Journal of Computational Physics 300 (2015) 254–268

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Computationally-efficient stochastic cluster dynamics method for modeling damage accumulation in irradiated materials

Tuan L. Hoang [a,b], Jaime Marian [b,c,*], Vasily V. Bulatov [b], Peter Hosemann [a]

[a] Department of Nuclear Engineering, University of California, Berkeley, CA 94720, USA
[b] Physical and Life Sciences Directorate, Lawrence Livermore National Laboratory, CA 94550, USA
[c] Department of Materials Science and Engineering, University of California, Los Angeles, CA 94720, USA

ARTICLE INFO

Article history:
Received 22 August 2014
Received in revised form 27 July 2015
Accepted 28 July 2015
Available online 3 August 2015

Keywords:
Stochastic cluster dynamics
Irradiation damage
Stochastic simulation algorithm
Tau-leaping

ABSTRACT

An improved version of a recently developed stochastic cluster dynamics (SCD) method (Marian and Bulatov, 2012) [6] is introduced as an alternative to rate theory (RT) methods for solving coupled ordinary differential equation (ODE) systems for irradiation damage simulations. SCD circumvents by design the curse of dimensionality of the variable space that renders traditional ODE-based RT approaches inefficient when handling complex defect population comprised of multiple (more than two) defect species. Several improvements introduced here enable efficient and accurate simulations of irradiated materials up to realistic (high) damage doses characteristic of next-generation nuclear systems. The first improvement is a procedure for efficiently updating the defect reaction-network and event selection in the context of a dynamically expanding reaction-network. Next is a novel implementation of the τ-leaping method that speeds up SCD simulations by advancing the state of the reaction network in large time increments when appropriate. Lastly, a volume rescaling procedure is introduced to control the computational complexity of the expanding reaction-network through occasional reductions of the defect population while maintaining accurate statistics. The enhanced SCD method is then applied to model defect cluster accumulation in iron thin films subjected to triple ion-beam (Fe^{3+}, He^{+} and H^{+}) irradiations, for which standard RT or spatially-resolved kinetic Monte Carlo simulations are prohibitively expensive.

1. Introduction

The production and accumulation of defects in materials subjected to irradiation is a multiscale problem spanning multiple orders of magnitude in time and space. For the last several decades, the rate theory (RT) method for solving coupled ordinary differential equation (ODE) systems has been the workhorse for irradiation damage simulations [1–3], mostly owing to its much greater computational efficiency compared to more detailed methods such as molecular dynamics (MD) or kinetic Monte Carlo (kMC). RT involves solving a set of coupled ODEs such as:

$$\frac{dC_i}{dt} = \dot{\mathcal{F}}_i - \dot{\mathcal{L}}_i, \qquad (i = 1, \ldots, N) \tag{1}$$

* Corresponding author at: Department of Materials Science and Engineering, University of California, Los Angeles, CA 94720, USA.
E-mail address: jmarian@ucla.edu (J. Marian).

http://dx.doi.org/10.1016/j.jcp.2015.07.061
0021-9991/

where each equation describes the time evolution of the average concentration of a particular type (species) of defect cluster denoted by index i. The terms on the right hand side are the loss rate $\dot{\mathcal{L}}_i$ of species i due to various kinetic processes, and the production rate $\dot{\mathcal{F}}_i$ of species i due to irradiation and reactions involving defect cluster species other than i. RT models achieve a high level of simulation efficiency at the cost of drastic simplifications in the underlying physical model, chief of which is the mean-field approximation that neglects spatial correlations and finite volume fluctuations. Another significant reduction in computational complexity is gained by limiting the number of species considered. In practice, the number of admissible defect species (and ODEs in the system) is truncated to achieve a satisfactory balance between accuracy and available computational resources. Large defect clusters not explicitly included in the set are accounted for only approximately (if at all) using a truncation model for the tail of the defect size distribution [4,5].[1] Once defined, the number of ODEs in the set must remain the same through the simulation. To allow simulations to realistically high irradiation doses, this number may need to be as high as 10^6 even in the simplest materials, e.g. pure iron. Furthermore, the number of distinct ODEs that need to be included in the set grows exponentially with increasing number of complex defect cluster types, e.g. simulations of V_mHe_n complexes of m vacancies and n helium atoms requires $(m \times n)$ equations to be included. This is yet another case of combinatorial explosion where the number of equations to be solved is far too large for practical numerical simulations. Consequently, current RT models have been limited to defect populations having no more than two and, in most cases, only one size dimension. This need to allocate an ODE for every possible defect cluster type even before the simulation starts is a serious limitation of the ODE-based RT method.

To overcome these limitations, Marian and Bulatov recently developed the stochastic cluster dynamics (SCD) method to model defect evolution in irradiated materials [6]. The SCD method is based on the stochastic simulation algorithm (SSA) proposed originally by Gillespie for simulations of chemical kinetics in well-stirred systems [7,8]. Whereas RT is formulated in terms of average species concentrations that can take arbitrary fractional values, SSA considers integer-valued species populations in a finite volume and interprets the ODEs defining the RT model as a set of stochastic master equations. The so-defined species population is then evolved stochastically, one reaction at a time, following a standard kMC algorithm. The SSA method has been widely used in the chemical engineering and biochemistry communities [9–13] but is still relatively unknown to computational materials scientists. SCD achieves additional efficiency through the use of dynamic data handling mechanisms where only defect clusters with nonzero populations are kept track of throughout the simulation time. This is a major advantage over RT in which every admissible defect cluster must be allocated a variable and an equation that persist through all stages of ODE integration. Importantly, the computational complexity of a SCD simulation is controlled by the value of the simulation volume and does not depend on the complexity (number of size dimensions) of admissible defect cluster types. Thus, SCD does not suffer from combinatorial explosion and can handle cluster populations with arbitrary number of size attributes. Several proof-of-principle studies have been carried out to demonstrate the applicability of the SCD method to simulations of irradiated materials [6,14].

Although SCD sidesteps combinatorial explosion, the method relies on a kMC algorithm to sample stochastic evolution trajectories from the master equation. Thus, SCD simulations face the usual computational challenges characteristic of kMC simulation methods, such as stiffness caused by a wide spectrum of event rates. Further applications of SCD to technologically relevant materials and irradiation conditions require improvements to make the method more robust and computationally efficient. In this paper, we present several enhancements to SCD, specifically (i) a dynamic reaction-network expansion mechanism to efficiently update the reaction channels and the total reaction rate, (ii) an implementation of the τ-leaping algorithm to accelerate SCD simulations by allowing several reaction events to be leaped over in one single time-step τ, and (iii) a volume scaling method in which the reaction volume is reduced adaptively in order to control the computational cost while preserving statistically significant defect populations. The τ-leaping method [15] was originally developed and used in SSA simulations with fixed variable spaces [13]. In SCD, where the size of the reaction network varies with time, an efficient algorithm for updating noncritical reactions and noncritical species and for computing the leap time is needed to reduce the overhead associated with τ-leaping. We apply the enhanced SCD method to simulations of defect populations in pure iron subjected to triple ion-beam irradiation. The predicted damage accumulation kinetics are verified by comparing them to the original SCD algorithm predictions. The same comparisons are used to quantify gains in computational performance over the original SCD simulations.

The paper is organized as follows. In Section 2, we overview the theory behind the SSA and the τ-leaping methods. In Section 3, we briefly overview our original SCD algorithm, our material model for iron and the types of reaction events considered in our radiation damage simulations. Improvements to the SCD method are described in Section 4 together with their algorithmic details. In Section 5, we present the numerical verification of the new improved SCD algorithm and compare its computational performance to the original algorithm. Finally, Section 6 summarizes our findings.

[1] Existing truncation schemes are ad hoc and unlikely to correctly capture the statistic of extreme values in the defect size distribution believed to be important for understanding material degradation under irradiation.

