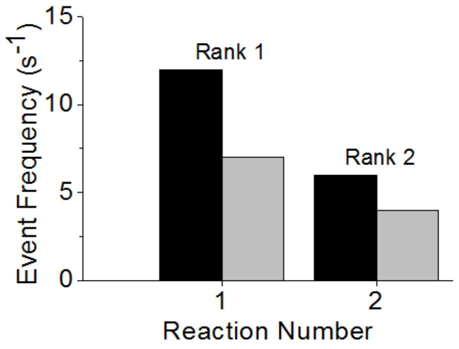


Figure 2. Process ranks example for paired throttling. Forward and reverse processes are shown in black and grey respectively.

Table 2 Process ranks example with forward and reverse process event frequencies separated into ranks using paired ranking.

Process Number	***Event Frequency (s^{-1})***	***Rank***
1F	12.1	1
1R	6.8	
2F	5.9	2
2R	3.8	

Subsequent to sorting and speed-ranking the process pairs, they are classified into several types which we will now define. A fast frivolous process (FFP) is any reaction process belonging to a pair where the forward and reverse reactions have equal event frequencies and are each faster than the fastest rate-limiting process (e.g. H^+ hopping back and forth rapidly, or rapid adsorption/desorption). Due to statistical noise, the EFs for the forward and reverse processes of individual FFPs will not generally be exactly equal over a finite period of time, such as a KMC snapshot. In this work, pairs of forward and reverse reaction process are identified as being sufficiently equal to be quasi-equilibrated (QE) when the ratio of the forward and reverse processes is within some threshold from being equal (here, we use a threshold of 0.1). One of the main challenges of KMC simulations with large separations between the fast and slow timescales is that an inordinate amount of time is spent simulating the FFPs with little opportunity for the execution of the slow processes that control the evolution of the system. The strategy used here to decrease the occurrence of the FFPs is to throttle down (reduce) their rate constants while maintaining the ratio of the forward and reverse rates, until the FFPs have event frequencies that do not inhibit observation of rate limiting processes during simulation (thereby also accelerating the elapsed time during each snapshot). The FFPs are not throttled to become so slow as to become comparable to rate limiting processes; the FFPs are only throttled to the extent necessary for the simulation to show sufficient activity of the rate limiting processes.

The benchmark for throttling is chosen as the fastest rate limiting process (FRP). The FRP is the fastest non-QE process: it is the fastest process among the processes for which the forward and reverse oEFs differ by greater than 10%. By using the FRP as a benchmark for throttling, the distortion of the time-steps within VSSM is limited. If the slowest rate-limiting process were used as the benchmark (such that process pairs faster than the slowest rate-limiting process were throttled down to have slower rates), there would be large distortions to the KMC time, where the time-steps would become significantly larger. Process pairs are classified as slow processes (SPs) if their speed-rank is slower than that of the FRP and they have event frequencies that are non-negligible. SPs may or may-not be QE. Negligibly slow processes (NSPs) belong to process pairs where both the forward and reverse event frequencies are so low that they are considered negligible over the timescale of interest (as defined in the last bullet of section 3.5). Just as the FFPs are intentionally throttled down, the SPs can be intentionally throttled up towards the FRP in order to increase the sampling of the slowest relevant processes, and thereby reduce the number of KMC steps needed to achieve a steady-state simulation. However, since throttling up of SPs distorts the time-steps and has a finite risk of distorting the underlying kinetics, the FFPs are always throttled down preferentially and the SPs are *only* throttled up if it is computationally necessary to achieve steady-state with the computational resources available.

3.3 Calculation of Throttling Factors and $EF^{Range\text{-}Pred}$

In essence, the objective of the SQERTSS algorithm is to compress the EF^{Range}, which is defined as the value obtained by taking *the EF associated with the fastest speed-rank* divided by *the EF associated with the slowest speed-rank*. This has the effect of ensuring that events of the slowest speed-rank happen frequently enough to observe during each snapshot. To accomplish this, the algorithm attempts to compress the event frequencies to fall within $EF^{Range\text{-}Req}$, which is a user-supplied value for the event frequency range that is required for the simulation to be computationally tractable. This means that $EF^{Range\text{-}Req}$ and the steps per snapshot should be chosen such that $EF^{Range\text{-}Req}$ is smaller than the steps per snapshot, which will allow processes of the slowest rank to occur a countable number of times per snapshot. After each snapshot, the $EF^{Range\text{-}Pred}$ (which is the predicted event frequency range for a given level of throttling) is calculated for up to several levels of throttling. The $EF^{Range\text{-}Pred}$ is first checked for no throttling, and then checked for progressively more aggressive levels of throttling until $EF^{Range\text{-}Req}$ is achieved or until no further throttling is possible without distorting the speed-rankings. $EF^{Range\text{-}Pred}$ is the ratio of the ptEFs associated with the fastest and slowest speed-ranks. The ptEFs thus play a role in determining what level of throttling should be applied for the next snapshot, with each ptEF calculated by ptEF = TF_m*oEF, where m is the number of the current snapshot, and TF_m represents the throttling factor that corresponds to applying a given level of throttling in the next (to-be-executed) snapshot. Thus, for each snapshot, we use the oEFs are from the just-executed snapshot to calculate the TFs for the next snapshot.

At this stage, we can describe what it means to throttle the processes into having a more 'compressed' EF^{Range}. The ranked event frequencies form a type of staircase in the speed-rankings.

The ratio of the event frequencies of the adjacent ranks (corresponding to the height of one of the steps in this staircase of speeds) is called a "staggering factor". For reasons that will be explained below, we do not throttle the FFPs or the SPs closer to the FRP than a staggering factor of N_{sites}, where N_{Sites} is the number of sites in the simulation. For process pairs that are not directly adjacent to the FRP, the staggering is limited to being within a factor of s_a, such that a smaller s_a corresponds to more aggressive throttling. We maintain the natural staggering factors between any adjacent ranks that are already within the specified tolerance. The visualization of the staircase after compression is shown in Figure 3, which shows that each staggering factor presents an upper limit for the EF spacing between speed-ranks. The objective is to make s_a as small as possible without distorting the system dynamics. This is accomplished using discrete throttling scales with progressively more aggressive throttling.

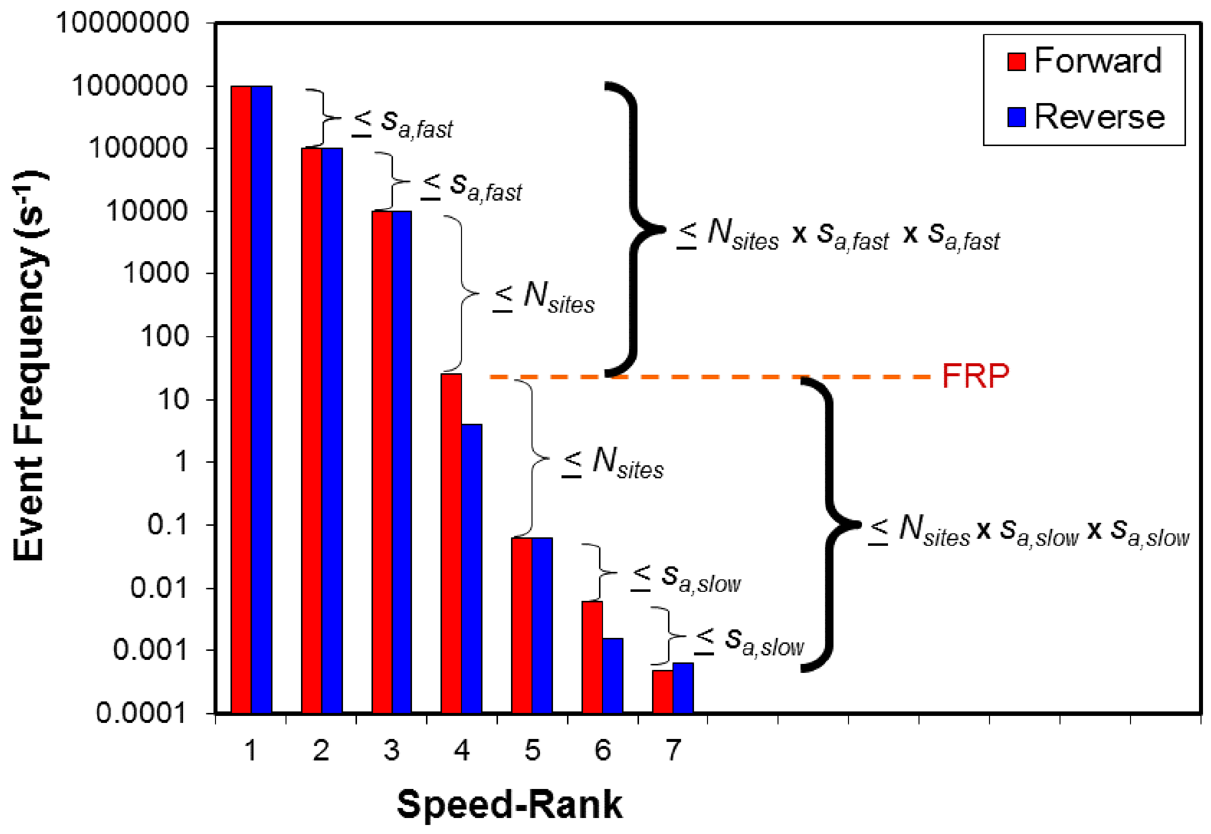


Figure 3. A staggering factor is the ratio between the event frequency associated with a particular speed-rank and that of the speed-rank below it. During throttling, the maximum staggering factors are set as $\leq N_{sites}$ for speed-ranks directly adjacent to the FRP, and are set as $\leq s_a$ for other speed-ranks. The values of s_a can be varied independently for the upper wing of processes (processes with speed-ranks faster than the FRP) and lower wing of processes (processes with speed-ranks slower than the FRP).

