I&EC research
Industrial & Engineering Chemistry Research

Article

pubs.acs.org/IECR

Multiscale, Multidomain Modeling and Parallel Computation: Application to Crystal Shape Evolution in Crystallization

Joseph Sang-Il Kwon,[†] Michael Nayhouse,[†] and Panagiotis D. Christofides*[,†,‡]

[†]Department of Chemical and Biomolecular Engineering and [‡]Department of Electrical Engineering, University of California, Los Angeles, California 90095, United States

ABSTRACT: We focus on the development of a parallelized multiscale, multidomain modeling scheme that directly reduces computation time requirements without compromising the accuracy of established chemical models. Specifically, creating a parallel computation program from a sequential one consists of three steps: (1) decomposition of the original problem into tasks or domains, (2) assignment of tasks to processors, and (3) orchestration (using a programming mechanism) to communicate and synchronize the information flow among all processors involved. In this work, we applied a parallelized multiscale modeling strategy according to the above three steps to a multiscale model of a batch crystallization process. First, we decomposed the nucleation and crystal growth processes in a batch crystallization system into the collection of tasks where each task represents the crystal growth of a nucleated crystal. Second, the tasks (i.e., simulating the crystal growth of nucleated crystals) are assigned according to a modulus function where the number of crystal modules is equal to the number of processors available. Third, the message-passing interface (MPI) settings that use the information passing between the processors is used to link the macroscopic model (e.g., mass and energy balance equations describing continuous-phase variables) to the microscopic models (e.g., kinetic Monte Carlo model describing crystal nucleation, growth, and shape evolution). The parallelized multiscale simulation utilizes a manager−worker MPI computational scheme: There is a processor (i.e., manager) that is responsible for decomposing an original sequential problem (e.g., multiscale model to simulate batch crystallization system) into a number of tasks (e.g., crystal growth of a group of crystals) and allocating those tasks to processors (i.e., workers). Workers are responsible for solving assigned tasks, and when a worker completes the simulation of an assigned task, it notifies the manager. Then, the manager allocates a new task to the processor. Before initiating their following task executions, all processors wait until they have received all the data computed by the other processors at the previous task execution step, allowing for a synchronization of the tasks over the processors used. Therefore, we are able to achieve a significant decrease in time required to complete the batch simulation as the number of cores is increased. A series of results demonstrating the computational efficiency of the approach using the batch crystallization process multiscale model are presented. Specifically, a linear speedup behavior is observed up to the use of 32 cores, and the maximum speedup of about 39-fold is achieved when we use 64 cores.

INTRODUCTION

The modeling of multiscale systems has made fundamental understanding and quantitative prediction possible for processes with complex behavior and product characteristics, and it has tremendous potential to significantly contribute to the chemical, pharmaceutical, and microelectronic industries.[1−4] Motivated by the advances in high-performance computing power, an increasing interest in multiscale, multidomain modeling has been triggered. Moreover, chemical engineers, among other scientists and engineers, have the potential to impact the field of multiscale process modeling because of our unique discipline ranging from molecular modeling to large-scale chemical process modeling.

More specifically, kinetic Monte Carlo (kMC) modeling has received growing attention for dynamic simulations of microscopic/mesoscopic process behavior. The basic principle of kMC is that in order to efficiently simulate a dynamical system with a variety of different rates of processes (e.g., adsorption, migration, and desorption of molecules on a surface in crystal growth) at each step in the simulation, the next process is determined on the basis of a probability proportional to the rate for that process, and after an event is executed, the rates for all processes are updated. The time of the next event is determined by the overall rate for the microscopic surface processes and a suitably defined random number. The standard kMC algorithm is a serial algorithm in the sense that one event can occur at each time step. For many problems of practical interest, however, one needs to simulate systems with larger temporal and spatial scales than the ones that can be simulated using a serial algorithm and available computing power. For these problems, motivated by the recent efforts to develop parallel computation frameworks for the simulation of multisacle process models,[5−14] it would be desirable to develop efficient parallel kMC algorithms so that many processors can be used simultaneously in order to accomplish realistic computations over extended temporal and spatial scales.

One of the most frequent uses of parallel architectures is simply performing independent simulations of a model under different conditions on different processors. Additionally, for very large problems, parallel architectures can be used to improve the speed of the simulation dramatically by decomposing the system into different components/domains

Received: June 29, 2015
Revised: November 3, 2015
Accepted: November 5, 2015
Published: November 5, 2015

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

and assigning each component/domain to a different processor. When we design a parallel computation algorithm, it is important to compare the relative order of magnitude difference between communication time and computation time because we can use it to compute the theoretical maximum speedup via Amdahl's law. For example, if the size of the individual tasks to be executed is too small, then the time used to communicate information between processors may not be small compared to the time needed by each processor for computation. In such a case, the performance may actually worsen as processors are added. In contrast, for microscopic systems with long-range molecular interactions, most of the computational effort goes into the calculation of energy changes; thus, the communication overhead is much less of a problem and is minimal relative to the computation time.

Recently, there has been a great deal of work on the development of rigorous asynchronous parallel algorithms for equilibrium Monte Carlo simulation (eMC).[15−18] However, there has been surprisingly little work completed on parallel algorithms for kMC simulation. This is because the interval between successive events in kMC algorithms depends on the surface microconfiguration (e.g., in the evolution of surface microconfiguration in this film growth and crystal growth); thus, it requires additional bookkeeping to keep track of the rates (probabilities for each event to be selected). In particular, for systems such as crystallization and thin film deposition, surface processes play a key role; therefore, the possible rates or probabilities for events can vary by several orders of magnitude. Several contributions have been made to the development of multiscale models used to simulate the deposition of thin films for a variety of applications,[19−23] whereas the development of efficient parallel algorithms for kMC simulations remains a challenging problem.

Motivated by this, in a previous work,[24] we explored a hybrid kMC algorithm originally developed for the growth of silicon films as in ref 25. More specifically, because of the high frequency of surface migration events relative to that of other surface processes, a brute force kMC algorithm would spend more than 99% of computation time on migration alone. The brute force kMC algorithm refers to the kMC simulation that uses a single core with no changes made to the surface processes of the program (e.g., adsorption, migration, and desorption) in the context of improving the computational efficiency. The simulation of surface migration process is decoupled from the standard kMC implementation and separately executed using a 1D lattice random-walk process.[24] As a result, a significant computation time savings was achieved by a very small compromise of the accuracy of the results, which is due to the fact that in the decoupling strategy all migration events are executed at the same time via 1D random walk whereas in the brute force kMC simulation every migration event is executed one by one keeping track of the effect of each migration event to the surface microconfiguration and thereby to other surface processes (e.g., adsorption and desorption).

In this work, we have attempted to directly deal with the problem of reducing computational requirements without compromising the accuracy of established chemical models via a parallelized kMC. We show that choosing an appropriate decomposition strategy is the key to reducing communication among processors, and it is ideally suited for parallel implementation without compromising precision. Specifically, the message-passing interface (MPI) settings that use the information passing between the cores are selected and are used following a "manager−worker" scheme: There is a processor (i.e., manager) that is responsible for partitioning a problem (e.g., kMC model to simulate batch crystallization system) into partitions (e.g., crystal growth of a group of crystals) and allocating the partitions to processors (i.e., workers). Workers are responsible for solving assigned partitions, and when a worker completes the simulation of a partition, it notifies the manager. Then, the manager allocates the worker a new task.

Although the decoupling strategy introduced by ref 26 reduces the computation requirement by compromising the accuracy of the simulation result, the proposed parallel computation algorithm reduces the computation requirement by directly assigning crystals to multiple cores one by one; thus, the accuracy of the simulation result remains identical. Furthermore, compared to the previous studies of the multiscale and parallel computation,[27−29] the novelty of the proposed multiscale parallel computation scheme lies in its applicability to the crystallization process by directly allocating crystals over multiple cores evenly. As aggregation events proceed, the number of crystals assigned to each core varies. Using the proposed parallel computation scheme, we can effectively deal with this issue. Furthermore, when we deal with a heterogeneous cluster (i.e., the processor speed is not identical over multiple cores), it is possible to assign less crystals to the core with the slower processor speed. Instead, for example, the fastest core will receive more crystals than the other cores and as a result, a better speedup can be achieved overall. A finite volume method for the parallel simulation of a population balance model is considered in ref 30. In the present work, a parallel computation of a multiscale framework is considered and compared to the population balance model; it can provide more fundamental understanding of the crystallization system such as the surface microstructure dependence of the crystal growth process and the codependence between the microscopic surface process and the macroscopic continuous phase in the crystallization process.

