Journal of Computational Physics 231 (2012) 1781–1796

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

An implicit parallel multigrid computing scheme to solve coupled thermal-solute phase-field equations for dendrite evolution

Z. Guo [a,*], J. Mi [b], P.S. Grant [a]

[a] Department of Materials, University of Oxford, Parks Road, Oxford OX1 3PH, UK
[b] Department of Engineering, University of Hull, East Yorkshire HU6 7RX, UK

ARTICLE INFO

Article history:
Received 27 February 2011
Received in revised form 3 November 2011
Accepted 5 November 2011
Available online 16 November 2011

Keywords:
Parallel computing
Multigrid method
Phase field
Dendrite evolution
Solidification

ABSTRACT

An implicit, second-order space and time discretization scheme together with a parallel multigrid method involving a strip grid domain partitioning has been developed to solve fully coupled, nonlinear phase field equations involving solute and heat transport for multiple solidifying dendrites. The computational algorithm has been shown to be stable and monotonously convergent, and allowed time marching steps that were 3–4 orders of magnitude larger than those employed in similar explicit approaches, resulting in an increase of 3–4 orders of magnitude in computing efficiency. Full solute and thermal coupling was achieved for metallic alloys with a realistic, high Lewis number of $>10^4$. The parallel multigrid computing scheme is shown to provide a scalable methodology that allowed the efficient use of distributed supercomputing resource to simulate the evolution of tens of complex shaped 2D dendrites in a computational domain containing tens or even hundreds of millions of grid points. The simulations have provided insight into the dynamic interplay of many growing dendrites in a more realistic fully coupled thermal-solute condition, capturing for the first time fine scale features such as dendrite splitting.

1. Introduction

Since the successful demonstration of the phase field (PF) approach for the simulation of dendrite shape during solidification by Kobayashi in 1994 [1], the PF method has been developed intensively worldwide for the simulation of pure metal and alloy dendrite microstructural evolution [2–7]. The PF method employs a diffuse interface concept with a continuous PF variable ϕ that varies smoothly but steeply across a diffuse interface that represents the separation between solid and liquid phases and avoids the need for explicit tracking of the physical position of the solid–liquid interface [8]. Generally, the evolution of ϕ is governed by the spatial and temporal distribution of alloy internal energy (specific and latent heat) and the interface gradient energy, and the introduction of ϕ produces a set of partial differential equations (PDEs) governing the phase field, solute and temperature. Because temperature and the composition of both liquid and solid phases in an alloy are linked by thermodynamic considerations, the PDEs are strongly coupled and non-linear, and consequently are difficult to solve efficiently. Hence, in most of the studies concerning PF based simulations of dendrite evolution during solidification reported so far simplified version of the PDEs have been used to decouple the thermal and solute fields e.g. dendrite growth in an isothermal field (without transient heat transfer) [9], at a constant pre-determined cooling rate [10], or under an applied predefined and fixed thermal gradient [11,12]. Finite difference discretization and an explicit time-marching method have generally been used to solve the discretized PDEs. Consequently, in order to make the complex, coupled case of alloy

* Corresponding author.
E-mail address: gzp04@mails.tsinghua.edu.cn (Z. Guo).

0021-9991/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2011.11.006

dendritic solidification tractable, the resulting simulations either omit some of the underlying physics, or predict unrealistic dendrite morphologies compared with reality. Further serious challenges concern the time-stepping constraint and small discretization distances required for stability in the explicit method so that computing times become enormous and even the most powerful supercomputers can calculate dendrite shape evolution for only a few tens of dendrites, making it very difficult, if not impossible, to simulate microstructures over meaningful volumes of material with a sensible computer burden, and with sufficient underlying physics (linked solute and thermal transport), to be insightful for practical solidification problems.

To take the coupled thermal-solute effect into account, Karma and Rappel [13,14] and Ramirez et al. [15] have recently developed a coupled thermal-solute PF model for dilute binary alloys. By introducing a term named the "anti-trapping" current to the solute conservation equation, non-equilibrium effects such as interface stretching and surface diffusion effects arising when the solid and liquid diffusivities are unequal were eliminated, and very good quantitative agreement between simulation and analytical equations based on the Gibbs–Thomson equation for curvature effects on solid–liquid interfaces were obtained. However, the required coupling of a thermal field into the solidifying system introduces further computing complexity due to the multi-scale character of the very large difference between the thermal and solute diffusion rate characterized by the Lewis number i.e. the ratio between the thermal and solute diffusivities, which is typically $\sim 10^4$ for metallic alloys. As a consequence, for the study of fully coupled thermal-solute dendrite growth in metallic alloys, typically only one quarter or half of a solidifying primary dendrite can be simulated, and even then an artificially low Lewis number of order of magnitude 10^1–10^2 must be assumed [15] in order to reduce computational cost.

In an effort to address more practical Lewis numbers, length and time scales, implicit rather than explicit algorithms have been developed. Rosam et al. [16,17] presented an adaptive mesh, multigrid algorithm and showed that this approach could simulate successfully dendrite shape evolution during solidification at a lower computational cost by refining the discretized grid at the diffuse interface area only. More importantly, due to the inherent high stability of the implicit approach, limits on the Lewis number were removed. In comparison with the other numerical methods that might be applied to phase field equations such as the generalized minimal residual (GMRES) or conjugated gradient (CG) [18], a multigrid approach can provide solutions involving a number of computational operations that are near linearly proportional to the problem scale or dimension [19].

In summary, the many phase field approaches to the simulation of the way in which dendrite shape evolves during solidification usually suffer from one or more of the following restrictions:

(1) artificially low Lewis number,
(2) very small length and time scales associated with the explicit method,
(3) a decoupling of thermal and solute fields (that are strongly coupled in practice), and
(4) only a few dendrites can be considered and therefore microstructure prediction is non-sensible.