There are three throttling scales used in this study, with increasingly more aggressive values for s_a. The approach used here is to check whether $EF^{Range\text{-}Pred} \leq EF^{Range\text{-}Req}$ without throttling, and (if the requirement is not met) to then to check whether $EF^{Range\text{-}Pred} \leq EF^{Range\text{-}Req}$ is met using the first level of throttling, then using the second level of throttling, and so on down the possible levels of throttling until $EF^{Range\text{-}Pred} \leq EF^{Range\text{-}Req}$ is met or until the system cannot be throttled further while maintaining the speed-rankings. As will be explained below, the maximum contribution to the EF^{Range} from either the upper or lower wing of the rankings (speed-ranks faster than the FRP and speed-ranks slower than the FRP, respectively) can be related to the number of non-FRP speed-ranks in that wing. For the three throttling levels here, the contributions to the EF^{Range} from one wing of the processes is then as follows (from least compressed scale to most compressed scale):

1. N_{Sites}^{P}
2. $N_{Sites} * C^{(P-1)}$
3. $N_{Sites} * 1.1^{(P-1)}$

where P is the number of process ranks in that wing and C is a user defined constant (set as 10 in the current study), where C is chosen to have increased compression relative to the N_{Sites}^{P} scaling, while being less extreme than the third scale. The "P-1" factor appears because for either wing of processes the process rank closest to the FRP is throttled to have a staggering of $\leq Nsites$ relative to the FRP, while the other process ranks are throttled to having a staggering of $\leq s_a$.

The first and third throttling scale (numbered 1 and 3 above) were chosen based upon two limits, and the second (middle) throttling scale was chosen to be between the two limits. The first throttling scale (N_{Sites}^{P}) represents the limit of allowing each site to turn over one time on-average for every occurrence of processes of the next rank. For example, if there are Y sites, then with the (N_{Sites}^{P}) scaling a Rank 1 process will occur Y times on average prior to each occurrence of a Rank 2 process (i.e., enabling the Rank 1 process to occur once on each site prior to any Rank 2 process occurring), while the Rank 2 process will occur Y time on average for each occurrence of a Rank 3 process. This throttling scale will introduce no inaccuracies when the relationship between the configurations and the event frequencies are strictly first order. The third throttling scale ($N_{Sites} * 1.1^{P-1}$) represents a limit in which the event frequencies of the fast speed-ranks (or slow speed-ranks) are throttled towards having a single event frequency that is a factor of N_{Sites} away from that of the FRP (i.e. the scaling becomes, $N_{Sites} * L^{P-1}$, with the limit of $L \rightarrow 1^{+}$). L is kept above 1 in order to retain separate rankings between the various process pairs. In this study, 1.1 is used for L, for reasons explained below. For example, if there are 2000 sites, and the fastest rate limiting process has an event frequency of 2000, then if there are three ranks of FFPs above the FRP, the third throttling scale would constrain the three ranks of FFPs to having ptEFs within the following limits for each rank: (1) Y * N_{Sites} , (2) Y * N_{Sites} * 1.1, and (3) Y * N_{Sites} * 1.1 * 1.1. In other words, when a process requires throttling, the TF for that process is the product of the staggering factor between it and the next faster (for slow processes) or slower (for fast processes) process and the TF for that adjacent process.

Effectively, in the third throttling scale, the FFPs will be throttled to be very near each other while having their rankings retained (if they are truly FFPs, this should not affect the system dynamics). We choose a factor of 1.1 for our limit as this choice is consistent with our previous choice of setting the quasi-equilibrium threshold for forward and reverse processes to being within a factor of 10%. The third scale of throttling has a greater risk of affecting the system dynamics compared to Scale 1. The second throttling scale ($N_{Sites} * C^{P-1}$) is chosen to be intermediate between the limits of the first and third throttling scales, such that $1 < C < N_{Sites}$. During dynamic throttling, $EF^{Range\text{-}Req}$ specifies the level of compression required for the simulation to achieve the desired time-scales (given the computational resources and wall-clock time available), and based on this value the minimum level of compression to achieve $EF^{Range\text{-}Req}$ is used. We note that the three compression scales used here can each be reduced to the form $N_{Sites} * s_a^{(P-1)}$, where the factor s_a takes values in this work of N_{Sites}, 10 or 1.1. Recognizing that the various throttling scales can be encompassed by a single function of s_a, we note that the factor of s_a could, in principle, be implemented as a continuous variable for even greater optimization of accuracy and efficiency – but some criteria would need to be developed on how to modulate the compression in that context, which is beyond the scope of this work.

The progression of throttling scales is listed in Table 3: the first index represents the scale for the FFP wing and the second index represents the scale for the SP wing. The scales are listed starting from no compression (first row) and progressing to the most compressed scale (last row). As can be seen, the FFPs are always given preference in throttling prior to any throttling of SPs. An example of compression is shown in Figure 4, which depicts the effects of compression on an example set of event frequencies. Figure 4 shows four of the scales: these four are a subset of the scales in Table 3, and represent the subset of cases where both the FFP wing and the SP wing are set to the same compression. It is clearly shown that the compression increases from scale 1 to scale 3, and that the compression can be orders of magnitude in extent. Due to the paired ranking and throttling, the ratios of the forward and reverse process are maintained. As mentioned previously, during dynamic throttling, we apply compression preferentially to the FFPs prior to SPs, since the FFPs can be compressed without affecting the system dynamics and without time distortion when our assumptions are met.

During dynamic throttling, the throttling scale with the highest accuracy (least compression) that meets the required compression criteria is used. Such compression is necessary to achieve the goal of increased computational efficiency and simulations with time-scales of experimental relevance (on the order of minutes to hours, in this case). The concept of throttling is based on (1) reducing the computational effort expended on simulating FFPs and (2) ideally ensuring that the slowest process rank occurs at least once per computationally feasible snapshot. For the KMC simulations in this study, it was determined that if the slowest process rank occurred on the order of once per one million configuration steps, then detecting significant changes in the system would be computationally feasible (i.e., it was determined that 1 million KMC steps per snapshot was computationally feasible). Thus, $EF^{Range\text{-}Req}$ was set as 10^6.

Table 3. The progression of throttling scales for setting the throttling factors of the fast processes and the slow processes.

Throttling Scale (Fast, Slow)	***Fast Processes Scaling***	***Slow Processes Scaling***
0 , 0	*Unthrottled*	*Unthrottled*
1 , 0	N_{Sites}^{P}	*Unthrottled*
2 , 0	$N_{Sites} * \; C^{P-1}$	*Unthrottled*
3 , 0	$N_{Sites} * 1.1^{P-1}$	*Unthrottled*
1 , 1	N_{Sites}^{P}	N_{Sites}^{P}
2 , 1	$N_{Sites} * \; C^{P-1}$	N_{Sites}^{P}
3 , 1	$N_{Sites} * 1.1^{P-1}$	N_{Sites}^{P}
1 , 2	N_{Sites}^{P}	$N_{Sites} * \; C^{P-1}$
2 , 2	$N_{Sites} * \; C^{P-1}$	$N_{Sites} * \; C^{P-1}$
3 , 2	$N_{Sites} * 1.1^{P-1}$	$N_{Sites} * \; C^{P-1}$
1 , 3	N_{Sites}^{P}	$N_{Sites} * 1.1^{P-1}$
2 , 3	$N_{Sites} * \; C^{P-1}$	$N_{Sites} * 1.1^{P-1}$
3 , 3	$N_{Sites} * 1.1^{P-1}$	$N_{Sites} * 1.1^{P-1}$

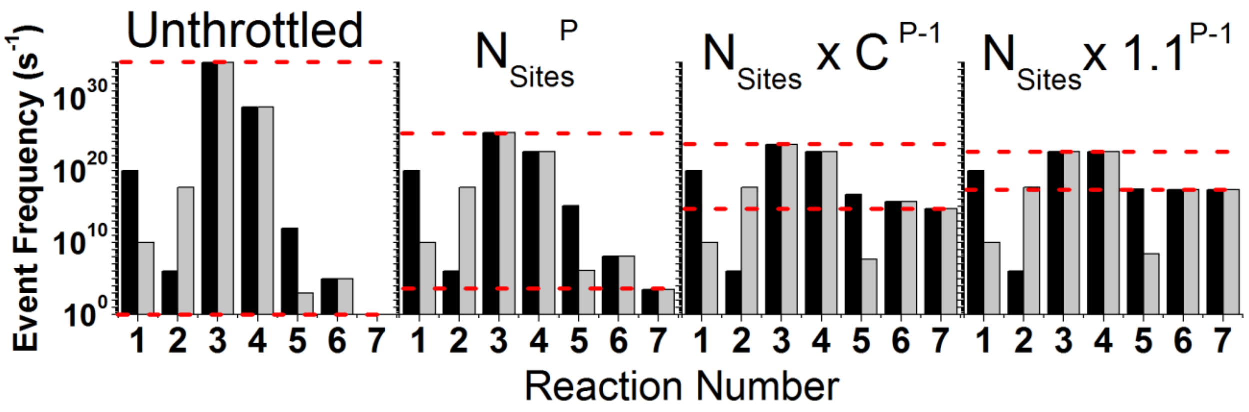


Figure 4. Example of EF^{Range} compression with different throttling scales. Black and grey bars represent forward and reverse processes respectively. The cases shown are the subset of cases for which the same throttling scale is applied to both FFPs and SPs. This subset corresponds to the rows in Table 3 denoted by throttling scales of (0,0) , (1,1), (2,2), and (3,3). The scaling shown is indicated above each figure panel. The red horizontal lines are used to illustrate the differences in EF^{Range} compression between the different scalings.