The manuscript is structured as follows: We initially describe a parallel computational framework suitable for multiscale models. Then, we discuss the multiscale model of our case study, a batch crystallization process used to produce tetragonal hen-egg-white (HEW) lysozyme crystals. The proposed parallel computation scheme is applied to the multiscale model, and a series of results that demonstrate the computational efficiency and accuracy of the approach are presented.

PARALLELIZED COMPUTATIONS

Motivation. There are three reasons why one might want to use parallelized computation. First, one may want to speed up simulations by using multiple processors. More specifically, parallelization can reduce the simulation time required for the simulation of a large system that can be done on a single processor. Second, one might want to do many simulations at different conditions (e.g., in order to find suitable model parameters by testing parameters over a large range of different parameter values). We can also reduce the noise in a stochastic method such as kMC simulations by running a simulation multiple times. The process of creating a parallel program from a serial one consists of three steps: (1) decomposition of the original serial computation problem into small tasks, (2) assignment of tasks to processors, and (3) orchestration of the communication among processors and synchronization at each

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

time step.[31] Below we discuss these three tasks as they pertain to parallelized simulation of multiscale models.

Decomposition. Decomposition concerns how to break up or divide a computation problem into a collection of tasks that may be executed concurrently. This does not simply imply a straightforward division of a computation problem into a number of tasks equal to the number of available computers. In some cases, the number of tasks can vary dynamically as the program executes, which is known as an irregular decomposition problem.[31] The main objective in decomposition is to expose enough concurrency to keep all processors busy at all times, yet not decompose so much that the overhead of managing the task's decomposition through communication between processors becomes substantial compared to the computation time. In parallel computations, the theoretical maximum speedup using multiple processors can be computed via Amdahl's Law.[31] More specifically, if P is the fraction of an original serial program that can be parallelized and $1-P$ is the fraction of a program that cannot be parallelized (thus, remain serial), then the maximum speedup that can be achieved via parallelization using N processors can be computed as follows:

$$S(N) = \frac{1}{(1-P) + \frac{P}{N}} \tag{1}$$

where S is the maximum speed up. If some portions of a program's execution do not have as much concurrency as the number of processors used, then some processors will have to be idle for those portions, and speedup will be suboptimal.

Initially, the parallel computations may be slower than the serial ones because of communication and synchronization overheads that are not incurred by the sequential program. After the parallel program overcomes this overhead, it provides improved performance as the number of processors increases. Eventually, there is a tail-off region where performance does not substantially increase as the number of processors increases. This region occurs because the number of available tasks obtained after the decomposition step can be bounded. Eventually, adding computers does not improve computational performance because there is not sufficient work to keep all processors busy. Therefore, decomposition should provide a number of tasks considerably greater than the number of processors available.

There are many decomposition techniques. For example, domain decomposition is used to divide up the data of a program and operate on the parts concurrently (e.g., matrix calculations and inner product calculations), whereas functional decomposition refers to dividing up the function (e.g., computing the integral of a function $f(x)$ on the closed interval $[a, b]$). Furthermore, irregular problem decomposition refers to decomposition in the case where the program structure evolves dynamically as the program executes and cannot be determined a priori (e.g., flow dynamics and particle flow simulations). The size of the tasks in the irregular decomposition problem may vary widely; thus, a method of load balancing must be employed to keep computers busy. More detailed discussion on the load balancing scheme will be covered in the following section in the context of parallelized computation of a multiscale model of a batch crystallization process.

Assignment. Assignment refers to the mechanism by which tasks will be distributed among processors. The primary goal of the assignment is to balance the workload among processors to reduce communication between processors and the overhead of managing the assignment. Specifically, the workload to be balanced includes computation (i.e., task execution), input/output data access, and communication between processors. The simplest assignment strategy is to divide the total task number by the number of cores available; thus, consecutive partitions are packed into the same processor (i.e., packed allocation). The other widely used strategy is to use a modulus function such that the group number modulus is equal to the number of processors available (i.e., round-robin allocation).

In particular, there are several load-balancing strategies to deal with irregular problems where the size of each task changes dynamically. Specifically, bin packing is a technique used in cases where the time required to run a task is proportional to the length of the task where the task size grows with time. The goal is to keep the computation load at each processor balanced. The manager−worker scheme is a centralized scheme that involves a manager processor and a collection of worker processors. The manager processor is responsible for assigning tasks decomposed from an original problem to worker processors. The worker processors are responsible for processing tasks and are generally independent processors. When a worker processor completes the assigned task, it notifies the manager, and the manager allocates a new task.

Specifically, there are two kinds of manager−worker schemes: synchronous and asynchronous. For the synchronous scheme, before initiating their subsequent task executions, all processors wait until they have received all of the data computed by other processors at the previous task execution step. In contrast, for the asynchronous scheme, all of the processors perform their computations without waiting for the data computed by other processors (i.e., they do not account for the progression of the other processors). In general, the synchronous manager−worker scheme is suitable for small homogeneous clusters with fast communication, whereas the asynchronous manager−worker scheme provides better performance on large-size heterogeneous clusters. Also, when the processors have significantly different performance from one another, the speed of a synchronous manager−worker scheme is limited by the slowest processors. In this case, the asynchronous scheme achieves better performance than the synchronous one.

Orchestration. To execute their assigned tasks, processors need mechanisms to name and access data and to communicate and synchronize with other processors. Orchestration uses available mechanisms to accomplish these goals correctly and efficiently. The major goal in orchestration is to reduce the cost of the communication and synchronization (i.e., the overheads of parallelism management) by preserving locality of data and scheduling tasks so that those on which many other tasks depend on will be located at a position which is easily accessible by many other processors.

The MPI is one of the most widely used techniques to deal with communication and synchronization between processors in order to solve chemical engineering problems on parallel processors. For example, suppose that a reaction occurs in a batch process and that because of imperfect mixing the spatial concentration distribution is not uniform throughout the batch system. Next, we can consider that each core runs a kMC code for a time step to compute the amount of reactant consumed by a reaction in a particular spatial domain of the continuous phase. Then, each core sends these values to the manager core via an MPI data manager, and the manager core adds up these values to compute the total amount of solute depletion over the

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

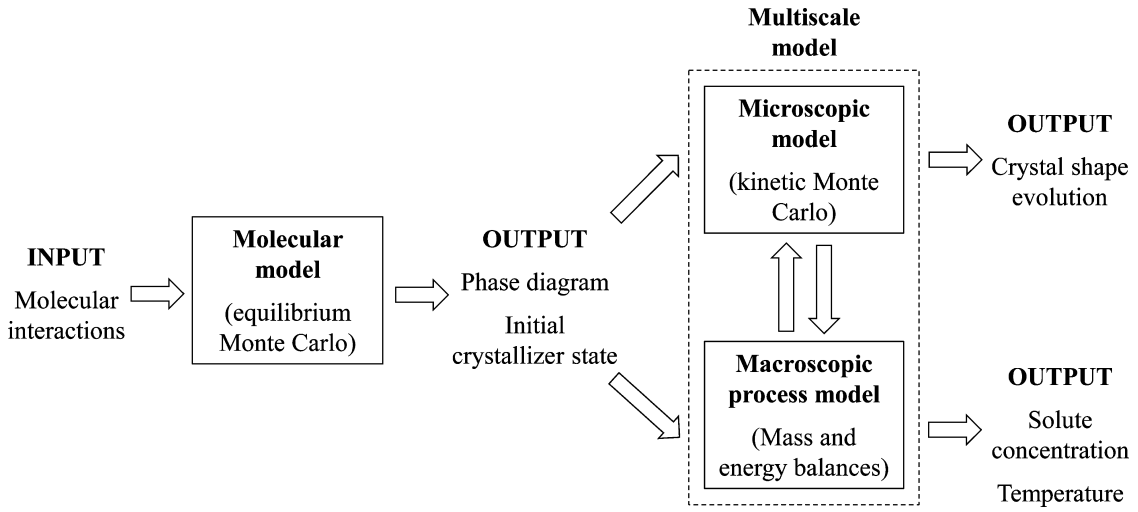

Figure 1. Schematic representation of multiscale modeling for batch crystallization process.

entire spatial domain in the continuous phase. Lastly, the manager core sends back updated values to each core (i.e., each spatial domain). This sequence is repeated until the specified length of time has been completed. The output files store all the simulation results that belong to the individual kMC.

MULTISCALE BATCH CRYSTALLIZATION PROCESS MODEL AND PARALLELIZATION

In a crystallization process, there is a large disparity of time and length scales of phenomena occurring in continuous phase and crystal surface. Therefore, the modeling of the crystallization process from a molecular level to a process level using the assumption of continuum is not practical because it requires nonviable computational power and time.