In this paper we present a new numerical approach with dramatic increases in computational efficiency for the phase field method in order to simulate the evolution of dendritic microstructures, including multiple dendrite growth, impingement and solute segregation. The approach is a major extension of the multigrid algorithm applied by Rosam et al. [16,17] but implemented here with a new highly parallelized computing scheme. As a consequence of the improved robustness and computational efficiency, we then present results for the first time of multiple dendrite growth for realistic Lewis numbers of $\sim 10^4$ corresponding to metallic alloys in a fully coupled thermal-solute field, including secondary dendrite arm stretching and dendrite impingement.

2. The phase field model

2.1. Governing equations

The coupled thermal-solute PF model for the solidification of dilute binary alloys proposed by Ramirez et al. [15] was adopted in this study. The governing PDEs are:

$$\frac{\partial \phi}{\partial t} = -K_\phi \frac{\delta F}{\delta \phi} \tag{1}$$

$$\frac{\partial c}{\partial t} = \vec{\nabla} \cdot \left(K_c \vec{\nabla} \frac{\delta F}{\delta c} - \vec{j}_{at} \right) \tag{2}$$

$$\frac{\partial T}{\partial t} = \alpha \nabla^2 T + \frac{L}{2c_p} \frac{\partial \phi}{\partial t} \tag{3}$$

where t is time, ϕ, c, T are phase field, solute concentration (molar), and temperature respectively, K_ϕ and K_c are constants, α is thermal diffusivity, L is latent heat, and c_p is alloy specific heat. F is the system free energy and during the solidification of a dilute binary alloy is given by [15]:

$$F[\phi, c, T] = \int dV \left[\frac{\sigma}{2}|\nabla\phi|^2 + f_{AB}(\phi, c, T)\right] \tag{4}$$

where σ is the gradient free energy coefficient, f_{AB} denotes the bulk free energy density of a binary mixture of A and B atoms/molecules, $\vec{j}_{at}$ is an "antitrapping" current introduced by Karma [8] to counter-balance spurious effects at the PF diffusion interface:

$$\vec{j}_a = -\frac{W_0}{\sqrt{2}} \frac{c/c_\infty}{[1+k-(1-k)\phi]} \frac{\partial\phi}{\partial t} \frac{\nabla\phi}{|\nabla\phi|} \tag{5}$$

where W_0 denotes the interface thickness, c_∞ is the initial solute concentration, and k is equilibrium concentration partition coefficient obtained from the binary alloy phase diagram.

According to thin interface limit analysis [8,15], the anisotropic dimensionless forms of the PF equations can be written as:

$$A(\psi)^2\left[\frac{1}{Le} + Mc_\infty(1+(1-k)U)\right]\frac{\partial\phi}{\partial t} = \nabla\cdot(A(\psi)^2\nabla\phi) - \frac{\partial}{\partial x}\left(A(\psi)A(\psi)'\frac{\partial\phi}{\partial y}\right) + \frac{\partial}{\partial y}\left(A(\psi)A(\psi)'\frac{\partial\phi}{\partial x}\right) + \phi(1-\phi^2) - \lambda(1-\phi^2)^2(\theta + Mc_\infty U)(1+\dot{\xi}) \tag{6}$$

$$\left(\frac{1+k}{2} - \frac{1-k}{2}\phi\right)\frac{\partial U}{\partial t} = \nabla\cdot\left(\tilde{D}\frac{1-\phi}{2}\nabla U - \vec{j}_a\right) + \frac{1}{2}[1+(1-k)U]\frac{\partial\phi}{\partial t} \tag{7}$$

$$\frac{\partial\theta}{\partial t} = \tilde{\alpha}\nabla^2\theta + \frac{1}{2}\frac{\partial\phi}{\partial t} \tag{8}$$

where $A(\psi) = 1 + \varepsilon\cos[\omega(\psi - \psi_0)]$ is an anisotropy function and ε denotes the amplitude of the anisotropy strength, ω is the symmetry pattern coefficient ($\omega = 4$, 6 represents a four and six fold crystal structure, respectively), and ψ indicates the orientation of the dendrite i.e. the angle between the main arm and the x axis ($\psi = \arctan(\phi_y/\phi_x)$). Le is the Lewis number, and M is a scaled slope of the alloy liquidus (assuming a straight line) in the binary alloy phase diagram i.e. $M = |m|(1-k)/(L/cp)$ where m is the actual slope of the liquidus taken from the phase diagram.

$$U = \frac{\frac{2c/c_\infty}{1+k-(1-k)\phi} - 1}{1-k} \tag{9}$$

and

$$\theta = \frac{T - T_M - mc_\infty}{L/c_p} \tag{10}$$

are the dimensionless solute and temperature respectively; T_M is the melting temperature of pure element A, and λ is a scaled parameter with its reciprocal measuring the height of the dimensionless energy barrier (H) of the double well potential [8,15]:

$$\lambda = \frac{15L^2}{16Hc_pT_M} = \frac{15RT_M(1-k)L}{16v_0Hc_p|m|} \tag{11}$$

where v_0 is molar volume.

In the phase field equation, modeling of the alloy system energy fluctuation is commonly achieved by introducing a noise term to each of Eqs. (6)–(8) [14]. In this study, a single noise term $\dot{\xi}$ was introduced in Eq. (6) and was implemented using a Gaussian random number with specific amplitude, an approach similar to that used by Mullis [19] for the simulation of the solidification of pure Ni. As discussed later, a single noise term embedded in the source term in Eq. (6) not only introduces fluctuations in the temperature field, but also in the solute field via the phase field variable ϕ.

In Eqs. (6)–(8), the time and length are scaled by the relaxation time τ_0 and diffuse interface width W_0, respectively, and $\tilde{D} = (D\cdot\tau_0)/W_0^2$ and $\tilde{\alpha} = Le\cdot\tilde{D}$ are dimensionless solute and thermal diffusivities. The physical time and length can be recovered via the following equations [15]:

$$W_0 = d_0\frac{\lambda}{a_1} \tag{12}$$

$$\tau_0 = \frac{d_0^2}{D}\frac{a_2}{a_1}\lambda^3 \tag{13}$$

where d_0 is the thermal capillary length, i.e. $d_0 = \Gamma/(L/c_p)$, Γ is the Gibbs–Thomson coefficient and D is the solute diffusivity. According to thin interface limit analysis, $a_1 = 0.8839$ and $a_2 = 0.6267$ [15].