3.4 Throttling Stepdown When there is No FRP

When the KMC simulation gets in to a metastable state where only QE processes exist, these processes must be FFPs. There are two possible ways that the simulation can exit such a state: A) by entering into a *rare configuration* from which a non-QE process can occur, or B) by having a rare (*low transition rate constant)* non-QE process occur. We do not have a way of predicting which type of exit path the simulation will. In such a state, the configuration space that the simulation is traversing is independent of the values of the transition rate constants, provided that the staggering is not changed (because we have only QE processes). Thus, we take the approach of decreasing each of the FFP transition probabilities by the same factor. In this approach, all of the FFPs are throttled down together, slowly, which preserves the system dynamics while increasing the relative likelihood of a rare process occurring in order to exit the metastable state. We term each iteration a "stepdown", and apply a small stepdown after each snapshot when only FFPs are present. In the event that the metastable state consisting of only FFPs persists, it is advantageous to stepdown repeatedly. The stepdown factor is chosen to be 10 in this work, which is safely within the bounds required for the marginal change assumption (described below) since our snapshots have greater than 10^2 steps per snapshot. If the FFP transition probabilities are decreased too much, the time-steps can become distorted, which can in turn distort the observed event frequencies for product formation. As such, we implement a floor value (FFP^{Floor}) for the throttling down of FFP QE processes, and the EFs of FFP processes are not throttled below the floor. The FFP^{Floor} should be orders of magnitude higher than the EF of the steady-state FRP, and is chosen to be 10^6 s^{-1} in this work, as explained in the discussion section.

3.5 Algorithm Steps and Flowchart

Putting together the above-mentioned concepts, the complete throttling algorithm takes the below structure, which is also expressed in flowchart form in Figure 5 and in mathematical form in the Appendix. The flowchart has been constructed such that the typical flow is in a counter clockwise loop from the top left after each snapshot. In the algorithmic text below, we intentionally do not include the level of detail of "for/while" loops across the different process ranks: this is because our intention is to present the high-level algorithm, which can be programmed in more than one possible way. However, an expanded form of the algorithm which includes some of the choices we utilized for implementation is included in the supporting information, and a version of the algorithm in mathematical form is included in the Appendix. The variable "*m"* is a counter for the current snapshot, and the variable M is a constant that corresponds to how often the system is periodically relaxed by unthrottling the SPs. The TFs are first *set* (calculated) and are then *applied* by adjusting the transition rate constants. As the TFs are *set* by an iterative process when an FRP is present (depicted in the lower right-hand corner of the flowchart), the TFs cannot be *applied* until after that iterative process.

We employ two closely related variations of throttling in the algorithm: one set of steps to accelerate reaching of steady-state, and another (very similar) set of steps for simulating steady state once it is reached. The primary difference between the two sets of steps is that during steady-state the SPs are set to be unthrottled during *every snapshot,* while the SPs are only set to be unthrottled *periodically* during acceleration towards steady-state. We note that it is possible to start directly with the steady state version of the algorithm, which does not distort the time steps, but also does not accelerate the system beyond induction periods.

SQERTSS Algorithmic Steps to Accelerate Reaching Steady State

0. set m=0. Run a KMC snapshot with no throttling applied. Increase m by 1.
1. Gather all oEFs and rank the processes:
    a. The processes are ranked with paired ranking. If a process is irreversible, then its reverse process is assumed to have an EF of zero.
    b. Classify each process pair as FFP, FRP, SP, or NSP.
        i. Any process in a pair that is negligibly slow is not throttled, and is not considered in the below algorithmic steps.
2. Calculate TFs:
    a. If any FRP is present, set TFs throttling by iterating over the compression scales from Table 3 until *either* $EF^{Range\text{-}Pred}$,< $EF^{Range\text{-}Req}$ is achieved *or* the final compression scale is reached.
        i. During this process, each process pair that is not the FRP is set to have TFs to have the correct staggering ratio (based on Table 3) relative to the FRP or relative to the process pair which is one rank closer to the FRP.
    b. If all process pairs are FFPs, then set TFs to step down all FFP transition rate constants by a factor of 10 -- unless the ptEF of any FFP will reach the FFP^{Floor}, in which case the stepping down of FFPs will be constrained to not go below the FFP^{Floor}.
    c. If m/M is an integer, unthrottle the SPs by setting their TFs to 1.
3. Apply the calculated TFs to the transition rate constants. Run a snapshot, and increase m by 1.
4. Repeat steps 1-3 until steady-state is reached.
5. The above steps accelerate the reaching of steady-state. After steady-state is reached, the user should run additional snapshots using the steps listed below, to ensure that the steady-state behavior is captured correctly.

SQERTSS Algorithmic Steps to Simulate Steady State

0. Set m=0. Run a KMC snapshot using the most recent throttling level applied. Increase m by 1.
1. Gather all oEFs and rank the processes:

    a. The processes are ranked with paired ranking. If a process is irreversible, then its reverse process is assumed to have an EF of zero.
    b. Classify each process pair as FFP, FRP, SP, or NSP.
        i. Any process in a pair that is negligibly slow is not throttled, and is not considered in the below algorithmic steps.
2. Calculate TFs:
    a. If any FRP is present, set TFs throttling by iterating over the compression scales from Table 3 until *either* $EF^{Range\text{-}Pred}$,< $EF^{Range\text{-}Req}$ is achieved *or* the final compression scale is reached.
        i. During this process, each process pair that is not the FRP is set to have TFs to have the correct staggering ratio (based on Table 3) relative to the FRP or relative to the process pair which is one rank closer to the FRP.
    b. If all process pairs are FFPs, then set TFs to step down all FFP transition rate constants by a factor of 10 -- unless the ptEF of any FFP will reach the FFP^{Floor}, in which case the stepping down of FFPs will be constrained to not go below the FFP^{Floor}.
    c. Always unthrottle the SPs by setting their TFs to 1.
3. Apply the calculated TFs to the transition rate constants. Run a snapshot, and increase *m* by 1.
4. Repeat steps 1-3 until $m \geq M$, and as long as desired to collect further steady state statistics.

During the snapshots where the SPs are periodically unthrottled (and after steady-state is reached), compression is applied first, then the slow processes are unthrottled (thus, the same compression of FFPs is maintained during these snapshots). Within this algorithm, Step 2 shows that all possible throttling combinations are explored until the compression is sufficient to allow all slow process ranks to occur at least once within a computationally feasible number of steps (excluding NSPs). Step 3c is performed once every *M* times as a means to relax the system (but it is reasonable to assume that primarily FFP events will occur during this Step if FFPs are present). Once steady-state is achieved, the simulation is run with the SPs unthrottled (in the same fashion as during the periodic SP unthrottling, such that the compression of FFPs is maintained during these snapshots).