Motivated by this, we present an integrated multiscale modeling and parallel computation framework for crystallization processes that elucidates the relationship between molecular-level processes such as crystal nucleation, growth, and aggregation and macroscopically observable process behavior and allows computation of optimal design and operation conditions. The multiscale framework encompasses (a) eMC modeling for computing solid−liquid phase diagrams and determining initial crystallization conditions that favor crystal nucleation, (b) kMC modeling for simulating crystal growth and aggregation and predicting the evolution of crystal shape distribution, and (c) integrated multiscale computation simultaneously linking molecular-level models (e.g., kMC simulation) and continuous-phase macroscopic equations (e.g., mass and energy balance equations) covering entire batch crystallization systems.

Specifically, it is schematically illustrated in Figure 1 how the information is exchanged between models used to describe molecular, microscopic, and macroscopic levels. In the beginning, the molecular interactions such as a Lennard-Jones type potential are used as input to the equilibrium Monte Carlo simulation in order to construct a phase diagram through which we can predict the conditions under which a nucleation process is favored and to determine suitable nucleation and crystal growth conditions for initialization of the batch crystallization. More specifically, this initial condition is used to begin the kMC simulation that describes the evolution of crystal shape distribution in a batch crystallizer. Similarly, using the same initial condition, the mass and energy balance equations are used to compute the evolution of the protein solute concentration and the temperature in the crystallizer. By exchanging the information (e.g., concentration, temperature, crystal size and shape distributions, etc.) between the microscopic and macroscopic models, we are able to construct the multiscale process model that can be used to describe the multiscale dynamic behavior of a batch crystallization process.

The multiscale framework introduced in this work provides a fundamental understanding of the crystallization system such as the surface microstructure dependence of the crystal growth process, which can be completed only through the stochastic simulation methods, and the codependence between the microscopic surface process and the macroscopic continuous phase in the crystallization process. Furthermore, the stochastic nature of the aggregation process is also well taken into account by considering all of the pairwise combination of crystals through the probability-based approach introduced by ref 32.

Molecular Model. The eMC method has been applied to the modeling of crystallization system at molecular level to calculate suitable phase diagrams.[33,34] Here we briefly review this approach to demonstrate how the molecular level results are used in the multiscale model of the crystallization process. There are many types of Monte Carlo moves such as particle displacement, volume changes, and particle switching. Additionally, periodic boundary conditions are used to approximate the infinite dimensional system with a model with a finite number of lattice sites. Then, the probability that the system transits from a current state m to another state n, Prob $(m \rightarrow n)$, is computed following the standard Metropolis algorithm as follows:

$$\text{Prob}\,(m \rightarrow n) = \min\left\{1, \exp\left(-\frac{E_n - E_m}{k_B T}\right)\right\} \tag{2}$$

where E_n and E_m are the energies of the system in states n and m, respectively, k_B is the Boltzmann constant, and T is the temperature in Kelvin. To compute the energy of the system, we develop a model to compute the interactions between the particles. A Lennard-Jones type potential is used as follows:

$$U = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^a - \left(\frac{\sigma}{r}\right)^b\right] \tag{3}$$

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

where U is the potential energy between two particles, σ is the radius of particles, r is the distance between the particles, ε is the depth of the potential well, and a and b are model parameters to be determined. Please note that E_n and E_m in eq 2 are the sum of U in eq 3 over all possible pairwise combinations of particles. Then, we measure some properties (e.g., solubility) of the material through experiments, where the measurement is available for a small region, and use them to determine the model parameters such as a, b, and ε in eq 3 that provide a good agreement between simulated and experimental data. We use eq 3 with the known parameters to calculate the phase diagram to predict the conditions under which a nucleation process is favored and to determine suitable initial nucleation and growth conditions to initialize batch crystallization. (Figure 2 shows a typical phase diagram like the one that can be found in ref 35.)

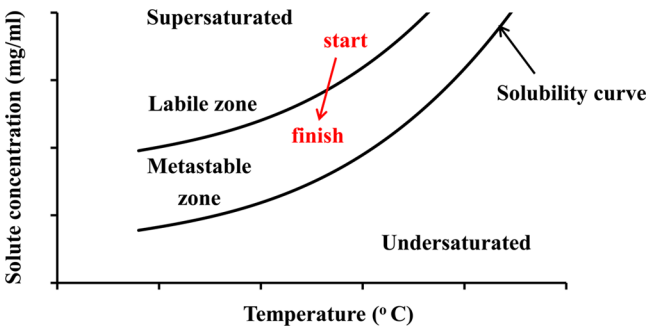

Figure 2. Typical phase diagram. Arrow shows where the initial condition of the batch crystallization system should be to favor nucleation early and how to manipulate the temperature to move into the metastable region to favor growth later in the batch.

In this section, we present a framework that is used to obtain the solid−liquid phase diagram qualitatively. Theoretically, in order to obtain a quantitative phase diagram, we need a molecular simulation approach that takes into account the molecular-level interaction phenomena, which is described in detail in ref 36. Specifically, the paper focuses on the determination of the phase diagram of a system of particles that interact through a pair potential in the form of a Lennard-Jones type potential. Then, the parameters of a Lennard-Jones type potential can be adjusted in order to have good agreement with the phase diagram that is available from the literature.

Microscopic Model. The solid-on-solid model, which is one of the most widely used techniques to simulate crystal growth accounting for crystal surface microstructure, is employed in this work to model the growth of lysozyme crystals as a specific example. In this work, the phenomena of appearing and disappearing faces in crystal growth processes are not considered. Each event of our kMC simulation is chosen randomly on the basis of the rates of the three-surface microscopic phenomena. Please note that the following description of the surface kinetics for the present model follows closely that of ref 37, which was further extended by ref 38 to account for migration events.

The adsorption rate is independent of crystal surface microconfiguration and is defined as

$$r_a = K_0^+ \exp\left(\frac{\Delta\mu}{k_B T}\right) \tag{4}$$

where K_0^+ is the attachment coefficient, k_B is the Boltzmann constant, T is the temperature in Kelvin, and $\Delta\mu = k_B T \ln(C/S)$, where C is the protein solute concentration, S is its solubility, and $\Delta\mu$ is the crystal growth driving force. It is noted that $r_a \propto C$.

The desorption rate of a surface particle depends on its local environment. Thus, the desorption rate of a lattice site with i nearest neighbors is given by

$$r_d(i) = K_0^- \exp\left(-i\frac{E_{pb}}{k_B T}\right) = K_0^+ \exp\left(\frac{\phi}{k_B T} - i\frac{E_{pb}}{k_B T}\right) \tag{5}$$

where K_0^- is the desorption coefficient, i is the number of bonds, ϕ is the binding energy per molecule of a fully occupied lattice, and E_{pb} is the average binding energy per bond. The second equality in eq 5 holds true because of the relationship between K_0^- and K_0^+, which can be found in ref 39. Specifically, surface particles with fewer nearest neighbors have a higher desorption rate. The migration rate is defined similar to the desorption rate in ref 38 as shown below.

$$r_m(i) = K_0^+ \exp\left(\frac{\phi}{k_B T} - i\frac{E_{pb}}{k_B T} + \frac{E_{pb}}{2k_B T}\right) \tag{6}$$

The migration rate is the desorption rate multiplied by the extra exponential term, $E_{pb}/(2k_B T)$, to account for the fact that the migration rate is higher than the desorption rate.

In this work, simulations with hundreds of different E_{pb} and ϕ values were carried out in parallel until satisfactory agreement was achieved among the growth rate computed by the kMC simulation and the experimental growth rates obtained from the literature.[37] From this, we determined a set of model parameters as follows: E_{pb}/k_B = 1077.26 K and ϕ/k_B = 227.10 K for the (110) face, E_{pb}/k_B = 800.66 K and ϕ/k_B = 241.65 K for the (101) face, and K_0^+ = 0.211 s^{-1}.

Macroscopic Model. The following mass and energy balance equations are employed to compute the dynamic evolution of the protein solute concentration and temperature in the batch crystallization process with time:

$$\frac{dC}{dt} = -\frac{\rho_c}{V_{batch}}\frac{dV_{crystal}}{dt}, \quad C(0) = 48\text{mg/mL} \tag{7}$$

$$\frac{dT}{dt} = -\frac{\rho_c \Delta H_c}{\rho C_p V_{batch}}\frac{dV_{crystal}}{dt} - \frac{U_c A_c}{\rho C_p V_{batch}}(T - T_j), \quad T(0) = 15°\text{C} \tag{8}$$

where $V_{crystal}$ is the total volume of crystals growing in the crystallizer (refer to ref 40 for more details on the computation of $dV_{crystal}$), and T_j is the jacket temperature (i.e., manipulated input). The process parameter values are shown in Table 1.