2.2. Boundary conditions and energy source terms

A Neumann boundary with zero fluxes is applied to Eqs. (6)–(8) and therefore there are no heat and solute flux across the boundaries. In order to simulate cooling (or heating during melting), an "artificial" energy source term $\dot{q}$ is introduced in Eq. (8), acting as either a heat source ($\dot{q} < 0$) or sink ($\dot{q} > 0$) to numerically mimic the effect of the heat flux into/out of the computational domain:

$$\frac{\partial \theta}{\partial t} = \tilde{\alpha}\nabla^2\theta + \frac{1}{2}\frac{\partial \phi}{\partial t} - \dot{q} \tag{14}$$

In the approach presented here, there is no nucleation barrier to the transformation of liquid to solid, and growth of solid occurs on pre-existing spherical solid "seed" crystals introduced into the computational domain. In this way, the simulation of many dendrites with latent heat release under a heating/cooling condition can be performed, and the benchmark tests described later show that this approach gives a good qualitative approximation to real solidification conditions.

2.3. The orientation field

In the case of multi-dendrite growth with randomly oriented crystals, the orientation of the dendrite is usually addressed by applying extra equations describing the orientation field. In these studies, the free energy equation is thus modified by including a term related to the dendrite orientation within a fixed reference frame. However, there are concerns that this approach is unphysical because the free energy should not be dependent on the reference frame of the system. To address this concern, extensive work has been conducted towards achieving the invariant formulation of the free energy under rotation of the reference frame [5,20–22]. Because the focus of our study is on the dendrite growth and morphology evolution, a simpler approach for dendrite orientation is employed comprising two steps. First, the orientation of each of the dendrites is randomly specified with a pre-existing solid seed introduced in the computational domain. Second, as the dendrite grows very small volumes of liquid are added/transformed to solid, the crystal orientation for this new increment in solid is assumed the same as that of the local crystal orientation of the dendrite. In other words, the dendrite grows by transformation of nearby liquid into solid of the same orientation, which is physically similar to the real case and consistent with numerical approaches used in these sharp interface methods [23,24].

3. Numerical methods

3.1. Space and time discretization

Eqs. (6)–(8) were discretized on a two dimensional (2D) Cartesian square grid of equal spacing Δx using a finite difference (FD) scheme. For the first-order derivative of a variable, i.e. $\partial\phi/\partial x$ or $\partial\phi/\partial y$, a simple central discretization was used, and for the Laplace operator, a compact 2nd order nine point stencil was employed [25]:

$$\nabla^2 u = \frac{1}{3\Delta x^2}\left(\sum_{i,j} u^n - 8u^n_{i,j}\right) \tag{15}$$

$$\sum_{i,j} u^n = u^n_{i-1,j} + u^n_{i+1,j} + u^n_{i,j-1} + u^n_{i,j+1} + u^n_{i-1,j-1} + u^n_{i-1,j+1} + u^n_{i+1,j-1} + u^n_{i+1,j+1} \tag{16}$$

where the superscript n indicates time step. In order to maintain 2nd order precision for both space and time during discretization, a 2nd order backward difference formulae (BDF2) method was employed for time discretization that retrieves information from two previous time steps [17]:

$$\frac{\partial u}{\partial t} = \frac{u^n + a_0 \times u^{n-1} + a_1 \times u^{n-2}}{b \times \Delta t} \tag{17}$$

where $a_0 = -4/3$, $a_1 = 1/3$, and $b = 2/3$. Rosam et al. [16,17] proved that the BDF2 method for time discretization is more stable than that of the 1st order Euler method. Accordingly, the final forms of the discretized Eqs. (6)–(8) are:

$$G_\phi(\phi^{n+1}) = P_{S,\phi} + p_{0,\phi}\sum_{i,j}\phi^{n+1} + p_{x,\phi}\left(\phi^{n+1}_{i+1,j} - \phi^{n+1}_{i-1,j}\right) + p_{y,\phi}\left(\phi^{n+1}_{i,j+1} - \phi^{n+1}_{i,j-1}\right) = R_\phi \tag{18}$$

$$G_U(U^{n+1}) = p_{S,U}U^{n+1}_{i,j} + p_{0,U}\sum_{i,j}U^{n+1} + p_{x,U}\left(U^{n+1}_{i+1,j} - U^{n+1}_{i-1,j}\right) + p_{y,U}\left(U^{n+1}_{i,j+1} - U^{n+1}_{i,j+1}\right) = R_U \tag{19}$$

$$G_\theta(\theta^{n+1}) = p_{S,\theta}\theta^{n+1}_{i,j} + p_{0,\theta}\sum_{i,j}\theta^{n+1} = R_\theta \tag{20}$$

A detailed description of the related terms in Eqs. (18)–(20) is given in Appendix A.

3.2. Full approximation storage (FAS) multigrid method

To solve the nonlinear PF Eqs. (18)–(20), the FAS multigrid method that was first introduced by Brandt [26,27] is employed. The advantage of using the FAS method is that global linearization commonly needed in the Newton process (used for nonlinear problems) is avoided and the only linearization required is a local one in order to define the relaxation procedure e.g. Gauss–Seidel–Newton procedure. To solve a function $A_k(u_k) = f_k$ the FAS multigrid algorithm includes the following steps [25]:

(1) If $k = 1$, solve $A_k(u_k) = f_k$ directly.
(2) Pre-smoothing steps on the fine grid: $u_k^l = S(u_k^{l-1}, f_k), l = 1, \ldots, \nu_1$.
(3) Computation of the residual: $r_k = f_k - A_k u_k^{\nu_1}$.
(4) Restriction of the residual and solution: $r_{k-1} = R_k^{k-1} r_k, \hat{u}_{k-1} = \hat{R}_k^{k-1} u_k^{\nu_1}$.
(5) Set $f_{k-1} = r_{k-1} + A_{k-1}(\hat{u}_{k-1})$.
(6) Call γ times the FAS scheme to solve $A_{k-1}(u_{k-1}) = f_{k-1}$.
(7) Coarse-grid correction: $u_k^{\nu_1+1} = u_k^{\nu_1} + I_{k-1}^k(u_{k-1} - \hat{u}_{k-1})$.
(8) Post-smoothing steps on the fine grid: $u_k^l = S(u_k^{l-1}, f_k), l = \nu_1 + 2, \ldots, \nu_1 + \nu_2 + 1$.