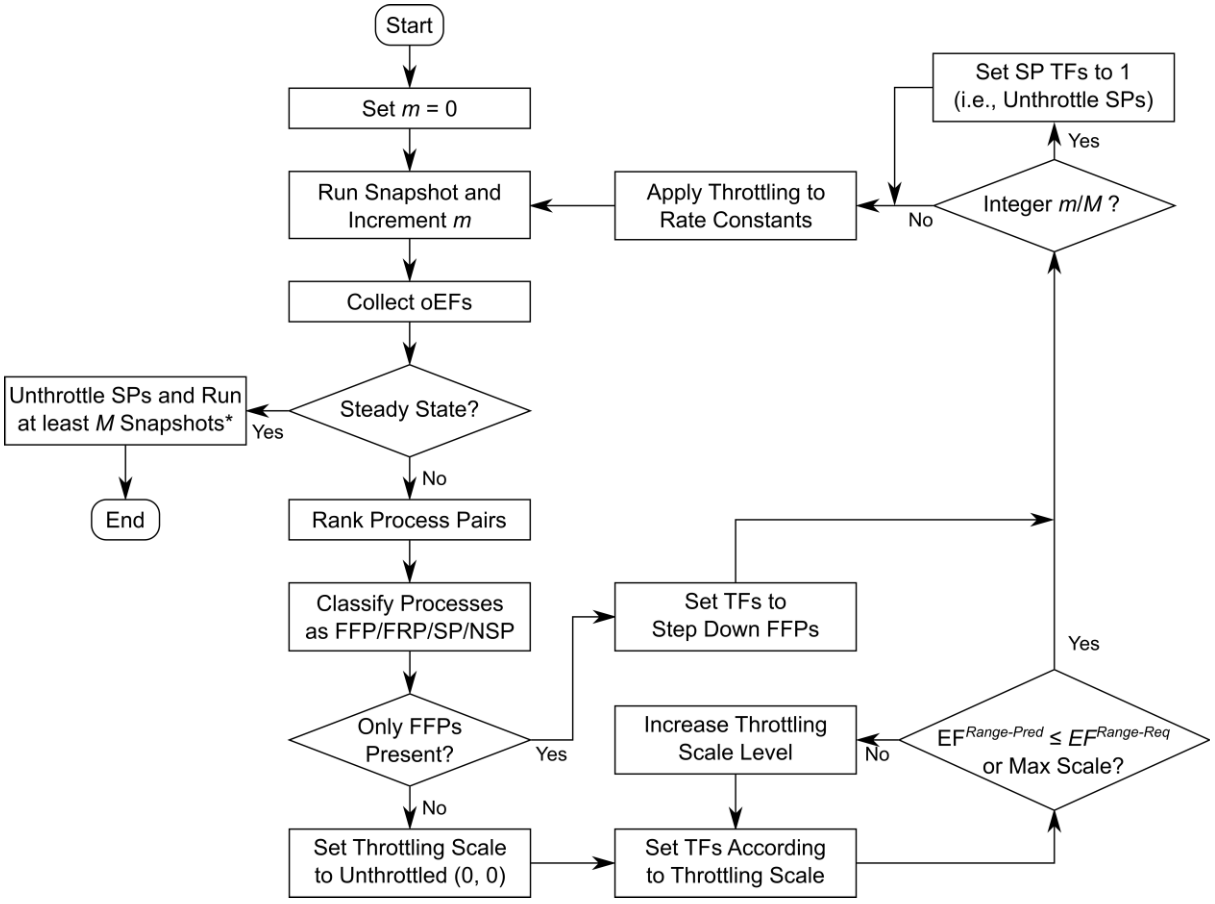

Start
Set *m* = 0
Run Snapshot and Increment *m*
Collect oEFs
Steady State?
Yes
Unthrottle SPs and Run at least *M* Snapshots*
End
No
Rank Process Pairs
Classify Processes as FFP/FRP/SP/NSP
Only FFPs Present?
Yes
Set TFs to Step Down FFPs
No
Set Throttling Scale to Unthrottled (0, 0)
Set TFs According to Throttling Scale
EF^Range-Pred ≤ EF^Range-Req or Max Scale?
No
Increase Throttling Scale Level
Yes
Integer *m*/*M* ?
No
Apply Throttling to Rate Constants
Yes
Set SP TFs to 1 (i.e., Unthrottle SPs)

Figure 5. Flowchart of throttling algorithm. Abbreviations: oEF = Observed Event Frequency, FFP = Fast Frivolous Process, FRP = Fastest Rate-Limiting Process, SP = Slow Process, NSP = Negligibly Slow Process, TF = Throttling Factor. Note that the first snapshot run has no throttling applied to it. The compression restriction of $EF^{Range\text{-}Req}$ is met if $EF^{Range\text{-}Pred} < EF^{Range\text{-}Req}$. The relaxation period M is set by the user and controls how often the SPs are temporarily unthrottled in order to temporarily relax the system dynamics.. *After steady-state is reached, additional snapshots should be run with the SPs unthrottled to ensure correct capturing of the steady-state system dynamics.

3.6 Assumptions and Approximations involved in Algorithm

The following approximations and assumptions have been made which must be fulfilled or distortions of the system behavior will occur. All of these approximations and assumptions will usually be valid, enabling this algorithm to be quite general:

- The event frequencies are assumed to be linear with respect to the appearance of the local configurations required for a process to occur: this will typically be true for systems for which the VSSM KMC algorithm is appropriate. This enables us to predict the EF^{Range} for the next snapshot ($EF^{Range\text{-}Pred}$) during throttling by assuming that (for example) reducing the transition rate constant for a reaction by half will reduce the event frequency for that reaction by half. This assumption is present in the calculation of the uEFs and the ptEFs from the oEFs.

- Based on the linearity assumption stated above, the least aggressive throttling scale has staggering factors of N_{Sites}. When this assumption is valid, N_{Sites} scaling enables processes of a given rank to occur (on average) once on each of the sites before any processes of the next fastest rank can occur (i.e., process rank 1 has the opportunity to completely change the nature of the surface between each execution of any event from process rank 2). To the extent that this assumption is accurate, the first throttling scale will introduce no inaccuracies in the system dynamics.
- Throttling is performed with respect to the FRP in order to maintain the least amount of distortion to the variable time-steps; this is because the FRP determines the rate of production of the major product in a steady-state system. If we throttled with respect to any of the slower RPs, we would more significantly distort the time-steps and the rate of production of the major product.
- FFPs constitute a quasi-pre-equilibrium for reactants or intermediates (or are outside of the main reaction pathway), and thus no significant changes in the reaction system are driven by the processes *provided* that the FFP retains QE during throttling: this requirement is preserved in our algorithm, which applies the same throttling factor to both the forward and reverse processes in a given speed-rank. In contrast, the slow reactions are not generally QE, and systemic changes are thus still being driven by these processes. It is more important to preserve the accuracy of the SPs, and throttling of SPs is thus only applied if compression of the FFPs is insufficient to achieve computationally tractable simulations.
- The SPs are unthrottled after every M steps to allow the system to relax locally in time: this better ensures that the system's dynamics are progressing along the same trajectory as if the SPs were not throttled. M is user supplied, and chosen to be 10 in the present work.
- To be free from distortions of either the time-steps or system dynamics, the algorithm relies on the assumption that any change in rate of a QE FFPs and other process pairs are only marginal between snapshots -- such that any departure from QE is gradual enough for the dynamic throttling to adapt (i.e., that significant changes in event frequencies are not more abrupt than a snapshot), and also that any changes in the staggering are gradual. This assumption is valid provided that no FFP or other process pair changes rates faster than their throttling staggering factor *within a single snapshot*. An example is provided in the supporting information. Gradual changing of the positions in the rankings does not cause any distortion, since processes will stop being throttled relative to each other when their rates approach each other (thus, they will not have any throttling relative to each other during the period of snapshots where they are switching ranks). It is important to note that even when the marginal change assumption is not met, any distortion would only be for the period of the snapshot during which the marginal change assumption is not met and is unlikely to result in any qualitative changes of the system dynamics. Further, even when the marginal change assumption is exited in a particular snapshot, the throttling factors will be recalculated for the next snapshot, and thus the size of the time-steps and system

dynamics will be driven back towards the natural system dynamics (particularly if the marginal change assumption is becomes met once more). If there is only one possible steady-state, then that same steady-state will still be reached. If there is more than one possible steady-state, it is anticipated that the natural steady-state will still always be reached (see supporting information), except for systems very near a bifurcation point. It is possible to define the marginal change requirement through an inequality.

If we have departure from QE for the oEFs of forward and reverse process of reaction *i* between two snapshots at time *t* and *t+1* we can express the change in rate as:

$$\Delta R_i = R_i^t \text{-} R_i^{t+1}$$

And similarly, the staggering S_i^t can be defined as the *ratio* between the oEFs of two adjacent processes ranks (*i* and *i+1)* at a given time-step *t*, when ordered from least to greatest. So the change in the staggering between two snapshots can be expressed as:

$$\Delta S_i = S_i^t - S_i^{t+1}$$

From the above definitions, we see that the marginal change assumption is fulfilled when $\Delta R_i \ll \Delta S_i$ between any two snapshots (since in this case there is no change in rate greater than the staggering).

Given that the marginal change assumption is important, we make a few notes here. 1) As noted above, any distortion of the system dynamics during the transient period will only be for the duration during which the marginal change assumption is not met; 2) For a snapshot to not meet the marginal change assumption is unlikely and rare with appropriately chosen snapshot sizes, since an appropriately chosen snapshot size will still have FFPs as the vast majority of the events; 3) The probability of such a snapshot then pushing the system configuration into a trajectory that will reach a different steady-state (relative to what would be reached without throttling) is even lower, particularly since the algorithm re-ranks the processes and re-calculates the throttling factors for every snapshot; 4) the probability for a snapshot to not meet the marginal change requirement (even temporarily) is yet more rare during steady-state relative to during the transient period.

- If a given process rank is projected to occur less than N_{Sites} times in the maximum relevant simulation time (MRST) -- which is 10 hours in the test case presented here -- then no throttling is applied to that process rank at the current snapshot. These processes are considered “negligibly slow processes” (NSPs) because these processes are so slow that they should not drive significant change of the system on the relevant time scale nor be significant participants in the primary reaction pathway. In addition, if extremely slow poisoning processes happen to be autocatalytic, the risk of poisoning the surface would increase by making these reactions occur more frequently. The choice of MRST is based upon the reactor timescales that are relevant to the system of study, and the choice for MRST will thus be system specific, leading to a threshold ($EF^{Negligible}$) below which process pairs are considered negligible. A process that starts as an NSP can of course become relevant later during a simulation since the processes are re-ranked and re-classified after every snapshot.