Table 1. Parameters for the Batch Crystallization Process Model

ρ_c	crystal density	1400	mg/cm^3
ΔH_c	enthalpy of crystallization	−4.5	kJ/kg
$\rho(t)$	density of the continuous phase	1000 + $C(t)$	mg/cm^3
C_p	specific heat capacity	4.13	kJ/(K·kg)
V_{batch}	volume of the crystallizer	1	L
A_c	contact area of the crystallizer wall and jacket	0.25	m^2
U_c	overall heat transfer coefficient	1800	kJ/(m^2·h·K)

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

In the scale-up of a crystallization process, agitation is required in order to maintain the crystal phase in suspension. The resulting shear force induces aggregation processes that have a significant impact on the quality of crystal products because they decrease the total number of crystals and increase the average particle size. Motivated by these considerations, aggregation is taken into consideration in the modeling of large-scale crystallization processes.

More specifically, we assume that the continuous phase is sufficiently dilute that only binary aggregation between two particles is possible. Furthermore, according to the Kolmogorov length analysis, shear forces are the major contribution to the aggregation of crystals considered in this work.[41] The corresponding kernel (cf. eq 9) can be used to calculate the number of aggregation events taking place during the sampling time Δ in terms of the aggregation kernel $\beta(V_i, V_j)$, the batch crystallizer volume V_{batch}, the collision efficiency $\gamma(V_i, V_j)$, and the concentrations of particles of volume V_i and V_j as follows:

$$N_{ij} = \gamma(V_i, V_j)\beta(V_i, V_j)m_i m_j V\Delta \quad 1 \le i, j \le C_{total} \tag{9}$$

where m_i is the number concentration (i.e., the number of particles of volume V_i per unit volume). The number C_{total} indicates the number of classes, and Δ = 0.5 s (i.e., the successful collision probability is computed every 0.5 s). Please note that the crystal size domain is linearly broken down into different size classes with identical intervals. The sampling time is chosen by trial and error until we obtain a negligible improvement when a smaller sampling time is used in predicting the number of aggregation events to occur. The rate of formation of aggregates of volume V_k from the collision of particles of volume V_i and V_j is $\frac{1}{2}\sum_{V_i+V_j=V_k} N_{ij}$ where the summation considers all the different combinations of aggregation which result in V_k as follows:

$$V_i + V_j = V_k$$

Within the simulation, it is assumed that the shape of the crystal resulting from aggregation is identical to that of the larger crystal participating in the aggregation event. The crystal shape is assumed to be constant only during the aggregation event. In other words, after the aggregation process between two participating particles is finished, the resulting aggregate will grow to a larger crystal with a shape which is determined by the subsequent supersaturation level. Although we only considered the binary collision between crystals, all the possible pairwise combinations among crystals with different sizes are considered, and the formation of new aggregates is updated every 0.5 s. Within this context, the collision of multiple crystals at the same time can be viewed as multiple successive binary collisions. The reader may refer to ref 32 for more extensive simulation studies on the phenomenological effect of aggregation on crystal shape distributions and on the comparison of crystal shape distributions obtained from the multiscale model and the population balance model.

Parallel Computation of Multiscale Model. The simulation of the crystal growth process of crystals formed via nucleation is executed in parallel by using MPI through which we are able to divide the crystals to multiple cores by achieving the distribution of the computational cost and memory requirements. More detailed discussion on the step-by-step parallelization of the crystallization process multiscale model that incorporates nucleation, crystal growth, and aggregation processes will be included below.

Decomposition. We can decompose the nucleation and crystal growth processes in a batch crystallization system into collection of tasks where each task is the crystal growth of a nucleated crystal. Furthermore, the time required to run one batch simulation can be further reduced by introducing a new variable N_{rp} to indicate the number of crystals represented by a single crystal that is actually running on a core. It can be viewed as a coarse-grained model in the sense that one crystal running in the kMC simulation actually represents N_{rp} crystals. At the nucleation stage, after we wait for the time required for the nucleation of N_{rp} crystals, we initiate the growth of a single crystal in the kMC simulation that actually represents a group of N_{rp} crystals. Therefore, the size and shape of the representing crystal changes with time as it grows to a larger crystal. If N_{rp} = 1000, then a single crystal running in the simulation will now represent 1000 actual crystals. Thus, we are only required to run 10 crystals in the simulation to represent 10000 crystals, and as a result, we compromise the computation time saving with the accuracy of the simulation results.

Assignment. More specifically, as soon as a crystal is nucleated, it will be assigned to one of the available cores, and it will grow to a larger crystal via the kMC simulation (The reader may refer to ref 42 for the equation used to determine how many crystals are nucleated.) Because crystals are continuously nucleated until the end of the batch process (i.e., at some point in time, the number of crystals growing in the crystallizer will exceed the number of cores), additional crystals need to be assigned to each core throughout the kMC simulation, which makes this an irregular problem where the total size of tasks assigned to each core grows with time. More specifically, nucleated crystals are assigned following the order described in Table 2 (i.e., crystal number modules is equal to the number of cores available). We note that this is a number-based allocation, assuming that all cores have identical processor speed and memory.

Table 2. Order in which Nucleated Crystals Are Assigned to Each Core[a]

core	crystal number	crystal number
worker 1	1	$n + 1$
worker 2	2	$n + 2$
⋮	⋮	⋮
worker n	n	$2n$

[a]Suppose that there are $2n$ crystals.

The volume of each class is a range, and crystals are allocated to each class if their volumes fall within the volume range corresponding to the specific class. Therefore, as crystals grow to larger crystals, they will move from one class to another; thus, the number of crystals contained in each class changes with time. Subsequently, the successful collision probability between crystals of class i and class j (i.e., crystals with volumes V_i and V_j, respectively) during a time period can be calculated via eq 9. Next, a random number from [0, 1) is generated, and the aggregation event is executed if the random number is less than the successful collision probability. If an aggregation event occurs between class i and j, then the volume of the smaller crystal will be removed from the kMC simulation and will be added to the larger one, forming an aggregate whose volume is equal to the total volume of the two crystals participating in the aggregation event. This process applies to all possible pairwise combinations of crystal volumes over the course of the batch

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

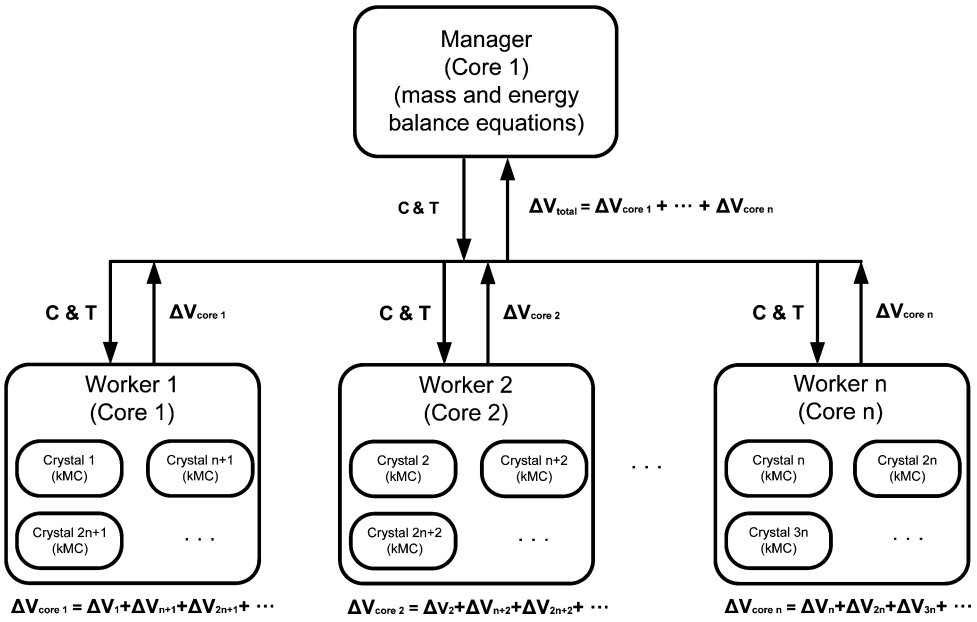

Figure 3. Manager−worker parallel computation scheme for multiscale model of batch crystallization process.

crystallization simulation. Furthermore, the computation requirement for the manager is not significant; thus, the core assigned to the manager is also used as a worker to contribute to the parallel computation as shown in Table 2 and Figure 3.