2. Background

2.1. The Stochastic Simulation Algorithm (SSA)

For clarity, we briefly summarize the SSA method developed by Gillespie for simulations of chemical reactions in well-stirred systems. The reader is referred to the original paper [7] for more details of the method and the theory behind it. Consider a population containing N defect-clusters $\{S_1, S_2, .., S_N\}$ that can participate in M reaction channels $\{\mathbb{R}_1, \mathbb{R}_2, \ldots, \mathbb{R}_M\}$. Let $\overrightarrow{X}(t)$ be the dynamic state vector of the system at an arbitrary time t, $\overrightarrow{X}(t) = \{X_1(t), X_2(t), \ldots, X_N(t)\}$, where $X_i(t)$ is the number of defect clusters of type S_i at time t. Each reaction channel is characterized by its reaction rate R_j and by its state change vector $\overrightarrow{\nu}_j = (\nu_{1j}, \nu_{2j}, \ldots, \nu_{Nj})$. The probability that a reaction of type j will take place within the next infinitesimal time interval $[t, t+dt)$ is given by the product $R_j dt$ whereas ν_{ij} specifies the change in the population of species S_i after a single reaction event along channel $\mathbb{R}_j$. The evolution of such reaction network obeys the following chemical master equation (CME)

$$\frac{\partial P(\overrightarrow{x}, t|\overrightarrow{x_0}, t_0)}{\partial t} = \sum_{j=1}^{M} [R_j(\overrightarrow{x} - \overrightarrow{\nu_j}) P(\overrightarrow{x} - \overrightarrow{\nu_j}, t|\overrightarrow{x_0}, t_0) - R_j(\overrightarrow{x}) P(\overrightarrow{x}, t|\overrightarrow{x_0}, t_0)] \tag{2}$$

where $P(\overrightarrow{x}, t|\overrightarrow{x_0}, t_0)$ is the conditional probability that $\overrightarrow{X}(t) = \overrightarrow{x}$ at time t if $\overrightarrow{X}(t_0) = \overrightarrow{x_0}$ at time t_0. The above CME defines a stochastic process referred to as a continuous time Markov chain. Rather than attempting to solve this CME equation directly, individual stochastic time trajectories of the state vector $\overrightarrow{X}(t)$ can be obtained using an appropriate kinetic Monte Carlo algorithm. In particular, in the following algorithm two random numbers r_1 and r_2 uniformly distributed in $(0, 1)$ are generated. The time to the next reaction event is then given by

$$\Delta t = -\frac{1}{\sum_j R_j} \log(r_1) \tag{3}$$

and the index of the same reaction event, R_k, is taken to be he smallest integer k that satisfies the following condition

$$\sum_{i=1}^{k} R_i > r_2 R_{tot} \tag{4}$$

where $R_{tot} = \sum_j^M R_j$, which is the sum of all the individual reaction rates in the volume. Once the next reaction event and its time increment are selected, the simulation time and the state vector are updated accordingly, $t = t_0 + \Delta t$ and $\overrightarrow{X}(t_0 + \Delta t) = \overrightarrow{X}(t_0) + \overrightarrow{\nu_k}$. The simulation proceeds to the next reaction event until the desired simulation time is reached. The method just described is referred to as *direct* SSA method. The direct SSA method rigorously generates stochastic trajectories sampled for the exact (even if often unknown) solutions of the CME. Several algorithmic enhancements have been proposed to improve efficiency of the direct SSA method, including the *first reaction* method [7], the *modified* direct method [10], the *optimized* direct method [16], the *sorting* direct method [17], or the *logarithmic* direct method [18], to name a few. Any such improvements notwithstanding, simulating every reaction event one at a time is often impractical for large reaction networks of practical interest. To address this problem, Gillespie proposed the τ-leaping method that allows many reactions channels to fire in a single time-step at the expense of some minor accuracy loss. Because conditions that justify the using of τ-leaping are often met in radiation damage simulations, in the following we briefly describe τ-leaping as a way to accelerate stochastic simulations.

2.2. The τ-leaping method

The τ-leaping method is based on the leap condition which assumes that a reaction channel may be fired multiple times within a small time interval $[t, t+\tau)$ if the reaction rate does not suffer significant changes over that interval. Then, given the state vector of the system $\overrightarrow{X}(t) = \overrightarrow{x}$, the number of times that each reaction channel $\mathbb{R}_j$ can fire is approximated by the Poisson distribution $\mathcal{P}(R_j\tau)$. The simulation proceeds as follows: (i) at each time-step we find a value of τ that satisfies the leap condition mentioned above; (ii) for each ν_j, a Poisson random number with mean $R_j\tau$, i.e. $\mathcal{P}(R_j\tau)$ is generated; (iii) the system is updated as $\overrightarrow{X}(t+\tau) \leftarrow \overrightarrow{X}(t) + \sum_j^M \mathcal{P}(R_j\tau)\nu_j$, and the simulation time advances to the new time $t \leftarrow t + \tau$. As a result, the simulation can be accelerated at a greater speed since it can leap through multiple reactions in one single step instead of firing the reactions one by one.

3. The Stochastic Cluster Dynamics algorithm

3.1. Model representation

At any point in time the state of the model is characterized by the set of all existing clusters $\overrightarrow{S}_{all} = \{S_i\}$. Dynamic updates of state vectors are efficiently handled using hash tables with dynamic resizing. More details on the hash functions and associated operations are given in the next section. Each cluster S_i contains several associated attributes such as the number of each component species contained in the cluster, the cluster species population count, its diffusion coefficient, the binding energies among the component subspecies and the cluster, and other relevant parameters. Mobile species with a nonzero diffusivity are regarded as a subset $\overrightarrow{S}_m$ (here m stands for mobile) of $\overrightarrow{S}_{all}$. Defect cluster species associated with a recently executed event are stored in a dynamic array whose purpose will be described in the following section. Such species can be reactants or products of a recently executed reaction event or a collection of defects and clusters that have just been introduced into the volume as a result of a defect insertion event (due to irradiation). The evolving reaction network $\overrightarrow{\mathbb{R}} = \{\mathbb{R}_i\}$ specifies all reaction channels available for the current defect population $\overrightarrow{S}_{all}$. Each binary reaction channel $\mathbb{R}(S_1, S_2)$ represents a reaction between species of type S_1 and type S_2 with an associated reaction rate $R(S_1, S_2)$ (clusters S_1 and S_2 can be identical when the reaction involves two like species). To implement the τ-leaping method, two more data sets will be defined. The first set $\overrightarrow{\mathbb{J}} = \{\mathbb{J}_i\}$ contains all noncritical reaction channels whose associated reactants have populations larger than a certain user-predefined value n_{cr}. Another set $\overrightarrow{P} = \{P_i\}$ contains all defect cluster species associated with the noncritical reactions. Each P_i contains a parameter specifying the highest order of possible reactions species i can participate in, as explained in more details later these reaction order parameters are utilized in computing the leap time τ.

3.2. Types of events

Hereafter, V_s and I_s denote a vacancy cluster or a self interstitial atom (SIA) cluster of size s. In our model, we only consider clusters with a maximum of three component species, specifically the clusters only contain He and H atoms together with either vacancies or interstitials of the host material. If desired, the model can be modified to admit defect clusters of arbitrarily complex compositions. The following reactions are currently admitted in our SCD model of iron:

0th-order reactions

- Defect insertion, e.g. generation of certain types of defects resulting from collisions of incoming energetic particles with the host matrix atoms.

1st-order reactions

- Defect absorption at sinks: mobile clusters can migrate towards sinks and become absorbed there. Sinks can be free surfaces, dislocation networks or grain boundaries.
- Emission of a monomer from a defect cluster: a cluster can emit a monomer of one of its constituent species, reducing its species count appropriately. A complex cluster $V_iHe_jH_k$ can emit a vacancy, or a He monomer or a H monomer. Following emission, the initial cluster's population is reduced by one and two new defect species are created or, if one or both species already exist, their counts are increased by one. For example, emission of one vacancy V (or one He monomer) produces a smaller defect cluster $V_{(i-1)}He_jH_k$ (or $V_iHe_{(j-1)}H_k$ in case the monomer is a He atom).

2nd-order reactions

- Defect annihilation: collisions of two clusters containing vacancies and self interstitial atoms result in their complete or partial recombination. For example, collision of a complex vacancy cluster $V_iHe_jH_k$ with a SIA cluster $I_{i'}$ produces $V_{(i-i')}He_jH_k$ (if $i > i'$) or $I_{(i'-i)}He_jH_k$ (if $i' > i$) or releases j monomers of He and k monomers of H monomers (if $i = i'$). He and H monomers are assumed not to bind unless vacancies and interstitials are also present.
- Defect aggregation: clusters containing like defects can combine to form larger clusters upon interaction. For example, a $V_iHe_jH_k$ cluster can collide with a $V_{i'}He_{j'}H_{k'}$ cluster producing a larger $V_{(i+i')}He_{(j+j')}H_{(k+k')}$ cluster.