4. Results

The focus of the current section is the application of the SQERTSS algorithm to the sample reaction network previously described in Section 2.3 and outlined in Table 1. The snapshot sizes used were 10^5 KMC steps per snapshot for the throttled simulations and 10^6 KMC steps per snapshot for the unthrottled simulations (the larger snapshot size was used for the unthrottled simulations to provide better averaging of the EFs shown in the figures). The number of snapshots was typically chosen to exceed 10^8 KMC steps per simulation, which was sufficient to achieve converged system dynamics. The reaction model (consisting of the transition rate constants and reactions) are a subset from one that is published. [35, 36] To test this algorithm, values of the transition rate constants have been altered relative to those in the references provided in order to find conditions where the CH breaking reactions (Reactions 10 and 11 in Table 1) were competitive enough to result in sustained output fluxes of methanol, H_2, and formaldehyde, over sufficiently long time-scales for the algorithm to be tested appropriately. The only incoming gas flux is that of methanol, with the surface initially free of adsorbates, such that the first process to occur will be methanol adsorption. Subsequently, dissociation of the methanol molecule to produce H^+ on the surface can occur via reaction number 9. Various processes with low activation barriers occur when H^+ is accumulated on the surface such as reaction number 3 and (when Ce^{3+} exists on the surface) reaction number 2. These low activation barrier processes are responsible for the small time-steps in an unthrottled simulation and give rise to the "KMC stiffness" that the throttling algorithm has been developed to address.

The current section will describe the effectiveness, and relative differences, for each of the throttling scales and the dynamic throttling scale algorithm. Additionally, the efficiency of dynamically throttled simulations will be compared to that of unthrottled simulations for both transient and steady-state cases based on the reaction system investigated. These comparisons will be mainly focused around formaldehyde desorption events. To show the generality of this algorithm, a fictional complex reaction network -- one designed to have multiple FFPs as well as changing FRPs during the simulation -- has also been tested and is presented in supplementary information. All simulations have been performed on a Linux virtual machine using an Intel Core 2 Duo CPU with a processing speed of 2.33 GHz and 4 GB of RAM.

The dynamic throttling algorithm as well as simulations constrained to each of the four throttling scales shown in Figure 4 have been applied to a subset of the sample reaction network, defined by reactions 1, 2, 8, 9, 10, and 13 from Table 1. The formaldehyde desorption event frequencies observed for the dynamic throttling algorithm as well as the four constrained throttling scale simulations are shown in Figure 6a and b for the same span of time (not the same span of KMC steps). In Figure 6a, the series labelled "Dynamic" refers to the full dynamic throttling algorithm, where the throttling algorithm can access all of the scales and progresses through the throttling scales applied with increasing compression until the most accurate computationally tractable compression is reached (and relaxes compression if needed). The grey, blue, and green markers correspond to simulations where only a single compression scale is applied throughout the simulation (e.g. the grey marker shows N_{Sites}^{P} scaling applied to the fast processes and also slow processes). Each of the scales is in good agreement with each other, as well as with the dynamic throttling algorithm, giving confidence that even as compression of the time scales increases, the system is still progressing along the same trajectory. We additionally ran a simulation with N_{Sites}^{P} scaling applied to the fast processes and no throttling applied to the slow processes, and that simulation was also in line with the simulations presented here.

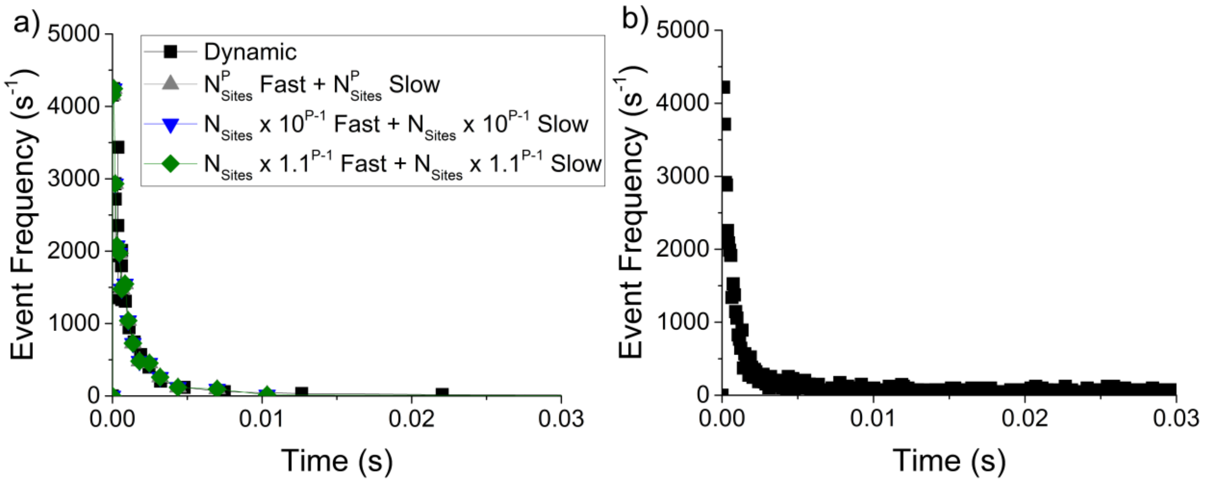


Figure 6. a) Comparison of formaldehyde production outputs from several throttling scales and the full dynamic throttling algorithm when applied to the simplified reaction mechanism. b) Formaldehyde production from unthrottled simulation of the simplified reaction mechanism. There is good agreement between the throttled and unthrottled simulations. The axes have been made equal between these graphs for clarity. The output from the throttled simulations extends to much longer time scales, as shown in Figure 7. In Figure 6a, not all symbols are visible as there is much overlap between the series plotted.

Comparing Figure 6a and Figure 6b shows that throttled simulations follow the same reaction trajectory as an unthrottled simulation. When the unthrottled case is performed using the same snapshot size as the throttled simulations, there are many points where the event frequency for formaldehyde production is zero, due to the fact that the reaction mechanism is largely dominated by FFPs during this time period. To further illustrate that the throttled simulation gives the same output of formaldehyde production, the unthrottled formaldehyde-production and throttled formaldehyde-production have been plotted together in Figure 7 (in units of event frequency). As can be seen, the throttled reaction network was capable of reaching simulation times significantly greater than the unthrottled reaction network. Figure 7 shows an order of magnitude difference in simulation time between the throttled and unthrottled simulations, but in fact the throttled data has been truncated and actually reached times on the order of 10^4 seconds with the same number of KMC steps as the unthrottled simulation.

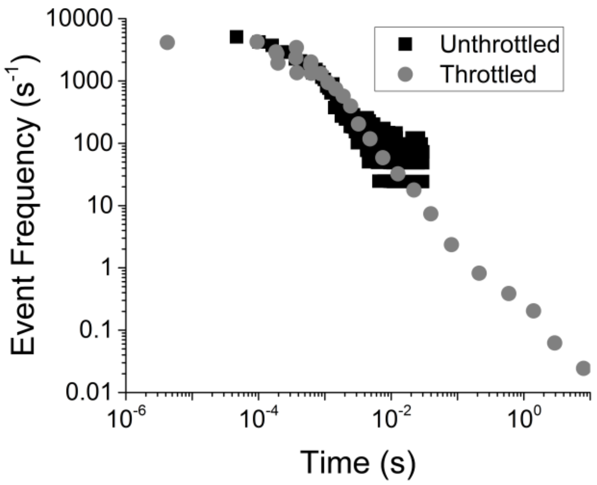
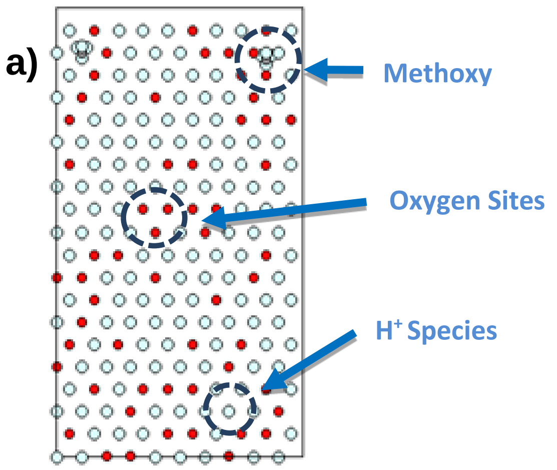


Figure 7. Comparison of the formaldehyde production event frequency for throttled and unthrottled simulations, when using the dynamic form of the throttling algorithm. The throttled simulation overlaps with the unthrottled simulation, but is able to achieve orders of magnitude higher timescales in the same computational time. The data is plotted on a log-log graph, and data points with values of zero are not shown.