Orchestration. Figure 3 illustrates schematically how the information passing between the cores is managed with the MPI settings in order to link the macroscopic model (i.e., mass and energy balance equations for the continuous phase) to the microscopic model (i.e., kMC model). The coupled simulation follows the manager−worker MPI computational scheme: There is a core (i.e., manager) that is responsible for collecting the change in the total volume of crystals assigned to each core (i.e., worker) at each time step required for parallel processing, which corresponds to the amount of solute transported from the continuous phase to the crystal surface. Then, the manager core computes the change in the total volume of the crystals in the crystallizer at each time step and computes the protein solute concentration C and the temperature T for the continuous phase in the crystallizer using mass and energy balance equations. The updated C and T will be sent back to the worker cores, and those values at each core will remain identical until they are updated again after a time step. Then, the crystals assigned to each core will grow under the updated conditions via kMC simulations.

In the parallel computation, we only consider synchronous iterations and synchronous communication (SISC). At each time step, all processors wait until they have received all the data computed by the other processors at the previous task execution step, before beginning their following computations. The design of SISC parallelized code is quite straightforward using the MPI setting. Please note that the SISC penalizes algorithms on systems with a slow and heterogeneous cluster.

Pseudocode.

Algorithm 1 Parallel computation of the multiscale batch crystallization process model

```
for i = 1 → n do
  if i == 1 then                                              ▷ Manager core
    1. assign nucleated crystals over Δt to each core according to Table 2
    2. compute ΔV_total(t) = Σ_{i=1}^{n} ΔV_core i(t)
    3. update C(t) and T(t) through Eqs. 7–8
    4. compute the total number of successful collisions between crystals via Eq. 9
  else                                                        ▷ Worker core
    1. have crystals assigned to each core grow via Eqs. 4–6
    2. compute ΔV_core i(t) and send it to the manager core
  end if
end for
```

Please note that $\Delta V_{\text{total}}(t)$ is the change in the total volume of crystals in the crystallizer from $t - \Delta t$ to t seconds, and $\Delta V_{\text{core } i}(t)$ is the change in the total volume of crystals particularly assigned to the core i from $t - \Delta t$ to t seconds. The time step to calculate $V(t)$, $C(t)$, and $T(t)$ is 0.033 s. The manager core computes the number of crystals nucleated over a time period for a given supersaturation level. Then, the nucleated crystals are assigned to each core in a round-robin fashion. Specifically, the computation of $\Delta V_{\text{core } i}$ for those crystals assigned to each worker is related to the computation of the total amount of solute consumed (similarly, heat generated) by crystal growth over the entire crystallization process through the mass and energy balance equations. The small crystal participating in the aggregation event will be removed from the core on which the small crystal was growing; thus, this core will run with one less crystal after the aggregation event. The reader may refer to refs 42−44 for the use of the parallelization scheme to different applications, including the plug flow crystallizer and the continuous-stirred tank crystallizer with a fines trap and a product classification unit.

Theoretically, by decoupling of the simulation of surface migration via a 2D random walk process, we can further improve the computation efficiency as it is shown by refs 24 and 25. However, it is not necessary in this work because the order of magnitude difference among the migration rate and those of other processes in the crystallization system is not as significant as the one in thin-film solar cells.

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

Results. In this work, the Hoffman2 cluster, which consists of 1200 nodes with a total of 13 340 cores and over 50 TB of memory, is used along with MPI settings for all of the simulations. The memory of each node varies from 8 to 128 GB, and there are many types of CPUs including Intel and AMD, with 8, 12, and 16 cores, respectively. Because of the variety of the CPU types, if a small number of cores (e.g., 1, 2, 4, and 8 cores) are requested to the Hoffman2 cluster for the simulation of the multiscale model of the batch crystallization process, then it is possible to get one bad CPU that consists of many bad cores, which will result in poor parallelization performance. To circumvent this issue, among the many CPU types in the Hoffman2 cluster, Intel-E5530 with 8 cores is specifically requested and used for the construction of the plots and tables presented in this section.

It is shown in Table 3 that the simulation times required to complete a batch simulation decrease as the number of cores is increased. Also, Figure 4 shows that as the number of cores is doubled the speedup achieved in comparison to the theoretical maximum speedup (i.e., the theoretical maximum speedup should be n times when n processors are used) decreases because of overhead costs generated by communication taking place between multiple cores. Overall, it is clear that the batch crystallization process greatly benefits from the use of MPI for the kMC simulations. Furthermore, we have compared concentration and temperature profiles obtained from the kMC simulations with different numbers of cores, and it is verified in Figures 5 and 6 that the concentration and temperature profiles obtained from the kMC simulations with n_{cores} = 1 and 64 are identical. Therefore, we can conclude that the computation time savings is achieved by the parallelized computations and the accuracy of the simulation results is maintained. If the multiscale framework is not properly parallelized, then the speedup achieved by the parallel computation scheme could be the result of compromising the accuracy of the simulation results. We note that the evolution of average crystal size and shape distributions is determined by the supersaturation level over the course of the batch process, which in turn is a function of the solute concentration and of the temperature in the crystallizer. Therefore, by comparing the solute concentration and the temperature rather than the crystal size and shape distributions, we can improve our fundamental understanding of the underlying dynamics of the crystallization process.

Table 3. Time Required to Run a Batch Simulation and the Speedup Achieved by Using Different Numbers of Cores[a]

n_{cores}	time (h)	speed-up (times)	theoretical speed-up (times)
1	34.97	1.00	1
2	17.63	1.98	2
4	8.98	3.89	4
8	4.71	7.44	8
16	2.47	14.18	16
32	1.38	25.28	32
64	0.92	38.15	64

[a]Note that n_{cores} is the number of cores and that the speedup is defined as t_1/t_n, where t_1 is the time the process takes on 1 core and t_n is the time the process takes on n cores.

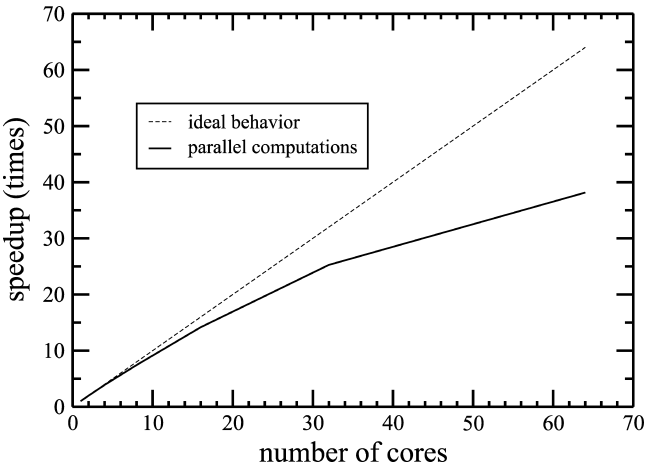

Figure 4. Speedup achieved by the parallel computation of the batch crystallization process multiscale model under open-loop operation with the number of cores used for the kMC simulation describing the evolution of crystal shape distribution. The ideal behavior represents the theoretical maximum speedup.

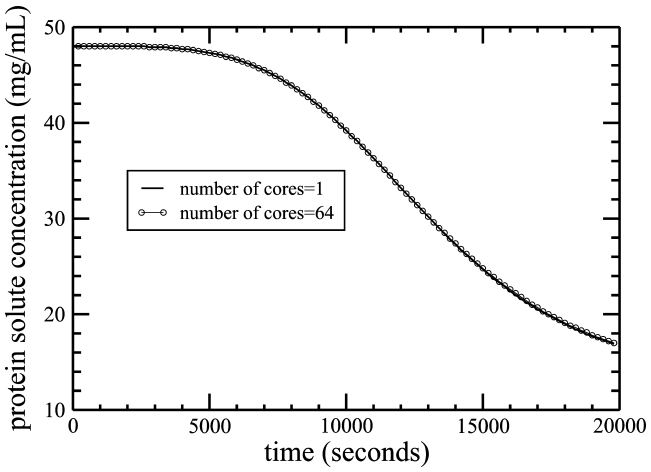

Figure 5. Profile of the protein solute concentration with time obtained under the open-loop operation of the batch crystallization process multiscale model for two different numbers of cores: 1 and 64.