For restriction of the residual, as indicated above by operator R, the full weight (FW) operator is adopted, while a direct injection is used for the restriction of the solution u, as indicated by $\hat{R}$. A bilinear operator is used for the interpolation, as indicated by I. Further details of the transfer operators for restriction and interpolation can be found in [25]. For relaxation, as denoted by operator S, a Gauss–Seidel line smoother is employed to achieve a higher degree of parallelism. This smoother is composed of two half-steps: during the first half step, the odd lines (or columns) of the grid points are smoothed, after which the even lines (or columns) are smoothed using the updated values of the neighbor grid points. In this way, each half-step of the relaxation is fully parallelized. The smoothing process is carried out using the Gauss–Seidel method:

$$u^{l+1} = u^l - \frac{G_u(u^l) - R'_u}{p^l_{S,u}} \tag{21}$$

where u denotes the target relaxation variable, i.e. ϕ, U, or θ. $G_u(u^l)$ is the target equation i.e. (18)–(20), and R'_u is the right hand side term during the smoothing procedure, which is always different from the equation right hand side terms R_u in Eqs. (18)–(20).

The relaxation for U and θ is straight-forward. However for the non-linear PF variable ϕ, a Newton–Gauss–Seidel method is employed, and $p_{S,\phi}$ is evaluated by the derivation of the PF source term $P_{S,\phi}$ using Eq. (29) (in Appendix A):

$$p^l_{S,\phi} = \left(\frac{1}{Le} + Mc_\infty[1 + (1-k)U]\right)A(\psi^l_{i,j})^2 - 8p^l_{0,\phi} - b\Delta t\left[1 - 3(\phi^l_{i,j})^2 + 4\lambda(\theta + Mc_\infty U)\left(1 + \dot{\xi}\right)\phi^l_{i,j}\left(1 - \left(\phi^l_{i,j}\right)^2\right)\right] \tag{22}$$

3.3. Parallel computing scheme

In general, parallel computing of discretized PDEs can be realized either by using a fast sequential solver with its components parallelized as efficiently as possible or by decomposing the given problem into a number of sub-problems or sub-domains and then solving them separately in a parallel manner [25]. Although global multigrid partitioning and domain decomposition are, conceptually, quite different, in practice certain variants of both approaches are similar or may even lead to the same algorithm.

In this study, a sequential FAS multigrid solver with parallelized multigrid components is employed. The crucial multigrid aspects with respect to parallelization are usually: (1) the smoothing procedure; (2) the transfer operations (restriction and interpolation); and (3) the residual determination procedure. To achieve a high degree of parallelism, two important issues need to be addressed: (1) grid partitioning of the computing domain; and (2) communication between adjacent processes. In addition, because the number of grids points is different at different grid levels, the communication "overhead" becomes worse towards and especially at the coarsest grid level and therefore special care needs to be taken.

3.3.1. Grid partitioning

The purpose of grid partitioning of a discretized domain is to distribute the entire computing overhead to multiple processors, and trying to achieve the same computing performance for each processor. A rectangular domain assumingly consisting of $M \times N$ (where $M = 2^m$, and $N = 2^n$) cells can be partitioned either into sub-squares with each containing approximately $M_P \times N_P$ cells or strips with each having $M \times N_P$ (or $M_P \times N$) cells. The latter can be programmed using a simpler communication mechanism and a full line smoother, and therefore is adopted here to further divide the square domain into $N_{proc} = 2^p$ subdomains (processes) as shown in Fig. 1a. The number of cells in each of the N_{proc} sub-domains is identical and equal to $2^m \times 2^{n-p}$. In order to achieve the communication between neighbor processes efficiently, as discussed in the following section, two overlap lines of cells were added at the left and right sides of the sub-domain respectively, as shown in Fig. 1a. Consequently, for each sub-domain, and corresponding to each process, there are a total of $2^m \times 2^{n-p+1}$ cells. One of

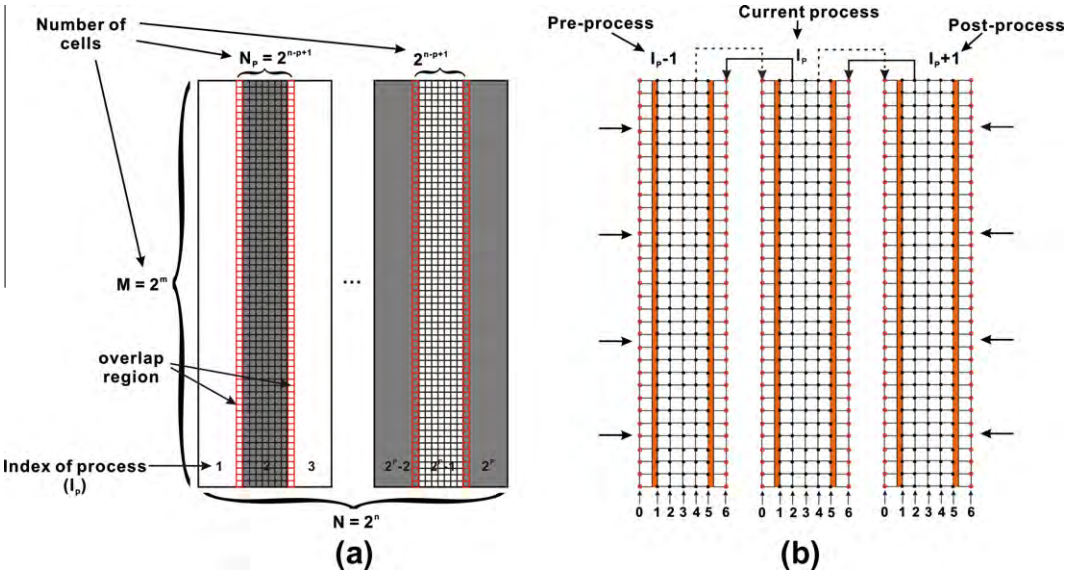

Fig. 1. Schematics show (a) grid partitioning of the computational domain and (b) communication mechanism between adjacent processes for parallel computing.