The simulations shown in Figure 6 and Figure 7 ultimately reach a steady-state formaldehyde production of 0 molecules/s due to a poisoning of the surface. Thus, to further test the algorithm, a second, larger subset of the full network's reactions was used to produce a simulation with a non-zero steady-state output of formaldehyde. This subset of reactions consists of reactions 1, 2, 3, 8, 9, 10 and 13. That is, in addition to the reactions in the system for the previous section, it also includes H^+ hopping across the surface; this reaction is vital for promoting the formation and desorption of H_2 (reaction 1), which frees up new sites for continued catalytic turnover. The throttling algorithm was applied until steady-state was reached. A steady-state surface configuration and the formaldehyde production rate from the throttled KMC simulation are shown in Figure 8. The steady-state surface configuration was then run without the slow process throttled, and also imported into a completely unthrottled KMC simulation to verify that the dynamics were unchanged by the throttling. In this configuration, the throttled simulation snapshots were about 0.01 s each. In the unthrottled simulation, the snapshots were on the order of 10^{-6} s when the same number of steps per snapshot was used, showing the effectiveness of the throttling algorithm at decreasing the occurrence of FFPs. In the unthrottled simulation, the simulation was dominated by H^+ hopping and electron transfer event FFPs. After 1500 snapshots with 10^5 steps per snapshot, ($>10^8$ KMC steps and 0.01 s) the unthrottled simulation only produced two formaldehyde molecules, while the throttled simulation was easily able to simulate on the order of hundreds of seconds and the production of tens of thousands of formaldehyde molecules.



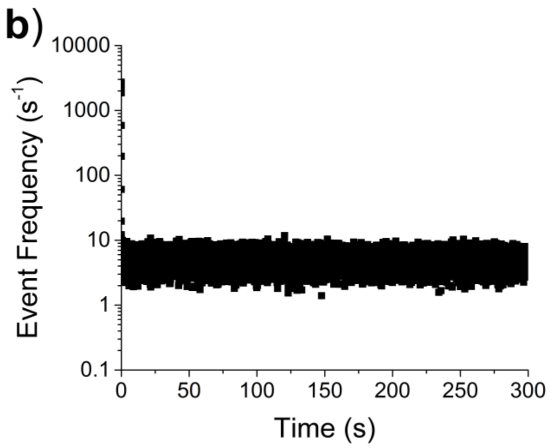


Figure 8. a) A surface configuration exported from a KMC simulation during steady-state formaldehyde production using the dynamic throttling algorithm. b) Formaldehyde production from a KMC simulation which achieves steady state using the dynamic throttling algorithm.

5. Discussion

5.1 Discussion of Results, Limitations, and Efficiency

In general, a chemical reaction system may have one or more steady states that are accessible from a given initial configuration of species. When more than one steady-state is accessible, the system can be called a "multistable" system. As noted previously, the SQERTSS algorithm only throttles the slow process as a last resort, if necessary to achieve computational tractability. In the supporting information, we follow the approach taken by Chatterjee and Voter[38] to justify what happens in the following two cases. 1) When the slow processes do not need to be throttled to achieve computational tractability, the SQERTSS algorithm will not distort the system trajectory towards any accessible steady states. 2) When the slow cases must be throttled to achieve computational tractability, the SQERTSS algorithm is still likely to follow the same trajectory that would be followed in the absence of throttling. In general, SQERTSS is designed to enable simulations that would not otherwise be possible while attempting to minimize the errors given the computational resources available (a paragraph on this philosophy is provided in the supporting information).

Figure 6 and Figure 7 demonstrate the similarity in dynamics between the three throttling scales providing confidence that none of the throttling scales produced significant distortion of this model. By comparison to the unthrottled simulations in those figures, it is shown that the transient behavior of this particular reaction network is well reproduced by the throttled simulations. In this example, even the time progression of the throttled simulation matched that of the unthrottled simulation. The fact that time progression matched so well between the throttled and unthrottled simulations can be regarded as fortuitous. In general, this algorithm is likely to reproduce the same configurational trajectory during transient kinetics, but is likely to distort the time-steps. However, since the configuration-space trajectory should not be distorted, if there is a unique steady-state (for a given reaction network), the throttling algorithm will drive the system toward that steady-state. Thus, for systems that may not otherwise be computationally tractable, this algorithm enables simulating the configuration trajectory towards steady-state, and also (perhaps more importantly) enables simulating the steady-state once it is reached. Even if the steady-state were already known, it would generally be computationally intractable to simulate without throttling for cases where there are FFPs present that span more than several orders of magnitude. Once a steady-state is achieved, it is prudent to try running an additional simulation using that configuration as a starting point and the slow reactions completely unthrottled, in order to verify that the system's dynamics do not change: This verification is included in the algorithm presented here (*SQERTSS Algorithmic Steps to Simulate Steady State*), at the end of the flowchart of Figure 5.

Within the throttling algorithm, the SPs are periodically unthrottled after M snapshots. In this study, periodic unthrottling occurred after every M=10 snapshots. This periodic unthrottling allows for a partial relaxation of the system to occur by slowing down the SPs back to their normal rates for a single snapshot. Thus, during these snapshots, in this example simulation, the slower gas production events would be less likely to occur relative to other throttled snapshots and faster reversible events such as hopping between sites would be more likely to occur (but the time passed during the throttled snapshot would also decrease, so the observed event frequencies of product generation remain unaffected). As the simulation progresses towards steady-state, the FFPs (such as H^+ hopping) remain throttled. Once the simulation has reached steady-state, the throttling factors stabilize after the system has converged, such that further throttling does not occur. Being able to access the steady-state during simulation is a consequence of the throttling algorithm allowing for the FFPs to occur, while not allowing the FFPs to dominate the calculations as they typically would.

There is some flexibility in the final throttling factors achieved: the final set of FFP transition rate constants could be throttled up or down (somewhat) without affecting the system dynamics. Thus, the $EF^{Range\text{-}Req}$ and FFP^{Floor} have order-of-magnitude acceptable ranges of values to achieve the same steady-state without distorting the system dynamics. The FFP^{Floor} prevents the FFPs from becoming too slow relative to processes that may suddenly appear or increase in frequency. When only QE processes are present, the system is in a metastable state which can be exited either due to accessing a rare configuration *or* accessing a long enough time-scale for a rare process. As described earlier in this paper, we do not have a way of predicting which exit path the simulation will take to have a non QE process occur. Slowing down the QE processes will increase the probability of a rare process occurring without distorting the configuration space that is explored. However, if the processes are throttled too far the simulation time may be significantly distorted when a rare configuration is one of the exit paths. We must make a compromise for how far we are willing to throttle the transition probabilities of the QE processes in order to encourage the likelihood of a rare process occurring (keeping in mind that it may instead be a rare configuration that exits the simulation from the state). As an approximation, we have calculated a floor value for how far the QE transition probabilities can be throttled to minimize the distortion of the times associated with the snapshots based on the characteristic timescale of the slowest QE process. The characteristic timescale is often given by the half-life of the process, and for a first order process the solution is:

$$t_{1/2} = ln(2)/k$$

For example, if we want the time-steps to be no larger than 10^{-6} s, then the floor of k takes an approximate value of 10^{6} s^{-1}. This implies that if the system enters a state in which only QE processes exist and remains there for many steps, that the lowest transition rate constant will be throttled no further than 10^{6} s^{-1} and that the time-steps will be on the order of 10^{-6} s. Once a non QE process enters in to the realm of possibility, the KMC time-steps will then become larger, resulting in more time elapsed per snapshot, and the metastable state will also naturally be exited. If we can afford simulating 10^{6} configurations to get out of a metastable state, then we are capable of capturing the exit from any metastable state that lasts 1 second or less. Additionally, the above choices imply that the time distortion will be insignificant provided that the FRP is significantly slower than the FFP^{Floor} during steady-state. These two time-related limits are the ones that become affected by the choice of the floor. In practice, the floor should be chosen based on what timescales are expected to be relevant (i.e., to keep the FFPs much faster than the timescales of interest). In principle, if there is no experimental knowledge of what timescale is relevant, the floor could be chosen empirically (i.e., by trial and error). We chose a floor that was orders of magnitude faster than the timescales we were simulating; then we checked that the floor had no effect by setting the floor more conservatively (i.e., by making the floor higher by two orders of magnitude, from to 10^{6} s^{-1} and 10^{8} s^{-1}) and verifying that this did not change the simulation. The simulations with floors of from to 10^{6} s^{-1} and 10^{8} s^{-1} were identical, which is an indication that the floor value of 10^{6} s^{-1} did not distort the time-steps.

The implemented throttling algorithm is effective at identifying QE FFPs that move in a back and forth direction (e.g. $A \leftrightarrow B$), where QE behavior is identified by investigating the ratio of the occurrence of the forward and reverse processes. When this ratio reflects that the forward and reverse processes are close to equal, a QE is identified. A current limitation of the SQERTSS algorithm is that it is unable to identify non-QE FFPs such as loops (e.g. $A \rightarrow B \rightarrow C \rightarrow A$).