3.3. Summary of the original SCD algorithm

The main motivation for the development of SCD was to circumvent combinatorial explosion in the number of equations encountered in traditional ODE-based RT simulations. In SCD, the simulation volume is finite and defect cluster species

have integer-valued populations. In a typical initial state, relatively few (if any) defect species exist, so, rather than allocating memory for all possible defect-clusters before the start of the simulation using a regular indexed array, cluster species are added or removed from the hash table dynamically, as needed. Therefore, only defect clusters that have nonzero populations are kept track of. The hash table is implemented as an associative array in which a hash function is used to map the identifying values – known as hash *keys* – to their associated values. In our model, hash keys that represent defect clusters are number strings comprised of the numbers of component species that form the clusters. For example, allowing for a maximum size of 1000 for each component species, a cluster formed by nine vacancies, one He atom and three H atoms will have the following key: −9001003, while 81011036 represents a cluster with 81 SIAs, 11 He atoms and 36 H atoms. More specifically, the first three digits of the key string represent the number of H atoms in the cluster, the next three digits represent the number of He atoms which are finally followed by the number of either vacancies or SIAs contained in the cluster. Here, we assign a negative value for clusters that are made of vacancies and positive value in the case of SIAs. These hash keys can be modified accordingly if there are more species implanted into the volume. The hash function maps the keys onto the index array elements (or buckets) where the associated values are stored. Operations on a hash such as adding, removing or locating buckets take constant time on average and do not depend of the size of the hash itself unlike operations on indexed arrays. In simulations of irradiated materials, the number of pre-existing defect clusters is usually small but increases rapidly after high energy particles begin to create defects. Furthermore, defect populations and their associated reaction channels change with each subsequent reaction event. It is our experience that in such conditions hashing is more efficient than using array structures for handling large and evolving data sets since defect clusters can be located and updated quickly. The original SCD algorithm consists of the following steps:

1. Construct two hash tables: one, $\overrightarrow{S}_{all}$, to store all the existing defect clusters and another one, $\overrightarrow{S}_{m}$, to store only the mobile defects in $\overrightarrow{S}_{all}$.
2. Construct a reaction table $\overrightarrow{\mathbb{R}}$ containing the reaction channels involving all existing defect clusters, and store $\overrightarrow{\mathbb{R}}$ in an array.
3. Calculate the total reaction rate by summing the rates of all currently existing reaction channels in the reaction table.
4. Randomly select the time increment to the next reaction as well as the type of the reaction event using Eqs. (3) and (4).
5. Execute the selected reaction event, update the hash tables accordingly and delete the reaction table $\overrightarrow{\mathbb{R}}$.
6. Return to step 2 and proceed until the total simulation time is reached.

Using an array to store the reaction channels $\overrightarrow{\mathbb{R}}$ proves to be inefficient due to the highly dynamic nature of stochastic evolution. Furthermore it is wasteful to build the reaction table anew after every reaction event since only a portion of the reaction channels is changed due to the executed event while most others are left intact. These two inefficiencies are addressed in the improved version of SCD presented in the following section.

4. An improved Stochastic Cluster Dynamics algorithm

Except for massive defect insertion events representing collision cascades, only a small number of defect clusters in the simulation volume are affected by a single reaction event. Therefore, only the reaction channels involving affected defect species need to be updated, while the rest of the reaction network remains untouched. In this enhanced version of SCD, we use hashing to maintain existing species and reaction channels and to expand the reaction network when new species are introduced by the reaction events. Such updates are typically more efficient than the reconstruction of the entire reaction table in between insertion events. Depending on the specific reaction model implemented, some defect species become quite numerous and their associated reaction channels can fire much more frequently than others. For example in our model for iron, SIAs and vacancies are observed to migrate in large numbers to defects sinks soon after irradiation commences, whereas defect insertion and defect association events are relatively infrequent. To expedite SCD simulations under such conditions, we implement a version of τ-leaping method in which several repetitive reaction events are executed at once. Lastly, we introduce and justify a volume rescaling procedure to reduce the computational complexity of SCD simulations at later stages of damage accumulation. This is when the density of defect clusters becomes high, and the diffusion length of mobile defects becomes small compared to the linear dimension of the simulation volume.

4.1. Dynamic reaction network updates and expansion

As follows from Eqs. (3) and (4), both the time increment to the next reaction event and the type of reaction are selected based on the total event rate summed over all existing reaction channels. In the original version of SCD, the net event rate was recomputed after each reaction event throughout the simulation. However, in a production scale SCD simulation the number of distinct reaction rates grows rapidly to thousands and even millions and yet only a small sub-set of reaction

channels is directly affected by each reaction event. Enabling incremental updates requires that reaction channels affected (modified or eliminated) by the last event be located and updated in the computer's memory efficiently during the course of the simulation. We rely on hashing to quickly add, remove, locate and update reaction channels in real time.

In the improved version of SCD reported here, in addition to the two hash tables $\overrightarrow{S}_{all}$ and $\overrightarrow{S}_{m}$ used to store and reference the total and the mobile cluster populations, all existing reaction channels are stored in a reaction hash $\overrightarrow{\mathbb{R}}$. These reactions are also represented by different hash keys. However, different from integer keys that represent defect clusters, the reactions will have structure keys comprised of the keys of all defect clusters that participate in the reactions. The reaction table expands or contracts as needed to accommodate new reactions associated with the creation (or extinction) of new defect species. The process for updating the affected hash tables goes as follows:

- A new hash key is created for all possible species resulting from these reactions. For each 2nd-order reaction, this key is generated from the keys of its constituent reactants stored in $\overrightarrow{S}_{all}$ while for 0th and 1st-order reactions, dummy keys – two and one, respectively – are used as appropriate.
- Each cluster in the $\overrightarrow{S}_{all}$ hash table is assigned a parameter f_1 indexing its count change due to the recently executed event; another parameter f_2 indicates whether the defect already existed in the simulation volume in the previous time-step. These parameters let SCD know whether it should look up and update the existing reaction $\mathbb{R}(S_1, S_2)$ or add it as a newly created one into the reaction hash table $\overrightarrow{\mathbb{R}}$.
- As a cost-savings measure, defect clusters that have participated in a recent reaction event are stored in a dynamic array so that product species can be updated efficiently. As the number of these clusters is not very large, a dynamic array is simpler than a hash table in this case.

4.2. Reaction rate updating

The first step of the reaction update process is to visit each reaction in $\overrightarrow{\mathbb{R}}$ and remove those whose component reactants no longer exist due to the previous event(s). Subsequently, we visit each element S_i in $\overrightarrow{S}_{all}$ and update all the reaction channels that this cluster associates with. As a result, some existing reactions in S_i will be modified and new reactions will be added into the reaction hash table $\overrightarrow{\mathbb{R}}$.

Based on the values of f_1 and f_2 mentioned previously, it can be established whether a cluster was a reactant or product of the last reaction event. If S_i is a new defect cluster, all the reactions associated with it will be added directly into the reaction hash table because it is not necessary to check for their existence in it. On the other hand, if the cluster S_i only increases or decreases in number, all of its associated reactions will be first located in the reaction hash and updated accordingly based on the value of f_1. If f_1 is the change in population of cluster S_i and R is the rate of a reaction channel involving S_i, then the total reaction rate R_{tot} can be updated as follows:

1st-order reaction:

$$R(S_i) \leftarrow R_0(S_i)\left[1+\frac{f_1(S_i)}{X_0(S_i)}\right], \qquad R_{tot} \leftarrow R_{tot} + R_0(S_i)\frac{f_1(S_i)}{X_0(S_i)} \tag{5}$$

For 2nd-order reactions between a cluster S_i and another cluster S_j (assuming $\mathbb{R}(S_i, S_j)$ already exists):

$$\begin{cases} R(S_i,S_j) \leftarrow R_0(S_i,S_j)\left[1+\frac{f_1(S_i)}{X_0(S_i)}\right]\left[1+\frac{f_1(S_j)}{X_0(S_j)}\right] & (S_i \neq S_j) \\ R(S_i,S_j) \leftarrow R_0(S_i,S_j)\left[1+\frac{f_1(S_i)}{X_0(S_i)}\right]\left[1+\frac{f_1(S_i)}{X_0(S_i)-1}\right] & (S_i \equiv S_j) \end{cases} \tag{6}$$

$$\begin{cases} R_{tot} \leftarrow R_{tot} + R_0(S_i,S_j)\left[\frac{f_1(S_i)}{X_0(S_i)}+\frac{f_1(S_j)}{X_0(S_j)}+\frac{f_1(S_i)f_1(S_j)}{X_0(S_i)X_0(S_j)}\right] & (S_i \neq S_j) \\ R_{tot} \leftarrow R_{tot} + R_0(S_i,S_j)\left[\frac{f_1(S_i)}{X_0(S_i)}+\frac{f_1(S_i)}{X_0(S_i)-1}+\frac{f_1^2(S_i)}{X_0(S_i)[X_0(S_i)-1]}\right] & (S_i \equiv S_j) \end{cases} \tag{7}$$

where $R_0()$ and $X_0()$ are the equivalent old reaction rate and old population. Therefore, the reaction rate updates depend only on values of the f_1 parameters and the old populations of the clusters.