the significant advantages of distributing the computational load to 2^p processes is that in terms of grid points, the structure of the multigrid components of each process stays the same and the computational effort is practically divided into 2^p parts, which are independent of one another, achieving the so called single program multiple data (SPMD) [28] architecture.

3.3.2. Communication between processes

For each process denoted by I_P in Fig. 1b, communication occurs only locally with nearest neighbors i.e. $I_P - 1$ and $I_P + 1$. In order to reduce communication overhead, a thin overlap area of width 1 is applied to copy the data located in its neighbor sub grids (Fig. 1b). In this way, during communication, each process sends all data belonging to one side of the overlap area of a neighbor sub-grid collectively i.e. in one long message, to its corresponding neighbor process and receives the data corresponding to its own overlap area for that neighbor in a similar efficient fashion.

A careful examination of the multigrid procedure reveals that data communication only needs to be carried out during the smoothing process and the transfer operations. For smoothing, during the first half step of relaxation sweeps, the odd columns of data (as indicated in Fig. 1b by column number 1, 3, $N_P - 2$, etc.) can be updated according to Eq. (21), and no communication is needed. The second half step of relaxation can then be performed on the even columns of data (as indicated in Fig. 1b by column number 2, 4, $N_P - 3$, etc.) using the newly calculated values from the first half step. After the whole relaxation procedure is finished, the newly updated data in the first even column, i.e. column 2 and the last even column, i.e. column $N_p - 3$ (as shown in Fig. 1b) needs to be sent from the current process I_P to its pre-process $I_P - 1$ (as indicated in the upper part of Fig. 1b by the solid arrow line), and post-process $I_P + 1$ (as indicated in the upper part of Fig. 1b by the dash arrow line), respectively. The receiving operations are performed accordingly and the new data is stored in the overlapped region. Similar communication occurs during restriction and interpolation.

3.3.3. Coarsest level procedure

As discussed above special care has to be taken at the coarsest level of the parallel multigrid algorithm. Taking the "V" cycle for example, when the multigrid reaches the coarsest level, there must be at least one cell (or two columns) of data left for each process. If the process number is $N_{proc} = 2^p$, the domain of the target problem at coarsest level must then include at least $2^p \times 2^p$ cells. The multigrid algorithm can be preceded either by further shrinking the dimension of the domain through more communication or sending the related data from all the processes to another single process (or all processes) and then retrieving the solution there. In practice, the second way is always more efficient (which is also adopted in this study) because the computational costs (operations and time) are always trivial for a problem of this scale.

After the data, including parameter and solution matrices, have been transferred to a single process (such as the root process), a fast solver such as a further multigrid (employed here), LU decomposition, or preconditioning CG or GMRES can be employed for solving the linear equations e.g. solute and temperature equations. For the PF equation, i.e. Eq. (18), a Newton-Line-Searching method was employed due to its global convergence. Details of this method can be found in [18].

3.3.4. Analysis of the parallel performance

Two principles are used here to judge the performance of the parallel algorithm. The first is speedup (denoted here as S_p), which is defined as the ratio of the time taken to solve a problem on a single process element to the time required to solve the same problem on a parallel computer with N_{proc} identical processing elements. The second is parallel efficiency (denoted here as E), which is a measure of the fraction of time for which a processing element is usefully employed; it is defined as the ratio of speedup to the number of processing elements [28], i.e. $E = S_p/N_{proc}$. An ideal parallel system should be $S_p = N_{proc}$ and $E = 1$, but due to the inevitable communication overhead the actual speed-up will be less than N_{proc} and the efficiency will be a value between 0 and 1.

Firstly a two dimensional square domain (which is commonly used for the PF simulation) is taken for an example, assuming the domain has a size of $N \times N$ cells (equivalent to $(N+1) \times (N+1)$ grid points). For simplicity, the number of arithmetic operations (including smoothing, restriction, interpolation, and residual estimation, etc) needed for a single grid point is assumed to be the same at all grid levels and can be characterized by a fixed computing time t_{comp}. If the infinite Taylor series is used and taking the "V" cycle for instance, the required total computing time for a sequential multigrid algorithm is $t_{seq} = (N+1)^2 \times t_{comp} \times (1 + 1/4 + 1/16 + \cdots) = 4/3 \times (N+1)^2 \times t_{comp}$. For a parallel system, the communication time needed for a message of length L_m is $t_{comm} = \alpha_c + \beta_c \times L_m$ where α_c is the start-up time (or latency) which has to be spent whenever a message is sent, and $1/\beta_c$ is the bandwidth of the respective communication channel [25]. If $N = 2^n$ and the coarsest level has $2^P \times 2^P$ cells, then for the multigrid algorithm there are totally $n - p$ grid levels.