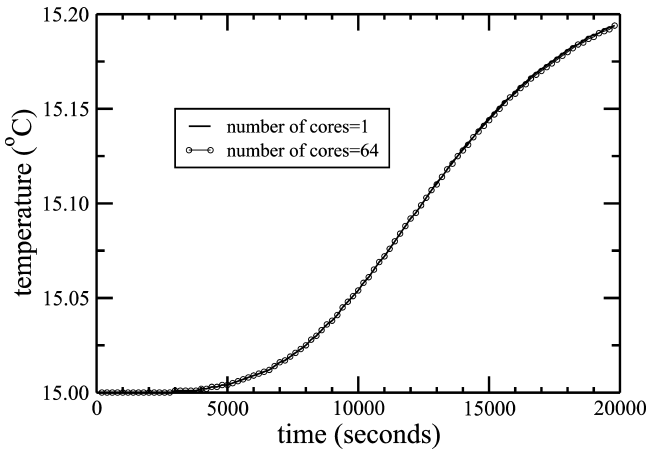

Figure 6. Profile of the crystallizer temperature with time obtained under the open-loop operation of the batch crystallization process multiscale model for two different numbers of cores: 1 and 64.

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

In the parallel computations, we might observe a superlinear speed-up behavior (i.e., speedup is greater than n times when n processors are used), as is shown in Figure 7. In Figure 7, the

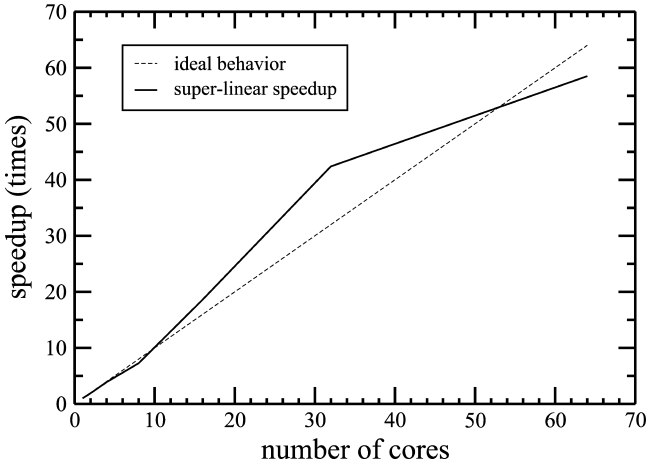

Figure 7. Speedup achieved by the parallel computation of the batch crystallization process multiscale model under open-loop operation with the number of cores used for the kMC simulation: superlinear speedup vs ideal speedup are compared.

batch crystallization process is operated at a higher supersaturation level than that of Figure 4. As a result, more crystals are nucleated, increasing the number of crystals assigned to each core significantly (thereby increasing the computational requirement on each core). In general, the superlinear speedup in low-level computations is caused by the cache effect due to the different memory hierarchies of a modern computer.[31] More specifically, with the larger accumulated cache size, more simulation tasks can fit into caches; thus, the memory access time required to reach the higher level (e.g., RAM) for additional memory can be reduced significantly, which results in extra speedup on top of that is achieved by parallelizing the serial computations.

In particular, when we do not have access to a sufficient number of identical CPUs, we can still improve the computational efficiency of the parallelized computations by introducing a new variable N_{rp}. Table 4 shows that we can

Table 4. Time Required to Finish the Batch Crystallization Process under the Open-Loop Operation by Varying N_{rp} for n_{cores} = 64

N_{rp}	time (h)
1	34.97
10	4.41
50	1.87
100	0.557
500	0.236

significantly reduce the simulation time required to run one batch simulation by increasing N_{rp}. However, it is shown in Figure 8 that the speedup achieved by increasing n_{cores} decreases as N_{rp} is increased. This is because increasing N_{rp} may unnecessarily simplify the sequential problem such that it is not significantly beneficial to further parallelize the kMC simulation. Furthermore, it is shown in Figures 9 and 10 that as N_{rp} is increased (i.e., each crystal represents more crystals, less crystals are used to run the same batch crystallization process,

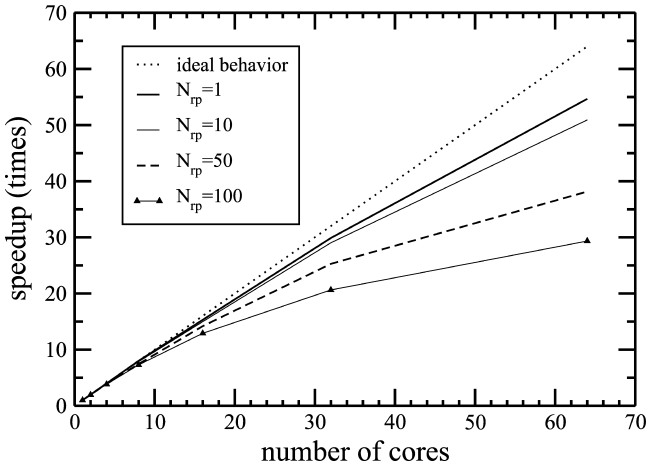

Figure 8. Speedup achieved by the parallel computation of the batch crystallization process multiscale model under the open-loop operation with the number of cores used for the kMC simulation for different N_{rp} values.

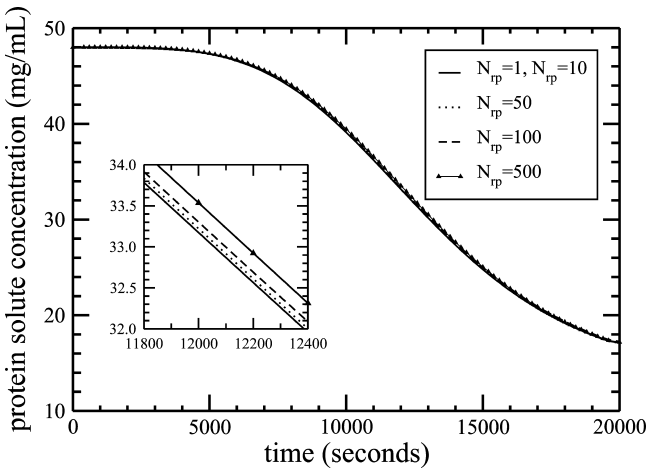

Figure 9. Profile of the protein solute concentration with time obtained under the open-loop operation of the batch crystallization process multiscale model for different N_{rp} values. The inset shows the C profile for t = 11 800−12 400 s and C = 32.0−34.0 mg/mL.

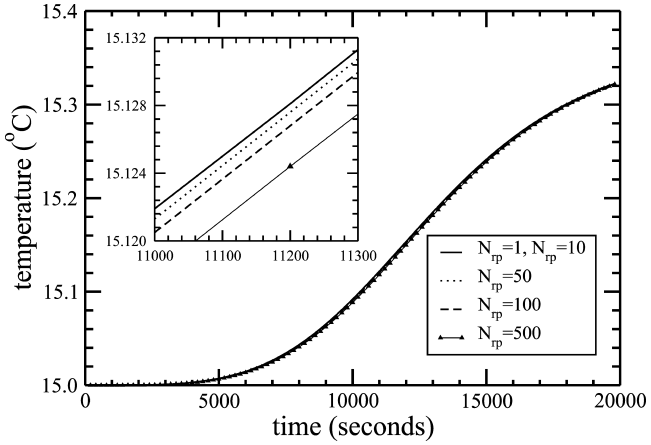

Figure 10. Profile of the crystallizer temperature with time obtained under the open-loop operation of the batch crystallization process multiscale model for different N_{rp} values. The inset shows the T profile for t = 11 000−11 300 s and T = 15.120−15.132 °C.

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

and error may thus follow) the concentration and the temperature profiles obtained from the parallelized kMC simulation progressively deviate from those profiles obtained from the kMC simulation with N_{rp} = 1. By introducing N_{rp}, therefore, we may compromise the accuracy of the simulation results to decrease computation time.

Increasing the stirrer speed induces the aggregation process, which requires transferring crystals from one core to another in parallel computations to model the aggregation of two crystals. Thus, additional overhead costs associated with transferring crystals will be incurred, and as a result, the speedup curve shifts downward as more aggregates are formed, which is shown in Figure 11.