If both f_1 and f_2 are zero, the cluster S_i is not affected by the selected event, but we need to check whether S_i can engage in any 2nd-order reactions with those clusters S_j that are affected by the recent reaction event. These clusters are stored in the dynamic array mentioned previously. The last step of this process is to check whether the clusters in the dynamic array can form 2nd-order reactions with one another. Some of these reactions, which may have been skipped in previous steps because of the way the reaction *keys* are assigned, are now accounted for in this step. If any pair of defect-clusters S_i and S_j can react, the corresponding reactions – as well as the total reaction rate – can be updated accordingly using Eqs. (5), (6), and (7).

4.3. Implementation of the τ-leaping method in SCD

In this section, we describe our implementation of the τ-leaping method within SCD. The method has been previously implemented on top of the direct SSA algorithm [15]. However, implementation of τ-leaping in an open system where new species are constantly added to or removed from the reaction network, as is the case of SCD simulations of irradiated materials, has not been attempted to our knowledge. Employing hash tables, we now show how τ-leaping can be added to the SCD method to make the simulations more efficient. Several improvements have been proposed to the τ-leaping method since it was first proposed by Gillespie [15,19–23], including efficient simulations of stiff reaction networks [24–26] or prevention of meaningless negative species populations that can be caused by leaping [27]. Cao et al developed an efficient τ-leaping SSA algorithm that avoids having to solve a complicated set of partial differential equations suggested in Ref. [21]. For the sake of clarity and to better explain our implementation of τ-leaping in the SCD algorithm, here we briefly summarize Cao et al.'s algorithm. The reader is referred to the original paper for more details of the method and the underlying theory [21].

In Cao et al.'s approach, the set of all existing reactions is divided into two non-overlapping subsets: the critical subset includes all reactions that are within n_{cr} (a pre-defined integer) firings away from extinguishing one of the component reactants and the noncritical subset includes all the other reactions. We add all 0th-order defect insertion reactions to the critical subset in which every reaction is advanced one at a time, just like in the *direct* SSA method. To enable efficient τ-leaping over the noncritical reaction subset, we make use of two more hash tables. The first one $\overrightarrow{P}$ is used to store the noncritical species, each element in $\overrightarrow{P}$ containing the species' attributes such as its *key*, population count and several additional parameters g, μ, σ^2 and O's as defined below. The second hash table $\overrightarrow{\mathbb{J}}$ contains the noncritical reactions $\mathbb{J}_j$ with corresponding reaction rates J_j. Here, $\overrightarrow{\mathbb{J}}$ is a subset of the all reaction hash $\overrightarrow{\mathbb{R}}$. Therefore, we use another set of notations to distinguish these noncritical reactions from the regular reactions for the sake of clarity. Similar to the regular reaction hash, each element of $\overrightarrow{\mathbb{J}}$ contains the key and the rate of a noncritical reaction. Following Cao et al.'s approach, a safe leap time τ for every noncritical reaction in $\overrightarrow{\mathbb{J}}$ is selected as

$$\tau' = \min_{P_i \in \overrightarrow{P}} \left\{ \frac{\max\{\epsilon X(P_i)/g(P_i), 1\}}{|\mu(P_i)|}, \frac{\max\{\epsilon X(P_i)/g(P_i), 1\}^2}{\sigma^2(P_i)} \right\} \tag{8}$$

with

$$\mu(P_i) = \sum_{\mathbb{J}_j \in \overrightarrow{\mathbb{J}}} \nu_{ij} J_j \tag{9}$$

$$\sigma^2(P_i) = \sum_{\mathbb{J}_j \in \overrightarrow{\mathbb{J}}} \nu_{ij}^2 J_j \tag{10}$$

The value of g_i depends on the highest order O_i of any reaction in which the noncritical cluster P_i appears as a reactant. As appropriate for our model of irradiated materials, we categorize these reaction-order parameters into three different types: 1st-order (O_1), 2nd-order (O_2), and 2nd-order with like reactants (O_3). When a reaction becomes critical or no longer exists due to exhaustion of one or both of its reactants, O_i parameters of the participating reactants are updated accordingly. The values of μ and σ^2 for each P_i are also updated every time a reaction involving a noncritical cluster is analyzed.

To determine the value of the leap time τ, our algorithm inspects all clusters P_i stored in the noncritical-reactant hash $\overrightarrow{P}$, determines the highest order of their associated reactions and calculates the corresponding values of g_i. When $O_2(P_i) = 0$, the highest order of reactions involving species P_i is 1st-order, and the corresponding value of $g(P_i)$ is 1. A positive $O_2(P_i)$ indicates that P_i takes part in at least one 2nd-order reaction in which case $g(P_i)$ is taken to be 2. However when a reaction exists that can involve two clusters of species P_i, $O_3(P_i)$ will also be positive and the value of $g(P_i)$ is determined instead as

$$g(P_i) = \left[2 + \frac{1}{X(P_i) - 1} \right] \tag{11}$$

After a safe value of the leaping time is estimated as described above, the number of times k_i each reaction $\mathbb{J}_i \in \overrightarrow{\mathbb{J}}$ in the noncritical reaction hash will fire during this interval is computed as a Poisson random variable $\mathcal{P}(J_i\tau)$. However the reactions are not executed immediately as it is still necessary to ensure that none of the noncritical reactant populations $\overrightarrow{P}$ becomes negative after τ-leaping is performed on all reactions in the current noncritical reaction hash $\overrightarrow{\mathbb{J}}$. To ensure that all species populations remain non-negative after τ-leaping, the total number $k^{tot}(P_i)$ of reaction events reducing the population of species P_i is obtained by summing k_j over all noncritical reactions $\mathbb{J}_j$ consuming P_i during the leap time τ. Only when the population of every noncritical cluster $X(P_i)$ is found to be larger than $k^{tot}(P_i)$, every reaction $\mathbb{J}_i$

stored in the noncritical reaction hash is executed k_i times and the f_1 and f_2 parameters of reactant clusters S_1 or S_2 are updated accordingly; otherwise, the value of τ is reduced, new firing times k_i's are determined and the previous non-negativity condition is re-examined. Should the need for reduction in τ persist, τ-leaping is abandoned in favor of the *direct* (single-reaction) SSA algorithm for some number of SSA steps (200 steps in simulations described in Section 5) after which τ-leaping is resumed.

4.4. Controlling the simulation complexity using volume rescaling

The computational complexity of SCD simulations is largely defined by the number of distinct cluster species currently present in the defect population. This number can be controlled by the size of the simulation volume. In selecting the volume, one needs to balance two conflicting requirements: (1) defect cluster populations should be statistically representative (which favors larger volumes) and (2) the computational cost of SCD simulations should remain acceptable (which favors smaller volumes). Here we introduce a method to balance these two requirements through volume rescaling.

Typically, at the start of a SCD simulation, most of defect clusters are mobile and their volume concentrations as well as the concentration and the net strength of pre-existing defect sinks (dislocations, grain boundaries, for example) are low. Under such conditions, mobile defects diffuse over long distances through the reaction volume before they meet a reaction partner. However, as time proceeds and progressively more defects are inserted by continued irradiation, clusters become more numerous while smaller mobile clusters combine and form increasingly larger clusters. Such kinetics result in a more or less steady reduction in the lifetime and diffusion length of mobile clusters defined as the average time and distance traveled by a mobile cluster from birth to death, respectively. In a given defect population, the average lifetime of a mobile cluster of species S_i is the inverse of the total rate of loss:

$$\mathcal{L}(S_i) = D(S_i)\sum_l Z_{il}\rho_l + \epsilon_m + \sum_j k_{ij}\frac{X(S_j)}{V}, \tag{12}$$

while the maximum diffusion length among all mobile cluster species $\overrightarrow{S'}_m$ can be estimated as

$$l_{max} = \max_{S_i \in \overrightarrow{S'}_m} l_{S_i}, \quad l_{S_i} = \sqrt{\frac{D(S_i)}{\mathcal{L}(S_i)}} \tag{13}$$

where $\mathcal{L}(S_i)$ is the net rate of loss and $D(S_i)$ is the diffusion coefficient of the mobile cluster species S_i, Z_{il} is the strength of a given sink of type l with respect to the same species (the sink's ability to remove clusters S_i), ρ_l is the volume density of sinks of type l, ϵ_m is the total rate of all dissociation reactions leading to splitting clusters S_i, and k_{ij} is the reaction rate of a 2nd-order reaction between the mobile defect S_i and the defect cluster of type S_j with a population of $X(S_j)$. Here $\overrightarrow{S'}_m$ denotes the set of all mobile species that will possibly appear in the simulation volume, not limited to only those exist at the current time-step.