Considering the same problem used for the sequential algorithm calculated using N_{proc} identical processes and assuming the total elapsed time is t_{par}, and for the current proposed model:

$$t_{par} = \tilde{t}_{cal-f} + \tilde{t}_{cal-c} + \tilde{t}_{cm-f} + \tilde{t}_{cm-c} \tag{23}$$

where

$$\tilde{t}_{cal-f} = \frac{4}{3} \times \left(\frac{N}{N_{proc}} + 1 \right) \times (N+1) \times t_{comp} \tag{24a}$$

$$\tilde{t}_{cal-c} = \frac{4}{3} \times (N_{proc} + 1)^2 \times t_{comp} \tag{24b}$$

$$\tilde{t}_{cm-f} = \mu_1 [(n-p-1)\alpha_c + 2(N+1)\beta_c] \tag{24c}$$

$$\tilde{t}_{cm-c} = \mu_2 [\alpha_c + \gamma(N_{proc} + 1) \times \beta_c] \tag{24d}$$

are the elapsed time for computation before reaching the coarsest grid level, time for computation at the coarsest level, time for communication before reaching the coarsest grid level, and the time for communication at the coarsest level, respectively. γ, μ_1, and μ_2 are constants. The parallel speed-up is then:

$$S_p^{-1} = \frac{t_{seq}}{t_{par}} = \frac{N/N_{proc} + 1}{N+1} + \left(\frac{N_{proc} + 1}{N+1} \right)^2 + \Gamma_{comm} \tag{25}$$

where

$$\Gamma_{comm} = \frac{3}{4} \left(\frac{\mu_1(n-p-1) + \mu_2}{(N+1)^2} \frac{\alpha_c}{t_{comp}} + \left(\frac{2\mu_1}{N+1} + \frac{\mu_2 \gamma (N_{proc} + 1)}{(N+1)^2} \right) \frac{\beta_c}{t_{comp}} \right) \tag{26}$$

The ideal parallel speed-up (disregarding the communication terms) is:

$$S_p^{-1} = \frac{N/N_{proc} + 1}{N+1} + \left(\frac{N_{proc} + 1}{N+1} \right)^2 \tag{27}$$

The ideal parallel efficiency is then:

$$E^{-1} = \frac{N + N_{proc}}{N+1} + \left(\frac{N_{proc} + 1}{N+1} \right)^2 N_{proc} \tag{28}$$

From Eq. (28), the parallel efficiency can be improved by either increasing the size of the problem domain N or by reducing the number of the processes applied during the parallel computation. Therefore, the current parallel algorithm is suitable for a large size problem with a comparatively small number of processes. If the communication effect is considered, i.e. by taking Γ_{comm} into account, the overall efficiency reduction will be further dependent on the structure and efficiency of the parallel system employed.

The parallel code was developed on the 640-core (1.9 TB RAM) supercomputing cluster named as SAL housed at the Oxford Supercomputing Center using C++ language and two message passing libraries namely Message Passing Interface (MPI) [29] and Open Multi-Processing (OpenMP) [30]. In the program, OpenMP was used in conjunction with MPI to provide a second level of parallelism, resulting in an overall so-called hybrid program structure. The hybrid structure of the code was

employed to allow implementation on a supercomputing system comprising a combination of distributed and shared memory systems [28,31]. The grid partitioning, job dispatching and data communication were handled in the first level of parallelism i.e. controlled by the MPI processes on the distributed memory level. For each MPI process, computing tasks such as matrix and vector related operations were then handled by the shared memory system through the second level parallelism i.e. OpenMP.

4. Model validation and parameter study

Validation for the PF model was carried out by comparing the simulated solute concentration at a dendrite tip with that calculated according to the well-known velocity-dependent Gibbs–Thomson condition [15] $U_i = [-d_0(1-15\varepsilon)/\rho - \theta_i]/Mc_\infty$ where U_i and θ_i are the dimensionless solute concentration and temperature at the dendrite tip. The interface temperature θ_i is determined at $\phi = 0.9$ [15]. The dendrite tip radius of curvature ρ is determined using $\rho = (\partial\phi/\partial x)/(\partial^2\phi/\partial y^2)$ [15,32]. It is worth stressing that dendrite tip velocity, radius of curvature, Peclet numbers etc simulated using the PF model can also be compared with classical dendrite growth theories e.g. LGK (according to Lipton, Glicksman and Kurz) and LKT (according to Lipton, Kurz and Trivedi) [33]. In this study, we focus on the evolution of the dendrite morphology, testing the validity of the PF model, and the efficiency of the numerical schemes developed.

4.1. Isothermal dendrite growth

Firstly isothermal dendrite growth was simulated in a square domain containing 1024×1024 cells with $dx = dy = 0.8$, and a solid nucleus with a radius $R_0 = 28d_0$ seeded in the liquid alloy at the bottom-left corner with symmetrical boundary conditions applied at the bottom and left boundaries. Isothermal dendrite growth is initiated by imposing a solutal undercooling of $\Omega = (c_l^0 - c_\infty)/[(1-k)c_l^0]$ to the system and $Mc_\infty = 1-(1-k)\Omega$, and $U_0 = \Omega/[1-(1-k)\Omega]$ where U_0 is the fixed value of U in the left-hand side of Eq. (6). The values of the system parameters adopted in this study are the same as those in [8,15], i.e. $\Omega = 0.55$, $\varepsilon = 0.02$, $k = 0.15$, and the scaling parameter $\lambda = 3.1914$.

The PF equations were solved using a simple adaptive time marching scheme similar to that used in [17]. A method based on proportional integral (PI) control theory [18] was used to control the marching of next time step in order to avoid unstable cyclic adaptivity: $\Delta t_{n+1} = S \cdot \Delta t_n \cdot err_n^{-\beta} \cdot err_{n-1}^{\gamma}$ where $S < 1$ is a safety factor (0.8 was used here), and err_n and err_{n-1} are the errors from the current and last time steps. According to [18] β and γ were chosen to be 0.133 and 0.233 respectively.

Fig. 2a shows the evolution of the calculated truncation error of ϕ and U as a function of the number of simulation steps (against the left y axis). The truncation error of U decreases abruptly from its initial value (about 6.6×10^{-4}) and then reaches a much lower value (about 10^{-5}) within the first 1000 time steps. On the other hand, the truncation error of ϕ increases gradually at first and then reaches an approximately constant value until the end of the simulation. Fig. 2a also shows the evolution of the adaptive time step as a function of the number of simulation steps (against the right y axis). The time step increases and reaches a maximum (about 0.36) and then decreases slightly, after which it rises again to the end of the simulation. This qualitative shape of this history for the adaptive time step has a similar evolution shape to that reported by Rosam et al. [16].