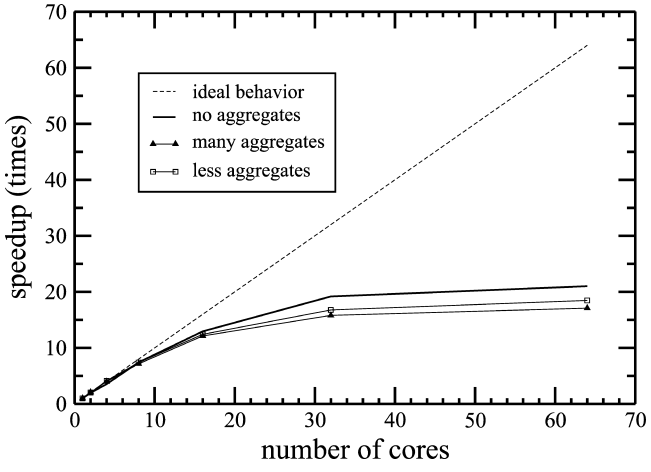

Figure 11. Speedup achieved by the parallel computation of the batch crystallization process multiscale model under open-loop operation with the number of cores used for the kMC simulation under conditions where aggregates are formed.

Initially, the parallel computation scheme is applied to open-loop systems where no feedback control is applied to the crystallization system. Then, the same parallel computation scheme is used to improve the computation efficiency for the closed-loop system where a model predictive control scheme is used to regulate the shape distribution of crystals produced at the end of the batch process by manipulating the crystallizer jacket temperature. Specifically, an in-batch model-predictive controller with formulation described in ref 40 is implemented for the batch crystallization process for the production of crystals with a desired crystal shape distribution at the end of the batch process (Figure 13). In this work, the crystal shape is defined as the ratio between the average crystal heights in the direction of the (110) and (101) faces, $\alpha = \frac{\langle h_{110} \rangle}{\langle h_{101} \rangle}$. It is shown in Figure 12 that the computation time can be reduced by the parallel computation of the closed-loop batch crystallization process multiscale model without any sacrifice of the accuracy of the calculated crystal shape distribution (Figure 13); however, the speedup achieved by the parallel computation depends on the desired set-point value for the crystal shape distribution. For example, the optimal supersaturation level to produce crystals with the higher set-point value, α_{set} = 1.05, favors the nucleation and crystal growth processes; thus, the computation time is significantly reduced by parallelizing the kMC simulation. In contrast, for the lower set-point value, α_{set} = 0.85, the optimal supersaturation level is low, and the nucleation and crystal growth processes are less favored, in

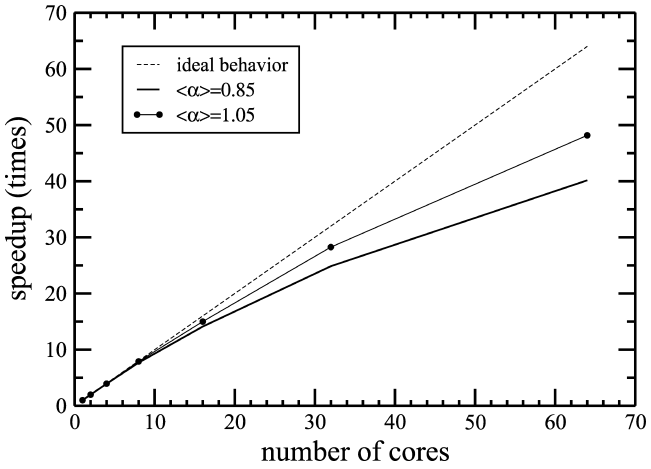

Figure 12. Speedup achieved by the parallel computation of the batch crystallization process multiscale model under closed-loop controlled operation with the number of cores used for the kMC simulation under different set-point values: $\langle \alpha \rangle$ = 1.05 and $\langle \alpha \rangle$ = 0.85.

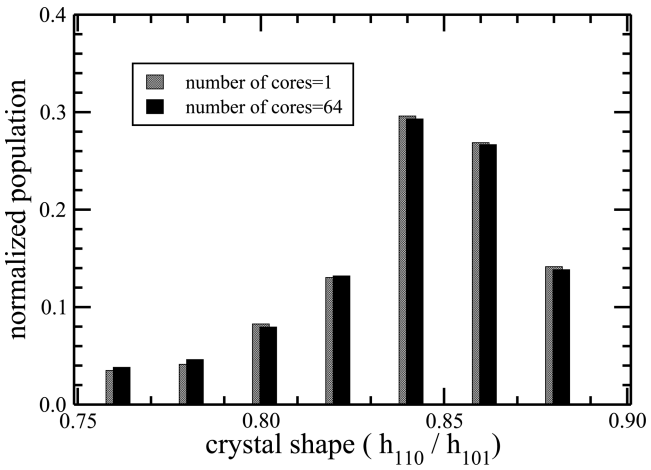

Figure 13. Normalized crystal shape distribution obtained under the closed-loop operation of the batch crystallization process multiscale model for two different numbers of cores: 1 and 64. The desired set-point value for the average crystal shape is α_{set} = 0.85.

which case the speedup achieved via the parallel computations is not as significant as the one for α_{set} = 1.05. The reader may find more results on the morphological evolution of lysozyme crystals from one of the previous works of our group. (See refs 32, 42, and 43.)

Remark 1 *The evolution of crystal shape is determined by the supersaturation level during the crystallization process. Specifically, at a high supersaturation level, the crystal growth rate in the direction of the (110) face is higher than that of the (101) face, and as a result, the production of crystals with a high shape factor (e.g., high aspect ratio $\alpha = h_{110}/h_{101}$) is favored. Also, the higher supersaturation level leads to faster nucleation and crystal growth processes. In contrast, at lower supersaturation levels, the production of crystals with a lower aspect ratio is favored, and the corresponding rates of the nucleation and crystal growth processes are lower compared to those at higher supersaturation levels.*

CONCLUSIONS

In this work, a parallelized multiscale, multidomain modeling scheme was proposed to directly reduce the computation time

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

and memory requirements without compromising the accuracy of simulation results. The parallelized multiscale modeling strategy that consists of the three steps of decomposition, assignment, and orchestration was applied to a batch crystallization process multiscale model. First, we decomposed the nucleation and crystal growth processes in the batch crystallization system into a collection of tasks where each task represents the crystal growth of a nucleated crystal. Second, tasks were assigned to processors according to a modulus function (i.e., round-robin allocation). Third, a manager−worker MPI computation scheme was used to link the macroscopic model (e.g., mass and energy balance equations for the continuous phase) to the microscopic models (e.g., kinetic Monte Carlo model). Using the proposed parallel computation scheme, a significant decrease in the time required to run the batch crystallization process multiscale model under both open-loop and closed-loop operations was achieved as the number of cores was increased. In particular, the performance of the parallel computation scheme applied to the closed-loop system was dependent on the desired crystal shape. Furthermore, we extended the use of the parallel computation scheme to the crystallization system with a high stirrer speed, in which case many crystal aggregates were formed, and we evaluated the effect of aggregation on the parallelization.

AUTHOR INFORMATION

Corresponding Author

*E-mail: pdc@seas.ucla.edu.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

We gratefully acknowledge the financial support from the Extreme Science and Engineering Discovery Environment (TG-CCR120003) and the National Science Foundation (CBET-0967291).

REFERENCES

(1) Maroudas, D. Multiscale modeling of hard materials: challenges and opportunities for chemical engineering. *AIChE J.* **2000**, *46*, 878−882.

(2) Kevrekidis, I. G.; Gear, C. W.; Hummer, G. Equation-free: the computer-aided analysis of complex multiscale systems. *AIChE J.* **2004**, *50*, 1346−1355.

(3) Vlachos, D. G. A review of multiscale analysis: examples from systems biology, materials engineering, and other fluid-surface interacting systems. *Adv. Chem. Eng.* **2005**, *30*, 1−61.

(4) Christofides, P. D.; Armaou, A.; Lou, Y.; Varshney, A. *Control and Optimization of Multiscale Process Systems*; Birkhäuser, Boston, MA, 2009.

(5) Ayton, G.; Noid, W.; Voth, G. Multiscale modeling of bimolecular systems: in serial and in parallel. *Curr. Opin. Struct. Biol.* **2007**, *17*, 192−198.

(6) Frantzdale, B.; Plimpton, S.; Shephard, M. Software components for parallel multiscale simulation: an example with LAMMPS. *Engineering With Computers* **2010**, *26*, 205−211.

(7) Avila, C. L.; Drechse, J. D. N.; Alcantara, R.; Villa-Freixa, J. Multiscale molecular dynamics of protein aggregation. *Curr. Protein Pept. Sci.* **2011**, *12*, 221−234.

(8) Cheimarios, N.; Kokkoris, G.; Boudouvis, A. An efficient parallel iteration method for multiscale analysis of chemical vapor deposition process. *Appl. Numerical Math.* **2013**, *67*, 78−88.