The significance of parameter l_{max} is that it defines the range of distances beyond which neighboring reaction subvolumes are no longer exchanging their reactants (defect clusters). Thus, reaction volumes with linear dimensions exceeding l_{max} can be viewed as causally isolated from each other. Typically, as a SCD simulation progresses l_{max} decreases due to a more or less steady increase in the magnitude of the last term on the right hand side of Eq. (12). A significant reduction in l_{max} justifies an appropriate reduction in the reaction volume, $V_{new} = \gamma V_{old} \geq l_{max}^3$ (with $\gamma < 1$). The essence of our volume rescaling method is that when conditions for volume reduction conditions are satisfied, the cluster population is reduced by allowing every cluster to be randomly eliminated with probability $(1-\gamma)$ before resuming the SCD simulation. Such a volume reduction procedure allows to maintain the size of the reaction network approximately constant even when damage accumulation increases the volume density of defects by orders of magnitude. However, volume rescaling should be avoided when there are large fluctuations in the defect population, for example right after a massive defect insertion event.

4.5. Algorithm implementation

In this section we present the key algorithmic elements of our improved SCD method in pseudocode format, including construction of hash tables for noncritical reactions and defect clusters and an algorithm for estimating a safe leap time τ in SCD. In the following $R(S_1, S_1)$ is the rate of a binary reaction $\mathbb{R}(S_1, S_2)$ between species S_1 and S_2. Similarly, $J(P_1, P_2)$ denotes the rate of the noncritical reaction $\mathbb{J}(P_1, P_2)$ between two noncritical species P_1 and P_2. The set of all critical reactions is represented by $\overrightarrow{\mathbb{R}}_{cr}$. For a 1st-order reaction $\mathbb{R}(S_1)$ or $\mathbb{J}(S_1)$, S_1 represents its one and only reactant cluster. $X(S_i)$ denotes the population (number of units) of cluster species S_i in the reaction volume. Finally, X_{min} denotes the smaller population of clusters among the two reactants of the reaction $\mathbb{R}(S_1, S_2)$.

4.5.1. Construction of the noncritical hash tables

1. If $\mathbb{R}$ is a 2nd-order reaction:
   (a) If $S_1 \equiv S_2$ (reaction between two like clusters):
      i. If $X(S_1) >= n_{cr} + 2$ (reaction $\mathbb{R}$ is noncritical, and cluster S_1 is noncritical, i.e. $\mathbb{R}(S_1) \equiv \mathbb{J}(S_1) \in \overrightarrow{\mathbb{J}}$ and $S_1 \equiv P_1 \in \overrightarrow{P}$)
         - If $\mathbb{J}(P_1)$ does not exist: add $\mathbb{J}(P_1)$ into the noncritical reaction hash $\overrightarrow{\mathbb{J}}$ and update $O_{2,3}(P_1) \leftarrow O_{2,3}(P_1) + 1$.
         - Else: update $J(P_1) \leftarrow R(S_1)$ and $X_{min}(\mathbb{J}) \leftarrow X(S_1)$, reset $k[\mathbb{J}(P_1)] \leftarrow 0$.
         - Locate P_1 in the noncritical cluster hash $\overrightarrow{P}$
            – If P_1 does not exist: add P_1 to $\overrightarrow{P}$ with $\mu_{tot}(P_1) \leftarrow \mu[J(P_1)]$ and $\sigma^2_{tot}(P_1) \leftarrow \sigma^2[J(P_1)]$ as determined by Eqs. (9) and (10).
            – Else: update $\mu_{tot}(P_1) \leftarrow \mu_{tot}(P_1) + \{\mu[J(P_1)]\}_{new} - \{\mu[J(P_1)]\}_{old}$ and $\sigma^2_{tot}(P_1) \leftarrow \sigma^2_{tot}(P_1) + \left\{\sigma^2[J(P_1)]\right\}_{new} - \left\{\sigma^2[J(P_1)]\right\}_{old}$.
      ii. Else ($\mathbb{R}$ is critical):
         - If $\mathbb{P}(P_1)$ exists: locate P_1 in $\overrightarrow{P}$. If P_1 exists: update $O_{2,3}(P_1) \leftarrow O_{2,3}(P_1) - 1$, $\mu_{tot}(P_1) \leftarrow \mu_{tot}(P_1) - \mu[J(P_1)]$, $\sigma^2_{tot}(P_1) \leftarrow \sigma^2_{tot}(P_1) - \sigma^2[J(P_1)]$. If $O_1(P_1) = O_2(P_1) = O_3(P_1) = 0$, remove P_1 from the noncritical cluster hash $\overrightarrow{P}$.
         - Remove $\mathbb{J}(P_1)$ from the noncritical reaction hash $\overrightarrow{\mathbb{J}}$.
   (b) Else (reaction between unlike species $S_1 \neq S_2$):
      i. If $min\{X(S_1), X(S_2)\} > n_{cr}$ (reaction $\mathbb{R}$ is noncritical, and clusters S_1, S_2 are noncritical, i.e. $\mathbb{R}(S_1, S_2) \equiv \mathbb{J}(S_1, S_2) \in \overrightarrow{\mathbb{J}}$ and $S_{1,2} \equiv P_{1,2} \in \overrightarrow{P}$)
         - If $\mathbb{J}(P_1, P_2)$ does not exist: add $\mathbb{J}(S_1, S_2)$ into the noncritical reaction hash $\overrightarrow{\mathbb{J}}$, update $O_2(P_{1,2}) \leftarrow O_2(P_{1,2}) + 1$.
         - Else: update $J(P_1, P_2) \leftarrow R(S_1, S_2)$ and $X_{min}(\mathbb{J}) \leftarrow min\{X(S_1), X(S_2)\}$, reset $k[\mathbb{J}(P_1, P_2)] \leftarrow 0$.
         - Locate P_1 and P_2 in the noncritical cluster hash $\overrightarrow{P}$
            – If P_1 and/or P_2 do not exist: add them into $\overrightarrow{P}$ with $\mu_{tot}(P_{1,2}) \leftarrow \mu[J(P_1, P_2)]$ and $\sigma^2_{tot}(P_{1,2}) \leftarrow \sigma^2[J(P_1, P_2)]$ as determined by Eqs. (9) and (10).
            – Else: update $\mu_{tot}(P_{1,2}) \leftarrow \mu_{tot}(P_{1,2}) + \{\mu[J(P_1, P_2)]\}_{new} - \{\mu[J(P_1, P_2)]\}_{old}$ and $\sigma^2_{tot}(P_{1,2}) \leftarrow \sigma^2_{tot}(P_{1,2}) + \left\{\sigma^2[J(P_1, P_2)]\right\}_{new} - \left\{\sigma^2[J(P_1, P_2)]\right\}_{old}$.
      ii. Else: ($\mathbb{R}$ is critical):
         - If $\mathbb{J}(P_1, P_2)$ exists: locate P_1 and P_2 in $\overrightarrow{P}$. If P_1 or P_2 exists: update $O_2(P_{1,2}) \leftarrow O_2(P_{1,2}) - 1$, $\mu_{tot}(P_{1,2}) \leftarrow \mu_{tot}(P_{1,2}) - \mu[J(P_1, P_2)]$, $\sigma^2_{tot}(P_{1,2}) \leftarrow \sigma^2_{tot}(P_{1,2}) - \sigma^2[J(P_1, P_2)]$. If $O_1(P_{1,2}) = O_2(P_{1,2}) = O_3(P_{1,2}) = 0$, remove P_1 and/or P_2 from the noncritical cluster hash $\overrightarrow{P}$.
         - Remove $\mathbb{J}(P_1, P_2)$ from the noncritical reaction hash $\overrightarrow{\mathbb{J}}$.
2. Else: $\mathbb{R}$ is a 1st-order reaction (emission or absorption of a defect cluster at sinks). Follow similar steps as described in 1(a), except that the noncritical condition for cluster S_1 is $X(S_1) > n_{cr}$ in this case.