By applying the implicit multigrid algorithm, a comparatively large time step was used successfully during the simulation. As shown in Fig. 2a, at a relatively small truncation error, i.e. $\sim 7.0 \times 10^{-4}$, the final time step can be as large as 0.5,

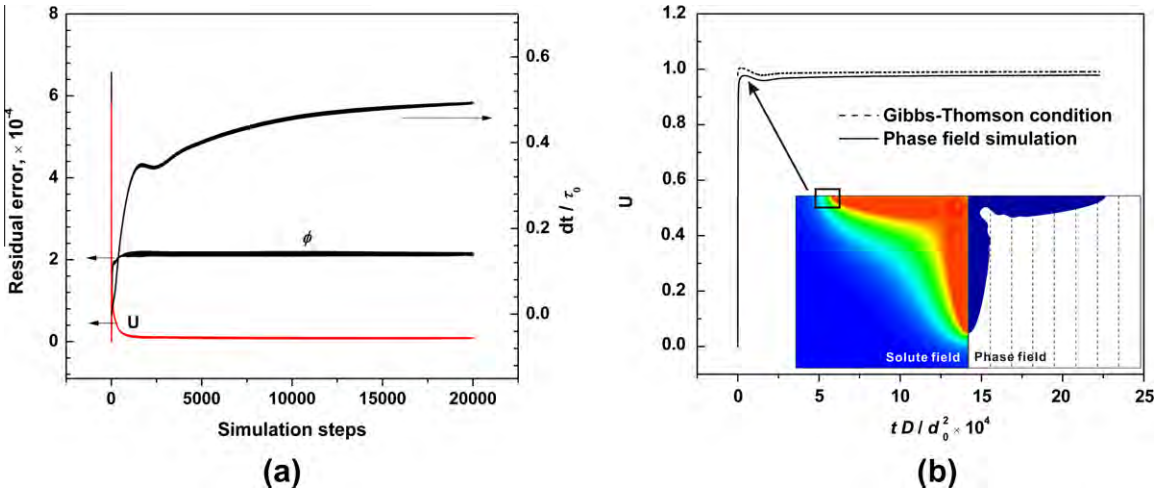

Fig. 2. (a) The evolution of the adaptive time step and the calculated truncation errors of ϕ, U as a function of simulation steps, and (b) comparison between U calculated using PF model and the Gibbs–Thomson condition.

~6.25 times larger than the limit used in the explicit algorithm [17]. By employing the adaptive timing and parallel computing schemes, the simulation was finished within 1.6 h using an eight core computing node in the SAL cluster. It is worth stressing that the efficiency of the method is not fully demonstrated until the temperature equation is also considered as discussed later.

Fig. 2b shows a comparison of U (at the tip of the evolving dendrite) obtained using the PF model and the same according to the Gibbs–Thomson analysis under the same condition (there is also an insert in Fig. 2b showing the contour maps of the solute and phase fields). The x axis tD/d_0^2 is dimensionless time where D and d_0 are solute diffusivity and capillary length, respectively. As seen from Fig. 2b, there is good agreement for the solute concentration at the dendrite tip with only a few percent difference at all stages of the evolving dendrite.

One of the parameters for the current phase field model, namely the scaling parameter λ is usually chosen arbitrarily in literature. As far as accuracy is concerned, it is recommended that λ should be chosen to be as small as possible (because $W_0 \propto \lambda$ according to Eq. (12)). However, from a computational point of view a decrease of λ usually delays the evolving rate of a growing dendrite, leading to elongating of the computing hours, which is not favorable in practice. Here, tests were also performed concerning the effect of λ on the dendrite morphology.

Fig. 3a–c show the calculated dendrite (one quarter) morphologies (retrieved at $\phi = 0$) with different-sized initial seeds, i.e. $R_0 = 15d_0$, $28d_0$, and $44d_0$, and all the profiles are retrieved when the dendrite tips reach the same position ($x_{tip} = 866.54\, d_0$ away from the seed center, corresponding to a dimensionless length of 300 at $\lambda = 3.1914$). Each profile in Fig. 3a–c shows the effect of a different scaling parameter, varied from 1.5957 to 12.7656. In Fig. 3a–c, the main arms of the dendrite propagate along x and y axis, with both lengths scaled by d_0.

The influence of λ on the dendrite profiles is amplified as the seed size decreases. The most prominent effect is the change in the morphology of the dendrite between the primary x and y dendrite arms, for example in Fig. 3a, this region changes from concave to convex curvature as λ is increased from 1.5957 to 12.7656. This effect is much less pronounced in Fig. 3c for larger initial seed crystals dimensions.

4.2. Convergence evaluation

Fig. 4a shows the dendrite profiles ($\phi = 0$) calculated using different spatial discretizations ($dx = 0.2$, 0.4, 0.8, 1.2, and 1.6) for an initial seed radius of $15d_0$. All the profiles were retrieved when the dendrite tips reach the same position as that shown in Fig. 3. In Fig. 4a, the local area near the seed center is enlarged and shown in detail in the zoom-in insert. The profile converges as the spatial step decreases and a clear deviation of the profile can be observed when the spatial step $dx > 1$. But such deviation only happens locally near the location of the initial seed, and as the seed grows, it gradually vanishes along the dendrite arms. Using this result, the spatial time step dx is subsequently chosen as 0.8 in the following simulations to evaluate both accuracy and efficiency. Fig. 4b shows the calculated dendrite profiles using five constant time steps, 0.04, 0.08, 0.32, 0.64 and 1.60 (the corresponding time step limit for the explicit method is approximately 0.08) and one adaptive time scheme. Notice that relatively "large" values i.e. $dt > 0.32$ were used to test the upper limit for maintaining the stability of the algorithm. As shown in Fig. 4b, excellent convergence can be achieved even when the time step is increased to 0.64, and the profiles are almost overlapping for time steps of 0.04, 0.08, 0.32 and 0.64. There is deviation of shape when the time step is further increased i.e. from 0.64 to 1.60. The most significant difference is that the concave structure in between the two primary arms tends to stretch further towards the seed centre as the time step is increased. Through this type of numerical tests, it was determined that for this isothermal case, the stability limit of the current algorithm was slightly higher than $dt = 1.6$ (approximately 20 times larger than the limit required for an explicit algorithm). To keep the time step in a "safe"