(9) Pesce, L.; Lee, H.; Hereld, M.; Visser, S.; Stevens, R.; Wildeman, A.; Van Drongelen, W. Large-scale modeling of epileptic seizures: scaling properties of two parallel neuronal network simulation algorithms. *Comp. & Math. Methods in Medicine* **2013**, *2013*, 182145.

(10) Guo, N.; Zhao, J. A coupled FEM/DEM approach for hierachical multiscale modeling of granular media. *Int. J. for Numerical Methods in Eng.* **2014**, *99*, 789−818.

(11) Griebel, M.; Ruttgers, A. Multiscale simulations of three-dimensional viscoelastic flows in a square-square contraction. *J. Non-Newtonian Fluid Mech.* **2014**, *205*, 41−63.

(12) Seny, B.; Lambrechts, J.; Toulorge, T.; Legat, V.; Remacle, J. An efficient parallel implementation of explicit multirate Runge-Kutta schemes for discontinuous Galerkin computations. *J. Comput. Phys.* **2014**, *256*, 135−160.

(13) Mosby, M.; Matous, K. Hierarchically parallel coupled finite strain multiscale solver for modeling heterogeneous layers. *Int. J. Numerical Methods in Eng.* **2015**, *102*, 748−765.

(14) Arbogast, T.; Xiao, H. Two-level mortar domain decomposition preconditioners for heterogeneous elliptic problems. *Comp. Methods in Appl. Mech. and Eng.* **2015**, *292*, 221−242.

(15) Ren, R.; Orkoulas, G. Parallel Markov Chain Monte Carlo simulations. *J. Chem. Phys.* **2007**, *126*, 211102.

(16) O'Keeffe, C. J.; Orkoulas, G. Parallel canonical Monte Carlo simulations through sequential updating of particles. *J. Chem. Phys.* **2009**, *130*, 134109.

(17) O'Keeffe, C. J.; Ren, R.; Orkoulas, G. Spatial updating grand canonical Monte Carlo algorithms for fluid simulation: generalization to continuous potentials and parallel implementation. *J. Chem. Phys.* **2007**, *127*, 194103.

(18) Landau, D.; Binder, K. *A guide to Monte Carlo simulations in statistical physics*; Cambridge University Press: New York, 2000.

(19) Lam, R.; Vlachos, D. G. Multiscale model for epitaxial growth of films: Growth mode transition. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2001**, *64*, 035401.

(20) Hansen, U.; Rodgers, S.; Jensen, K. Modeling of metal thin film growth: linking angstrom-scale molecular dynamics results to micron-scale film topographies. *Phys. Rev. B: Condens. Matter Mater. Phys.* **2000**, *62*, 2869−2878.

(21) Pricer, T.; Kushner, M.; Alkire, R. Monte Carlo simulation of the electrodeposition of copper 1. Additive-free acidic sulfate solution. *J. Electrochem. Soc.* **2002**, *149*, C396−C405.

(22) Ricardez-Sandoval, L. Current challenges in the design and control of multiscale systems. *Can. J. Chem. Eng.* **2011**, *89*, 1324.

(23) Rasoulian, S.; Ricardez-Sandoval, L. Uncertainty analysis and robust optimization of multiscale process systems with application to epitaxial thin film growth. *Chem. Eng. Sci.* **2014**, *116*, 590−600.

(24) Crose, M.; Kwon, J. S.-I.; Nayhouse, M.; Ni, D.; Christofides, P. D. Multiscale modeling and operation of PECVD of thin film solar cells. *Chem. Eng. Sci.* **2015**, *136*, 50−61.

(25) Tsalikis, D.; Baig, C.; Mavrantzas, V.; Amanatides, E.; Mataras, D. A hybrid kinetic Monte Carlo method for simulating silicon films grown by plasma-enhanced chemical vapor deposition. *J. Chem. Phys.* **2013**, *139*, 204706.

(26) Zheng, Z.; Stephens, R.; Braatz, R.; Alkire, R.; Petzold, L. A hybrid multiscale kinetic Monte Carlo method for simulation of copper electrodeposition. *J. Comput. Phys.* **2008**, *227*, 5184−5199.

(27) Drews, T.; Krishnan, S.; Alameda, J.; Gannon, D.; Braatz, R.; Alkire, R. Multi-scale simulations of copper electrodeposition onto a resistive substrate. *IBM J. Res. Dev.* **2005**, *49*, 49−63.

(28) Li, X.; Drews, T.; Rusli, E.; Xue, F.; He, Y.; Braatz, R.; Alkire, R. The effect of additives on shape evolution during electrodeposition. Part I: Multiscale simulation with dynamically coupled kinetic Monte Carlo and moving-boundary finite-volume codes. *J. Electrochem. Soc.* **2007**, *154*, D230−D240.

(29) Rusli, E.; Xue, F.; Drews, T.; Vereecken, P.; Andricacos, P.; Deligianni, H.; Braatz, R.; Alkire, R. Effect of additives on shape evolution during electrodeposition. Part II: Parameter estimation from roughness evolution experiments. *J. Electrochem. Soc.* **2007**, *154*, D584−D597.

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914

(30) Gunawan, R.; Fusman, I.; Braatz, R. Parallel high-resolution finite volume simulation of particulate processes. *AIChE J.* **2008**, *54*, 1449−1458.
(31) Culler, D.; Singh, J.; Gupta, A. *Parallel computer architecture: a hardware/software approach*; Morgan Kaufmann Publishers: San Francisco, CA, 1999.
(32) Kwon, J. S.-I.; Nayhouse, M.; Christofides, P. D.; Orkoulas, G. Modeling and control of shape distribution of protein crystal aggregates. *Chem. Eng. Sci.* **2013**, *104*, 484−497.
(33) Muschol, M.; Rosenberger, F. Liquid-liquid phase separation in supersaturated lysozyme solutions and associated precipitate formation/crystallization. *J. Chem. Phys.* **1997**, *107*, 1953−1962.
(34) Panagiotopoulos, A. Monte Carlo methods for phase equilibria of fluids. *J. Phys.: Condens. Matter* **2000**, *12*, R25−R52.
(35) Oksanen, E.; Blakeley, M.; Bonnete, F.; Dauvergne, M.; Dauvergne, F.; Budayova-Spano, M. Large crystal growth by thermal control allows combined X-ray and neutron crystallographic studies to elucidate the protonation states in Aspergillus flavus urate oxidase. *J. R. Soc., Interface* **2009**, *6*, S599−S610.
(36) Nayhouse, M.; Heng, V. R.; Amlani, A. M.; Orkoulas, G. Simulation of Phase Boundaries Using Constrained Cell Models. *J. Phys.: Condens. Matter* **2012**, *24*, 375105.
(37) Durbin, S. D.; Feher, G. Simulation of lysozyme crystal growth by the Monte Carlo method. *J. Cryst. Growth* **1991**, *110*, 41−51.
(38) Ke, S. C.; DeLucas, L. J.; Harrison, J. G. Computer simulation of protein crystal growth using aggregates as the growth unit. *J. Phys. D: Appl. Phys.* **1998**, *31*, 1064−1070.
(39) Kwon, J. S.; Nayhouse, M.; Christofides, P. D.; Orkoulas, G. Modeling and control of protein crystal shape and size in batch crystallization. *AIChE J.* **2013**, *59*, 2317−2327.
(40) Kwon, J. S.; Nayhouse, M.; Orkoulas, G.; Ni, D.; Christofides, P. D. A method for handling batch-to-batch parametric drift using moving horizon estimation: application to run-to-run MPC of batch crystallization. *Chem. Eng. Sci.* **2015**, *127*, 210−219.
(41) Kusters, K. A.; Wijers, J. D.; Thoenes, D. Aggregation kinetics of small particles in agitated vessels. *Chem. Eng. Sci.* **1997**, *52*, 107−121.
(42) Kwon, J. S.-I.; Nayhouse, M.; Orkoulas, G.; Christofides, P. D. Crystal shape and size control using a plug flow crystallization configuration. *Chem. Eng. Sci.* **2014**, *119*, 30−39.
(43) Kwon, J. S.; Nayhouse, M.; Christofides, P. D.; Orkoulas, G. Modeling and control of crystal shape in continuous protein crystallization. *Chem. Eng. Sci.* **2014**, *107*, 47−57.
(44) Kwon, J. S.; Nayhouse, M.; Orkoulas, G.; Christofides, P. D. Enhancing crystal production rate and reducing polydispersity in continuous protein crystallization. *Ind. Eng. Chem. Res.* **2014**, *53*, 15538−15548.

DOI: 10.1021/acs.iecr.5b02942
Ind. Eng. Chem. Res. 2015, 54, 11903−11914