4.5.2. Reaction update loop

1. Remove all illegal reactions whose reactants are no longer exist from the reaction hash $\overrightarrow{\mathbb{R}}$. Starting from the first cluster S_1 in the all-cluster hash $\overrightarrow{S}_{all}$:
2. If $f_1(S_1) \neq 0$ or $f_2(S_1) \neq 0$ (S_1 is affected by the recently executed event):
   (a) If $f_1(S_1) \neq 0$ and $f_2(S_1) = 0$ (S_1 is only the reactant not the product of the previously executed event):
      i. Check if S_1 is immobile, skip to ii; else: find 1st-order reaction channels associated with S_1 and update the reaction rates using Eq. (5).
      ii. Loop through the mobile-cluster hash $\overrightarrow{S}_m$, determine if $key(S_1) \geq key(S_2)$ where S_2 denotes the mobile cluster and find the associated reaction channel $\mathbb{R}(S_1, S_2)$.
         A. If $\mathbb{R}(S_1, S_2)$ exists: update values of $\mathbb{R}(S_1, S_2)$ and the total rate R_{tot} using Eqs. (6) and (7).
         B. Else: calculate the reaction rate $R(S_1, S_2)$ between clusters S_1 and S_2, add $\mathbb{R}(S_1, S_2)$ to the reaction hash $\overrightarrow{\mathbb{R}}$ and update the total reaction rate, $R_{tot} \leftarrow R_{tot} + R(S_1, S_2)$.
   (b) Else if $f_2(S_1) \neq 0$ (the cluster has just been created): similar to 2(a), except that all reactions associated with S_1 will be added directly into the reaction hash $\overrightarrow{\mathbb{R}}$. It is not necessary to locate these reactions in $\overrightarrow{\mathbb{R}}$ since they are completely new reactions.
3. Else $f_1(S_1) = f_2(S_1) = 0$ (the cluster does not participate in the previous reaction):
   (a) Loop through the effected clusters S_2 contained in the dynamic array and evaluate these following conditions: 1) $key(S_1) > key(S_2)$ and S_1 is mobile, 2) S_1 is immobile while S_2 is mobile.
   (b) If any of those conditions is satisfied: find the associated reaction $\mathbb{R}(S_1', S_2')$ (S_1' is the larger value of S_1 and S_2, the other is S_2').
      i. If $\mathbb{R}(S_1', S_2')$ exists: update $R(S_1', S_2')$ and R_{tot} using Eqs. (6) and (7).
      ii. Else: calculate the reaction rate $R(S_1', S_2')$ between clusters S_1 and S_2, add $\mathbb{R}(S_1', S_2')$ into the reaction hash $\overrightarrow{\mathbb{R}}$ and update the total reaction rate, $R_{tot} \leftarrow R_{tot} + R(S_1', S_2')$.
4. Proceed to the next cluster in the all-cluster hash $\overrightarrow{S}_{all}$ and repeat Step 2 until reaching the last cluster.
5. Loop through the clusters contained in the dynamic array. For all possible pairs of (S_1, S_2), if at least one of the clusters in the pair is mobile: find the associated reaction $\mathbb{R}(S_1', S_2')$ (S_1' is the larger value of S_1 and S_2, the other is S_2').
   (a) If $\mathbb{R}(S_1', S_2')$ exists: update $R(S_1', S_2')$ and R_{tot} using Eqs. (6) and (7).
   (b) Else: calculate the reaction rate $R(S_1', S_2')$ between clusters S_1 and S_2, add $\mathbb{R}(S_1', S_2')$ into the reaction hash $\overrightarrow{\mathbb{R}}$ and update the total reaction rate, $R_{tot} \leftarrow R_{tot} + R(S_1', S_2')$.
6. Locate in the all-cluster hash $\overrightarrow{S}_{all}$ the same clusters S_i that are stored in the dynamic array and reset the values of f_1 and f_2: $0 \leftarrow f_1(S_i)$ and $0 \leftarrow f_2(S_i)$ and clear the dynamic array.

If the τ-leaping method is implemented, update the noncritical cluster hash $\overrightarrow{P}$ and noncritical reaction hash $\overrightarrow{\mathbb{J}}$ at the end of Steps 2, 3 and 5 above as described in Algorithm 4.5.1.

4.5.3. Main event loop

1. If the simulation is resumed from a pre-existing one, enter input data into the hash tables $\overrightarrow{S}_{all}$, $\overrightarrow{S}_{m}$, and $\overrightarrow{\mathbb{R}}$. Set the appropriate initial time and compute the total rate R_{tot} of all reactions associated with existing defect clusters in $\overrightarrow{S}_{all}$. Skip to Step 3.
2. Update the $\overrightarrow{S}_{all}$, $\overrightarrow{S}_{m}$ and $\overrightarrow{\mathbb{R}}$ hashes and the total reaction rate R_{tot} as described in Algorithm 4.5.2 Perform volume rescaling if the conditions in Eq. (12) are satisfied.
3. If SSA has run less than N_{SSA} steps: select and execute a reaction event $\mathbb{R}(S_1, S_2) \in \overrightarrow{\mathbb{R}}$ and calculate the time to next reaction event using Eqs. (3) and (4), store identities of the effected clusters in the dynamic array and return to Step 2 until the final time is reached; else: go to Step 4.
4. Reset $N_{SSA} \leftarrow 0$. If the noncritical reaction hash $\overrightarrow{\mathbb{J}}$ is empty: τ-leaping cannot be performed, return to Step 3; else:
   (a) Calculate the value of $g(P_i)$ for each cluster $P_i \in \overrightarrow{P}$ based on the values of its O_i parameters as described in Section 4.3.
   (b) Determine the value of the noncritical time leap τ' using Eq. (8).
   (c) Calculate the total reaction rate R_{cr} of all the critical reactions in $\overrightarrow{\mathbb{R}}_{cr}$ and the critical time leap τ'' using Eq. (3).
   (d) If τ' is less than some small n-multiple (we set n equal 10) of $1/R_{tot}$, temporarily abandon τ-leaping and return to Step 3.
   (e) Else:
      i. Take the leap time to be the smaller value of τ' and τ'', $\tau = min\{\tau', \tau''\}$.
      ii. Calculate the number of times each reaction $\mathbb{J}_i \in \overrightarrow{\mathbb{J}}$ will fire during this time interval $[t, t+\tau)$ as described in Section 4.3.
      iii. If $\mathcal{P}(J\tau) > X(P_1)$ (if $\mathbb{J}_i$ is a 1st-order reaction) or $\mathcal{P}(J_i\tau) > min\{X(P_1), X(P_2)\}$ (if $\mathbb{J}_i$ is a 2nd-order reaction): reduce τ' by half and return to Step 4(d). Else: assign $k(\mathbb{J}_i) \leftarrow \mathcal{P}(J_i\tau)$, $k(P_{1,2}) \leftarrow k(P_{1,2}) + k[\mathbb{J}(P_1, P_2)]$. If $k(P_{1,2}) > X(P_{1,2})$: reduce τ' by half and return to Step 4(d).
      iv. Execute $\mathbb{J}_i \in \overrightarrow{\mathbb{J}}$ a number of $k(\mathbb{J}_i)$ times. Store the identities of the effected clusters in the dynamic array if $k(\mathbb{J}_i) > 0$. Update $t \leftarrow t + \tau$, then return to Step 2 or stop if the final time has been reached.
      v. If $\tau'' \leq \tau'$: select and execute a critical reaction event $\mathbb{R}(S_1, S_2) \in \overrightarrow{\mathbb{R}}_{cr}$ and store identities of the effected clusters in the dynamic array. If an insertion event is selected, process the event and store the identities of the new clusters in the dynamic array, then return to Step 2 or stop the simulation if the final time has been reached.

5. Triple ion-beam irradiation of bcc-Fe thin film

5.1. Simulation

Materials performance in nuclear fusion reactors is expected to degrade as a consequence of prolonged exposure to neutron irradiation. However, neutron irradiation experiments are costly, irradiation facilities are scarce and presently achievable neutron fluxes are low requiring years of exposure before material specimens receive a significant dose of irradiation. As a faster and more cost effective alternative for assessing irradiation-induced changes in physical and mechanical properties of materials, ion beam experiments are used for accelerated testing of material degradation because ion cascades can produce damage similar to neutron irradiation but on a much shorter time scale. In addition to the displacement damage, material exposure to fast neutrons results in simultaneous formation of He and H atoms through nuclear transmutation reactions. To mimic such specific conditions properly, triple ion beam irradiation can be used in which ions of He and H are co-implanted, either sequentially or concurrently, with the heavy ions imparting the primary (displacement) damage. Recent triple-beam experiments of this kind conducted on iron crystals have revealed pronounced synergistic effects associated with co-implantation of He and H under irradiation by self-ions of Fe [28]. Specifically, the amount of measurable swelling increased several fold when all three ion species were implanted simultaneously, relative to baseline sequential dual Fe^{3+}/He^{+} and Fe^{3+}/H^{+} irradiations.

As previously discussed, ODE-based simulations methods have so far proven incapable of coping with complex cluster species with more than two size attributes, as is the case of triple-beam irradiations reported in Ref. [28]. Here we show that our enhanced SCD method is capable of simulating of complex defect microstructures in pure iron subjected to simultaneous irradiation with Fe^{3+} ions and co-implantation with He^{+} and H^{+} ions. In setting up our model and SCD simulations we mimic as close as possible irradiation conditions used in the triple ion-beam experiments performed by Tanaka and coworkers. The model parameters used in SCD simulations reported here are the same as in Ref. [6].