The algorithm was shown to be effective at slowing down QE FFPs (and speeding up SPs) sufficiently to allow for a steady-state production of gas products H_2 and formaldehyde in the reaction system studied, where both gas production processes had steady-state event frequencies on the order of ~1-3 molecules/s. In contrast, the unthrottled simulation resulted in primarily H^+ hopping events and other QE FFPs *even* if a steady-state configuration found by the throttled simulation was fed as a starting point into the unthrottled simulation. The time-steps in the unthrottled simulation were approximately six orders of magnitude smaller. As a consequence, the unthrottled simulation would need to perform ~10^{6} snapshots in order to see between one and several gas molecule production events (which would still not be enough to accurately capture the statistics). Likewise, within the scope of the investigation of this particular chemical reaction network, the maximum relevant simulation-time is on the order of 10 h. With the throttling algorithm simulation, this 10 h simulation-time is computationally feasible in a matter of hours of CPU time and requires on the order of 10^4 snapshots. Without throttling, the computational time (wall clock time) is significantly increased and would require an estimated 10^{11} snapshots to achieve 10 h, which would require on the order of 10^7 hours.

5.2 Comparison of SQERTSS to AS-KMC and comment on migration between multistable steady states

There is a similar method to SQERTSS called accelerated superbasin KMC (AS-KMC).[38] To compare SQERTSS to AS-KMC, we start with connecting the terminology used by Chaterjee and Voter[38] to the terminology used in this work. The system can be considered to reside on a potential energy surface in which the transition between global spatial configurations is a transition between basins (potential energy wells). A superbasin is then a region of the potential energy surface where the system can become temporarily trapped within a set of basins (a set of global configurations) for an extended period of time. Each steady-state achieved in catalytic reactions is actually a superbasin comprising a set of global spatial configurations. These catalytic steady states are non-equilibrated superbasins, but may have some quasi-equilibrated processes within the steady-state. Having connected the terminology, we can now compare SQERTSS to AS-KMC. AS-KMC is to simulate the *migration out of* quasi-equilibrated superbasins. In contrast, SQERTSS is to simulate *migration into* and *within* non-equilibrated superbasins. Thus, the two codes serve different purposes. One important distinction is that because AS-KMC requires quasi-equilibrated superbasins (this is a strict requirement of AS-KMC), the probabilities of finding the system in a particular global spatial configuration can be calculated explicitly using thermodynamics. This is why AS-KMC can be used to calculate the escape from a quasi-equilibrated superbasin, but cannot be used to calculate migration into/out of/within a non-equilibrated superbasin. In catalytic transient states, as well as catalytic steady states, the system is non-equilibrium, so the probability of any global configuration cannot be calculated using thermodynamics and AS-KMC cannot be used. In principle, AS-KMC should be more efficient than SQERTSS (because the staggering does not need to be maintained), and thus AS-KMC should be used over SQERTSS for calculating the escape flux from quasi-equilibrated superbasins. For cases where AS-KMC cannot be used, such as calculating the dynamics or flux into/out of/within a non-equilibrated superbasin, SQERTSS should be used.

Having distinguished between the two methods at a qualitative level, we can now add a comment on the migration between multistable steady states, following the ideas laid forth by Chatterjee and Voter. Consider a chemical system where the system can convert between two bistable catalytic steady states, corresponding to a high catalytic activity FRP rate and a low catalytic activity FRP rate. These represent different regions on the potential energy surface which we can call Regions H and Region L. We can call the larger region of all states external to these regions as Region O. Going from region L to region H (and vice versa) thus requires the system to first migrate into Region O. The frequency of leaving the steady state with a low catalytic FRP (going from Region L to Region O) is then given by:

$$f_{L-O} = \sum_{i} \sum_{j} \pi_{L,i} w_{L,i-O,j}$$

Where $\pi_{L,i}$ is the probability of being located in global configuration i inside region L , and $w_{L,i-O,j}$ is the transition rate constant to global configuration j inside region O. similar expressions can be written for the other terms related to the frequency of leaving and entering Regions L and H (four terms total). Further explanation of this concept of migration between regions, including a figure, is available in the supporting information. For systems where all the transitions are at quasi-equilibrium, the π terms can be calculated *a priori* using thermodynamics (and that is the approach used in AS-KMC). However, when the system is not at quasi-equilibrium -- which is the situation that SQERTSS has been primarily designed for -- it is not possible to calculate the π terms *a priori*. However, it would be possible to make an advance in simulations of these types of systems by combining AS-KMC and SQERTSS. Although the π terms of global configurations within the super basin of a catalytic steady-state are not analytically calculable *a priori*, it would be possible to statistically accumulate the probabilities of each global configuration within a non-equilibrium steady-state region by running SQERTSS in that superbasin for an extended period of time. Thus, the π terms can be calculated *a posteriori* from a SQERTSS simulation. It would then be possible to create *effective* thermodynamics for the non-equilibrium superbasin, which would enable applying AS-KMC to transition out of non-equilibrium superbasins, including for catalytic steady states. Thus, it should be possible to extend AS-KMC to non-equilibrated superbasins using SQERTSS, which would enable even further accelerated simulations of the migrations between bistable (or multistable) non-equilibrium steady states. Such an undertaking would solve (another) currently unsolved problem, but is obviously beyond the scope of the current work.

6. Conclusions

A dynamic throttling algorithm, SQERTSS, based on speed-ranking and identification of quasi-equilibrium fast frivolous processes followed by throttling, has been developed and implemented for use in lattice KMC simulations. The algorithm has been applied to two chemical reaction networks, each containing reaction processes encompassing multiple orders of magnitude in reaction rates. By identifying and dynamically throttling the FFPs and SPs, steady-state conditions have been achieved with a significant improvement in the computational efficiency of the KMC. For the larger reaction network simulated, simulation times > 300 s were easily computed (in minutes), whereas in the unthrottled case, even simulation times on the order of 1 s required an additional several orders of magnitude in computational expense. The algorithm is designed to follow the correct trajectory for system dynamics during both steady-state and transient kinetic periods, and is designed to provide the correct time-steps and product formation event frequencies during steady-state periods. For the systems in this work, the dynamic throttling algorithm was shown to produce the same transient and steady-state output as unthrottled simulations. Though it is beyond the scope of this work, a further acceleration could be achieved for transitions between multistable steady states if AS-KMC were combined with SQERTSS and statistical sampling of the configurations in the relevant steady states.

Acknowledgments

Research sponsored by the Laboratory Directed Research and Development Program of Oak Ridge National Laboratory, managed by UT-Battelle, LLC, for the U. S. Department of Energy.

References

[1] D. Mei, J. Du, M. Neurock, Ind. Eng. Chem. Res., 49 (2010) 10364-103737.
[2] M. Stamatakis, D.G. Vlachos, ACS Catalysis, 2 (2012) 2648-2663.
[3] E.W. Hansen, M. Neurock, Journal of Catalysis, 196 (2000) 241-252.
[4] A. Farkas, F. Hess, H. Over, The Journal of Physical Chemistry C, 116 (2012) 581-591.
[5] A.P.J. Jansen, J.J. Lukkien, Catalysis Today, 53 (1999) 259-271.
[6] O. Deutschmann, Modeling and Simulation of Heterogeneous Catalytic Reactions: From the Molecular Process to the Technical System, Wiley2013.
[7] A.P.J. Jansen, An Introduction to Kinetic Monte Carlo Simulations of Surface Reactions, Springer Berlin Heidelberg2012.
[8] B. Temel, H. Meskine, K. Reuter, M. Scheffler, H. Metiu, The Journal of Chemical Physics, 126 (2007) 204711.
[9] K. Reuter, D. Frenkel, M. Scheffler, Physical Review Letters, 93 (2004) 116105.
[10] R.M. Nieminen, A.P.J. Jansen, Applied Catalysis A: General, 160 (1997) 99-123.
[11] C. Wu, D.J. Schmidt, C. Wolverton, W.F. Schneider, Journal of Catalysis, 286 (2012) 88-94.
[12] D. Hibbitts, E. Dybeck, T. Lawlor, M. Neurock, E. Iglesia, Journal of Catalysis, 337 (2016) 91-101.
[13] S.A. Trygubenko, D.J. Wales, J Chem Phys, 124 (2006) 234110.
[14] C.S. Deo, D.J. Srolovitz, Modelling Simul. Mater. Sci. Eng., 10 (2002) 581-596.
[15] C.D. VanSiclen, J Phys Condens Matter, 19 (2007) 072201.
[16] B. Puchala, M.L. Falk, K. Garikipati, J Chem Phys, 132 (2010) 134104.
[17] M.A. Novotny, Phys Rev Lett, 74 (1995) 1-5.
[18] D.R. Mason, R.E. Rudd, A.P. Sutton, Computer Physics Communications, 160 (2004) 140-157.
[19] D.G. Vlachos, Phys Rev E Stat Nonlin Soft Matter Phys, 78 (2008) 046713.
[20] A. Chatterjee, D.G. Vlachos, Journal of Computational Physics, 211 (2006) 596-615.
[21] D.T. Gillespie, Journal of Chemical Physics, 115 (2001) 1716-1733.
[22] A. Chatterjee, K. Mayawala, J.S. Edwards, D.G. Vlachos, Bioinformatics, 21 (2005) 2136-2137.
[23] A. Chatterjee, D.G. Vlachos, M.A. Katsoulakis, The journal of Chemical Physics, 122 (2005) 024112.
[24] Z. Zheng, R.M. Stephens, R.D. Braatz, R.C. Alkire, L.R. Petzold, Journal of Computational Physics, 227 (2008) 5184-5199.
[25] C.V. Rao, A.P. Arkin, The Journal of Chemical Physics, 118 (2003) 4999.
[26] V.G. Gorskii, M.Z. Zeinaloc, Theoretical Foundations of Chemical Engineering, 37 (2003) 184-190.
[27] A. Agarwal, R. Adams, G.C. Castellani, H.Z. Shouval, J Chem Phys, 137 (2012) 044105.
[28] T.R. Young, J.P. Boris, The Journal of Physical Chemistry, 81 (1977) 2424-2427.
[29] M.A. Snyder, A. Chatterjee, D.G. Vlachos, Computers & Chemical Engineering, 29 (2005) 701-712.
[30] M.J. Hoffmann, S. Matera, K. Reuter, Computer Physics Communications, 185 (2014) 2138-2150.
[31] K.A. Fichthorn, W.H. Weinberg, The Journal of Chemical Physics, 95 (1991) 1090.