We have performed simulations of triple-beam irradiation and tracked the accumulation of pure vacancy (V), V–He, V–H, and V–He–H clusters in the simulation volume using first direct (exact) SCD simulations [6] and then repeated the same

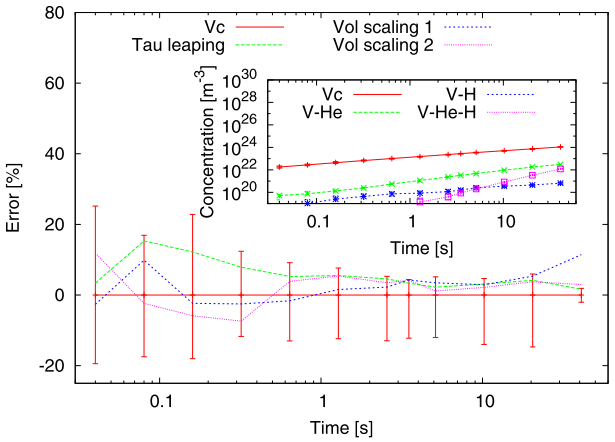

Fig. 1. Statistical errors of vacancy cluster (Vc) concentrations obtained from various enhancement methods compared to the one obtained from original SCD model using SSA method. The specimen is under triple ion irradiation of Fe^{3+}, He^{+} and H^{+}, total irradiation time is 40.96 s and the temperature is 783 K. The inset shows the concentrations of various defect-cluster types as functions of irradiation time, in this case the simulation is carried out using the original SCD algorithm with no improvement. Vol scaling 1 uses $\gamma = 0.9999$, and Vol scaling 2 uses $\gamma = 0.99999$.

simulations after turning on, one by one, the various enhancements described in the preceding sections. The inset to Fig. 1 shows the concentrations of various types of clusters as functions of simulated irradiation time as obtained with the original (unenhanced) SCD method.

Each curve in the inset was obtained by averaging over five independent simulations starting from different random seeds. The main figure shows the relative deviation from the reference (unenhanced) simulations in the net vacancy cluster population obtained in SCD simulations with enhancements. For consistency, five independent simulations were performed for every enhancement. The error bars shown on the plot can be used as a measure of statistical significance of the observed deviations. As the figure shows, the results of enhanced SCD simulations fall within the statistical errors to the exact (reference) simulations which verifies that the approximations used here to improve computational efficiency of SCD simulations, namely τ-leaping and volume rescaling, are not distorting the simulated kinetics of damage accumulation (for simulations shown in Fig. 1 we used the following values of runtime parameters: $n_{cr} = 10$ and $\epsilon = 0.03$ for τ-leaping and γ ranging from 0.99999 to 0.9999 for volume rescaling). The ratio γ can be reduced further to achieve even greater speedup, but accuracy is what we prefer here since we have already managed to reduce the computing time significantly with the current simulations.

However, volume reduction should not be applied when the simulation volume has become too small to be statistically representative. Therefore, a better way to implement volume rescaling is to combine this method with model parallelization using multiple replicas of the original reaction volume. Specifically, whenever the above volume reduction condition is satisfied, the simulation volume will be halved, defect evolution in one of these halves will then continue on another processor while that in the other half will continue on the same processor. As time progresses, further volume reduction on these reaction sub-volumes will be carried out in the same manner until the final simulation conditions are reached. This improvement approach will be addressed in future publication.

5.2. Performance

First of all, a rather significant – a factor of 20 or higher – speedup in SCD simulations is attained simply due to a greater efficiency of the incremental updates of the evolving reaction network and associated reaction rates, as described in Sections 4.1 and 4.2. This is a general improvement resulting from a better implementation of the standard SCD algorithm reported in Ref. [6]. We use the efficiency of our standard SCD simulations with incremental updates as a reference comparison with further enhancements.

We find that, in our SCD simulations of irradiated iron, conditions for τ-leaping are often satisfied and many reactions can be allowed to fire at once rather than one at a time. The key condition for τ-leaping to be accurate is that the change in the defect population caused by a leaping step should not affect too much the rates of existing reactions. Whenever it is safe to perform, τ-leaping results in longer time-steps compared to the standard (one reaction at a time) SCD algorithm, as shown in Fig. 2(a) for the same simulation setup as described in the previous section. It is clear that, with τ-leaping active, fewer short time-steps are taken than in the direct SCD resulting in a total reduction in the number of time-steps required to simulate the same evolution. Thus, the total number of steps taken is reduced significantly, and so is the overhead cost for system updating. This is confirmed in Fig. 2(b), where a histogram of the time-step size distribution for a τ-leaping simulation is plotted and compared to the same histogram obtained from a standard SCD simulation. In addition to showing that the distribution shifts to longer time-steps due to τ-leaping, the same histogram also shows that a few specific time-steps occur much more frequently than the rest, and that they are clearly separated from each another.

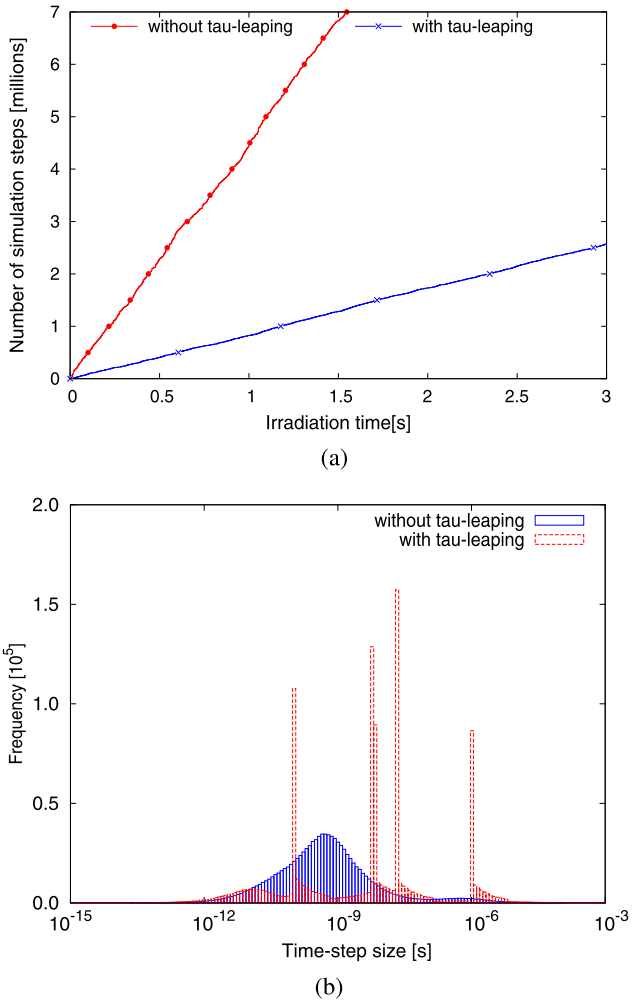

Fig. 2. (a) Number of simulation steps as functions of the irradiation time obtained from SCD simulations of triple-beam irradiation at 783 K with and without τ-leaping implementation. (b) Distribution of the time-steps in these two cases, here one millions time-steps are collected and analyzed.

The observed peaks in the distribution indicate that a handful of noncritical reaction channels dominate the kinetics in our model, and that our enhanced algorithm can identify and handle such reaction channels efficiently with τ-leaping. Specifically, the first peak in Fig. 2(b), which ranges from 89.2 ps to 123 ps is dominated by the absorption of SIA and SIA clusters by defect sinks. The second group, whose reactions with time-steps between 5.01 ns and 25.1 ns mostly consists of absorption of vacancies by sinks, and reactions in the last group from 0.87 μs to 1.19 μs are predominantly migration of vacancy clusters to defect sinks. As a result, τ-leaping is not only better for computational efficiency, but it also provides very useful physical information by identifying the reactions that control the kinetic evolution of the system. This has implications beyond efficiency improvements because it can indicate where to focus the efforts to calculate the physical parameters that matter the most with maximum accuracy. This can potentially be helpful in uncertainty quantification of the models and/or to learn where to devote efforts to improve the physical parameterization.

To quantify the speedup gained from the enhancements described in this paper, the computational cost of SCD simulations performed with and without the enhancements is plotted in Fig. 3 as a function of the simulated time. As the figure shows, significant gains in simulation efficiency are realized using τ-leaping and volume rescaling. τ-leaping is typically most efficient at early stages of SCD simulations but its associated speedup is subsequently negated by an increasing computational cost of updates of the growing reaction network. Under such circumstances volume rescaling is prescribed to control the size of a growing defect population. When used together, these two enhancements significantly reduce the wall clock time of a SCD simulation without detectable sacrifice in its accuracy. As an example, Fig. 3 shows that while it took the original SCD algorithm more than two days to achieve a trivial irradiation dose of 0.1 dpa, it only takes the enhanced algorithm only about 12 hours to reach a technologically significant dose of 50 dpa.

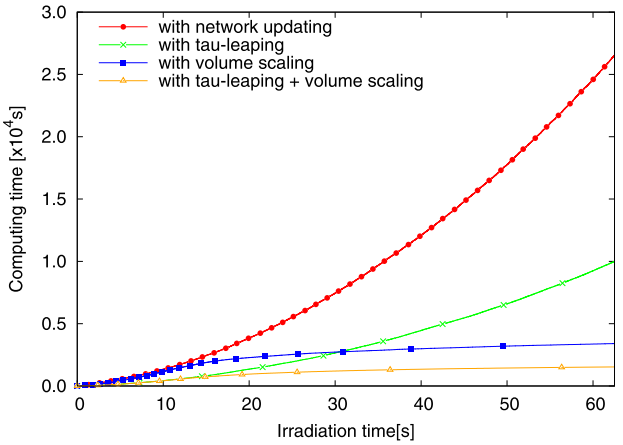

Fig. 3. Comparison of computational cost of the SCD model with different enhancement methods, here a scaling ratio, $\gamma = 0.99999$ is used whenever the volume scaling method is implemented.

6. Conclusions

We have presented a computationally efficient implementation of the SCD algorithm originally devised as an adaptation of the well-known SSA method to simulations of complex microstructure evolution in irradiated materials. The key advantage of the SCD method is that, unlike the traditional ODE-based rate theory approaches that notoriously suffer from combinatorial explosion, SCD handles with ease multi-species populations of arbitrary complexity. However, early applications of the original SCD algorithm to irradiated materials exposed several computational bottlenecks, e.g. wide disparity in reaction rates and stiff kinetics. Enhancements presented in this paper are introduced to address some of the bottlenecks in order to achieve reactor-relevant irradiation doses at a reasonable computational cost. Gains in computational efficiency of SCD simulations are achieved through the following: incremental updates of the evolving reaction network, τ-leaping permitting multiple reaction events to take place over a single simulation step, and volume rescaling to control the size of defect population. Further enhancements to the SCD algorithm reported here are being considered, e.g. a more robust method for SCD simulations of stiff reaction networks with wide spectra of reaction rates and an adaptive mechanism for deciding which method is best to use at each particular stage of an SCD simulation to optimize the overall computational performance. Efficient parallelization of the SCD algorithm is another interesting venue for further research, e.g. following replication strategies recently proposed in the context of parallel kinetic Monte Carlo algorithms [29]. We note that, in addition to our SCD development borrowing heavily from the SSA method ideas, algorithmic enhancements reported here can be re-used in other simulation contexts where reaction-diffusion processes with dynamic species populations are of interest, such as in combustion science, cellular process simulation, or chemical kinetics.

Acknowledgements

We would like to thank Daryl Chrzan, Athanasios Arsenlis, Alexander Stukowski, David Cereceda, and Ninh Le for many helpful discussions. TLH acknowledges support from the Lawrence Scholar Program, the UC Berkeley Chancellor's Fellowship, and the Nuclear Regulatory Commission Research Fellowship. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

References

[1] R. Bullough, B.L. Eyre, K. Krishan, Cascade damage effects on the swelling of irradiated materials, Proc. R. Soc. Lond. A 346 (1975) 81–102.
[2] N.M. Ghoniem, M.L. Takata, A rate theory of swelling induced by helium and displacement damage in fusion reactor structural materials, J. Nucl. Mater. 105 (1982) 276292.
[3] L.K. Mansur, The reaction rate theory of radiation effects, JOM 48 (1996) 28–32.
[4] M. Koiwa, On the validity of the grouping method – comments on "Analysis of the clustering process of supersaturated lattice vacancies", J. Phys. Soc. Jpn. 37 (1974) 1532.
[5] A.E. Sand, S.L. Dudarev, K. Nordlund, High-energy collision cascades in tungsten: dislocation loops structure and clustering scaling laws, Europhys. Lett. 103 (2013) 46003.
[6] J. Marian, V.V. Bulatov, Stochastic cluster dynamics method for simulations of multispecies irradiation damage accumulation, J. Nucl. Mater. 415 (2012) 84–95.
[7] D.T. Gillespie, A general method for numerically simulating the stochastic time evolution of coupled chemical reactions, J. Comput. Phys. 22 (1976) 403–434.
[8] U.M. Ascher, L.R. Petzold, Computer Methods for Ordinary Differential Equations and Differential Algebraic Equations, Soc. Ind. Appl. Math., Philadelphia, 1998.
[9] D.T. Gillespie, A rigorous derivation of the chemical master equation, Physica A 188 (1992) 404–425.

[10] Y. Cao, H. Li, L.R. Petzold, Efficient formulation of the stochastic simulation algorithm for chemically reacting systems, J. Chem. Phys. 121 (2004) 4059–4067.
[11] X. Cai, Exact stochastic simulation of coupled chemical reactions with delays, J. Chem. Phys. 126 (2007) 124108.
[12] Tae-Hyuk Ahn, Yang Cao, Layne T. Watson, Stochastic simulation algorithms for chemical reactions, in: BIOCOMP-2008, pp. 431–436.
[13] Cain: stochastic simulations for chemical kinetics (http://cain.sourceforge.net), StochKit: a stochastic simulation toolbox for biology (http://www.cs.ucsb.edu/~cse/index2.php?software.html).
[14] M. Gherardi, T. Jourdan, S. Le Bourdiec, G. Becteux, Comput. Phys. Commun. 183 (2012) 1966–1973.
[15] D.T. Gillespie, Approximate accelerated stochastic simulation of chemically reacting systems, J. Chem. Phys. 115 (2001) 1716–1733.
[16] M.A. Gibson, J. Bruck, Efficient exact stochastic simulation of chemical systems with many species and many channels, J. Phys. Chem. A 104 (2000) 1876–1889.
[17] J.M. McCollum, G.D. Peterson, C.D. Cox, M.L. Simpson, N.F. Samatova, The sorting direct method for stochastic simulation of biochemical systems with varying reaction execution behavior, Comput. Biol. Chem. 30 (2006) 39–49.
[18] H. Li, L.R. Petzold, Logarithmic direct method for discrete stochastic simulation of chemically reacting systems, Technical report, 2006.
[19] D.T. Gillespie, L.R. Petzold, Improved leap-size selection for accelerated stochastic simulation, J. Chem. Phys. 119 (2003) 8229–8234.
[20] A. Chatterjee, D. Vlachos, M. Katsoulakis, Binomial distribution based τ-leap accelerated stochastic simulation, J. Chem. Phys. 122 (2005) 024112.
[21] Y. Cao, D.T. Gillespie, L.R. Petzold, Efficient stepsize selection for the tau-leaping simulation method, J. Chem. Phys. 124 (2006) 044109.
[22] T. Tian, K. Burragex, Binomial leap methods for simulating stochastic chemical kinetics, J. Chem. Phys. 121 (2006) 10356–10364.
[23] Y. Cao, L.R. Petzold, Trapezoidal tau-leaping formula for the stochastic simulation of chemically reacting systems, in: Proc. Found. Syst. Biol. Eng. (FOSBE), 2005, pp. 149–152.
[24] E.L. Haseltine, J.B. Rawlings, Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics, J. Chem. Phys. 117 (2002) 6959–6969.
[25] M. Rathinam, L.R. Petzold, Y. Cao, D.T. Gillespie, Stiffness in stochastic chemically reacting systems: the implicit tau-leaping method, J. Chem. Phys. 119 (2003) 12784–12794.
[26] A. Samant, D.G. Vlachos, Overcoming stiffness in stochastic simulation stemming from partial equilibrium: a multiscale Monte Carlo algorithm, J. Chem. Phys. 123 (2005) 144114.
[27] Y. Cao, D.T. Gillespie, L.R. Petzold, Avoiding negative populations in explicit Poisson tau-leaping, J. Chem. Phys. 123 (2005) 054104.
[28] T. Tanaka, K. Oka, S. Ohnuki, S. Yamashita, T. Suda, S. Watanabe, E. Wakai, Synergistic effect of helium and hydrogen for defect evolution under multi-ion irradiation of Fe–Cr ferritic alloys, J. Nucl. Mater. 329–333 (2004) 294–298.
[29] E. Martinez, P.R. Monasterio, J. Marian, Billion-atom synchronous parallel kinetic Monte Carlo simulations of critical 3D Ising systems, J. Comput. Phys. 230 (2011) 1359–1369.