[32] D.T. Gillespie, The journal of Chemical Physics, 81 (1977) 2340-2361.
[33] D.T. Gillespie, Journal of Computational Physics, 22 (1976) 403-434.
[34] A.B. Bortz, M.H. Kalos, J.L. Lebowitz, Journal of Computational Physics, 17 (1975) 10-18.
[35] A. Savara, Journal of Physical Chemistry C, Submitted (2016).
[36] A. Savara, Surface Science, 653 (2016) 169-180.
[37] T. Danielson, C. Hin, A. Savara, Journal of Chemical Physics, 145 (2016).
[38] A. Chatterjee, A.F. Voter, Journal of Chemical Physics, 132 (2010).

Appendix I

The throttling algorithm can be expressed in mathematical pseudocode, with the following variable names, where brackets indicate sets and all other variables are scalars.

$EF^{Range\text{-}Req}$:The value for EF^{Range} required for the simulation to achieve relevant simulation timescales within the computational resources and wall clock time provided. The possible ranges are calculated according to the scales in Table 3.
$EF^{Range\text{-}Pred}$:The value for EF^{Range} that is predicted for the next snapshot given a particular throttling scale, based on the first bulleted assumption in Section 3. (i.e. $EF^{Range\text{-}Pred} = TF^{Fastest}EF^{Fastest}/ TF^{Slowest}EF^{Slowest}$, where $TF^{Fastest}$ and $TF^{Slowest}$ are the throttling factors for the fastest and slowest processes)
$[EF^{Range\text{-}Pred\text{-}Allowed}]$: The set of possible $EF^{Range\text{-}Pred}$ that are less than $EF^{Range\text{-}Req}$ given the throttling scales listed in Table 3. $[EF^{Range\text{-}Pred\text{-}Allowed}]$ is thus a subset of the possible ranges from the scales in Table 3 and excludes any scalings that have event frequency ranges greater than $EF^{Range\text{-}Req}$.
(S_F,S_S): The set of throttling scales for fast ($S_F = [0,3]$) and slow ($S_S = [0,3]$) processes
EF^i: Process event frequencies of rank i (this can be one or two processes depending on if there is both a forward and reverse process)
$EF_{Negligible}$: The cutoff such that any process pair whose event frequencies are lower than this value is not considered
Φ is the reaction number that a given process corresponds to.
λ is the direction (F or R) of a given process, for a given reaction.
$P^{\Phi,\lambda}$: The process with reaction Φ and direction λ
$[P]$: The set of all processes.
$EF^{\Phi,F}$: The event frequency for the forward process of reaction Φ
$EF^{\Phi,R}$: The event frequency for the reverse process of reaction Φ
$EF^{\Phi,\lambda}$: The event frequency for a particular process of reaction Φ and direction λ, such that λ can take values of either R or F
$EF^{\Phi,-\lambda}$: The event frequency for the opposite direction of a particular process of reaction Φ and direction λ, such that $EF^{\Phi,-\lambda}$ is the opposite direction of $EF^{\Phi,\lambda}$
EF^{Φ}: The event frequencies pair for reaction number Φ
$[EF^{i-1,F}, EF^{i-1,R}]$: This denotes the pair of forward and reverse processes whose rank, i, is located in the next slowest position to a reference process rank
$[EF^{i,F}, EF^{i,R}]$: This denotes the pair of forward and reverse processes whose rank, i, is the

reference process rank

[$EF^{i+1,F}$, $EF^{i+1,R}$]: This denotes the pair of forward and reverse processes whose rank, i, is located in the next fastest position to a reference process

$[EF]_{\Phi,\lambda}$: list of all individual event frequencies, with indices of Φ and λ

$EF^{Fastest}$: The fastest process event frequency within $[EF]_{\Phi,\lambda}$

$EF^{Slowest}$: The slowest process event frequency within $[EF]_{\Phi,\lambda}$

[FRP]: Set of processes which includes only one process pair, defined as the fastest non-QE process pair

[FFP]: Set of Fast Frivolous processes, includes all QE process pairs which have EFs faster than the FRP (if only QE processes exist, all processes are considered FFPs)

[SP]: Set of rate limiting processes slower than the FRP.

S_F: Throttling scale for FFPs

S_S: Throttling scale for SPs

TF^i: The throttling factor for processes rank i

$RC^{\Phi,\lambda,}$: The original rate constant for reaction Φ, with direction λ

$[RC]_{\Phi,\lambda}$: The set of all current transition rate constants, with indices of Φ and λ

FFP_{Floor}: The minimum EF value to which an FFP will be throttled

$[TF]_{\Phi,\lambda}$ = The all-inclusive list of throttling factors (this has a one to one correspondence with $[EF]_{\Phi,\lambda}$)

$FFP^{step\ down}$: The factor by which processes are throttled down when only FFPs exist (this is a user defined parameter and has been taken as 0.1 for the current work)

The following symbols are used:

$\forall$ means "for all"

$\exists$ means "there exists"

$\nexists$ means "there does not exist"

The syntax $(S_F,S_S) = \{(a,b) \mid EF^{Range\text{-}Pred}(a,b) = Max\ [EF^{Range\text{-}Pred\text{-}Allowed}]\}$ means: choosing values of "a" and "b" for S_F and S_S such that $EF^{Range\text{-}Pred}(a,b)$ is equal to the maximum value in the set $[EF^{Range\text{-}Pred\text{-}Allowed}]$.

The algorithm can be expressed as follows:

0. Set $m = 0$. Run a KMC snapshot with no throttling applied. Increase m by 1.
1. Gather process event frequencies into $[EF]_{\Phi,\lambda}$ from most recent snapshot and classify the processes.
   a. $\forall EF^{\Phi}$, *if* $((EF^{\Phi,F} \& EF^{\Phi,R}) < EF_{Negligible})$ remove $EF^{\Phi,F} \& EF^{\Phi,R}$ from $[EF]_{\Phi,\lambda}$.
   b. For each element of *[P]*, classify processes for each combination of Φ,λ as follows.
      i. classify $P^{\Phi,\lambda} \in [FFP]$ if $(EF^{\Phi,\lambda} \approx EF^{\Phi,-\lambda}$ *and* $[EF^{i,\lambda}, EF^{i,-\lambda}] > Max(FRP))$, or *if* $(EF^{\Phi,\lambda} \approx EF^{\Phi,-\lambda}$ *and* $[FRP]=\emptyset)$
      ii. classify $P^{\Phi,\lambda} \in [FRP]$ if $EF^{\Phi,\lambda} = Max[non\text{-}FFPs]$
      iii. classify $P^{\Phi,\lambda} \in [SP]$ if $(EF^{\Phi,\lambda} \notin [FRP]$ *and* $EF^{\Phi,\lambda} \notin [FFP])$
2. Calculate TFs:
   a. If $\exists P^{\Phi,\lambda} \notin [FFP]$

i. Then for $\forall TF^{i}$, set TF^{i} based on throttling scale $(S_F,S_S) = \{(a,b) \mid EF^{Range\text{-}Pred}(a,b) = Max\ [EF^{Range\text{-}Pred\text{-}Allowed}]\}$

b. If $\forall P^{\Phi,\lambda} \in [FFP]$

i. *If* $EF^{Slowest} > FFP_{Floor,}$ then $\forall TF^{i}, TF^{i} = Min(FFP^{step\ down}, FFP_{Floor}/EF^{Slowest})$

c. If $\mathrm{mod}(m/M) = 0,$

i. Then $\forall [SP]$ set TF^{i}=1 (equivalent to $TS^{SP} = 0$)

3. $\forall RC^{\Phi,\lambda} \in [RC^{\Phi,\lambda}]$, apply $TF^{i} * RC^{\Phi,\lambda}$. Run a snapshot, and increase *m* by 1.
4. Repeat Steps 1-3 until steady state is reached.
5. Run $\geq M$ snapshots with slow processes unthrottled
   a. This means running steps 1-3 repeatedly, except that the “if” statement in Step 2c is removed, such that $\forall [SP]$ set TF^{i}=1 occurs every snapshot.