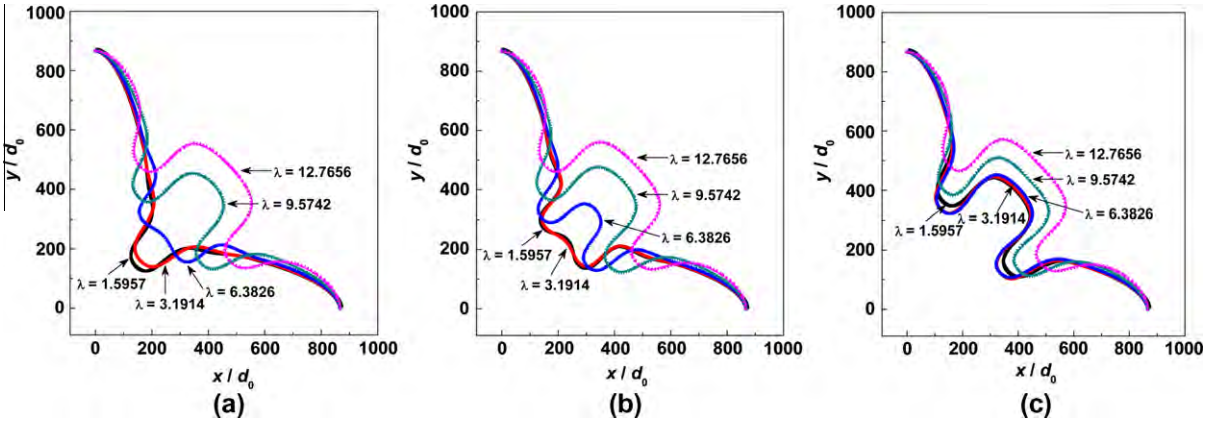

Fig. 3. Calculated dendrite morphologies (the ϕ profile) under different scaling parameters $\lambda = 1.5957$, 3.1914, 6.3826, 9.5742, and 12.7656 at different initial seed radius, i.e. R_0: (a) $R_0 = 15\ d_0$, (b) $R_0 = 28\ d_0$, and (c) $R_0 = 44\ d_0$.

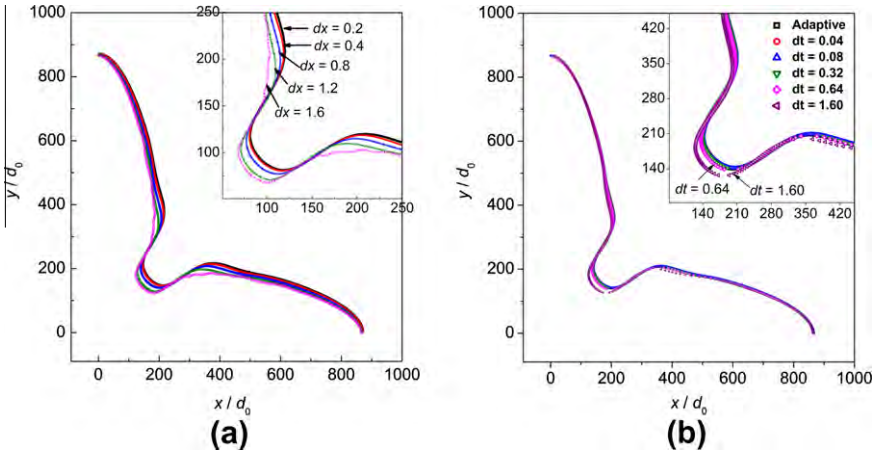

Fig. 4. Convergence evaluations for dendrite morphologies (the ϕ profile) in an isothermal condition at different (a) spatial and (b) time steps.

regime, the incremental dendrite growth should be restricted. Dendrite growth is mostly controlled by the diffusion of the solute (because heat transfers 3–4 orders of magnitude faster than solute), and so $dt < dx^2/4D$. In this respect, the time step for the following cases presented below in which solute was coupled with temperature was set at $dt = 0.8 \times dx^2/4D$, which is 8000 times larger than the time step for coupled conditions in the more commonly used explicit algorithm.

4.3. Parallel efficiency and scalability

Applying the same isothermal case, Fig. 5 shows the related testing results concerning the parallel efficiency and scalability of the method. All calculations were performed to a duration of 100 dt. The problem size was increased progressively from $\#\Omega = 512 \times 512$ to $\#\Omega = 8192 \times 8192$, with each simulation increased by a factor of 2. For each case, the process number used for calculation i.e. N_{proc} was changed from 1 to 64 (also each time increased by a factor of 2). Calculations were stopped when there was more than 10% deviation of the calculation time to that predicted by the linear scalability theory was observed.

Fig. 5a shows the calculation time against the number of processes. The symbols indicate the simulation results while the lines denote the ideal trend for the calculation time as a function of the process number (simply calculated by the calculation time using one process divided by N_{proc}). For the smallest case, i.e. $\#\Omega = 512 \times 512$ the calculation time decreases linearly as the process number increases until $N_{proc} = 8$ at which point the simulation time is significantly larger than the "ideal" calculation time. This situation is greatly improved as the problem size, i.e. $\#\Omega$ is increased. As seen from Fig. 5a, the same point is reached at $N_{proc} = 16$ when $\#\Omega = 1024 \times 512$, $N_{proc} = 32$ when $\#\Omega = 2048 \times 1024$, and $N_{proc} = 64$ when $\#\Omega = 4096 \times 4096$. This is expected, because the bigger the computing domain is, the more the effort will be used for calculation rather than com-

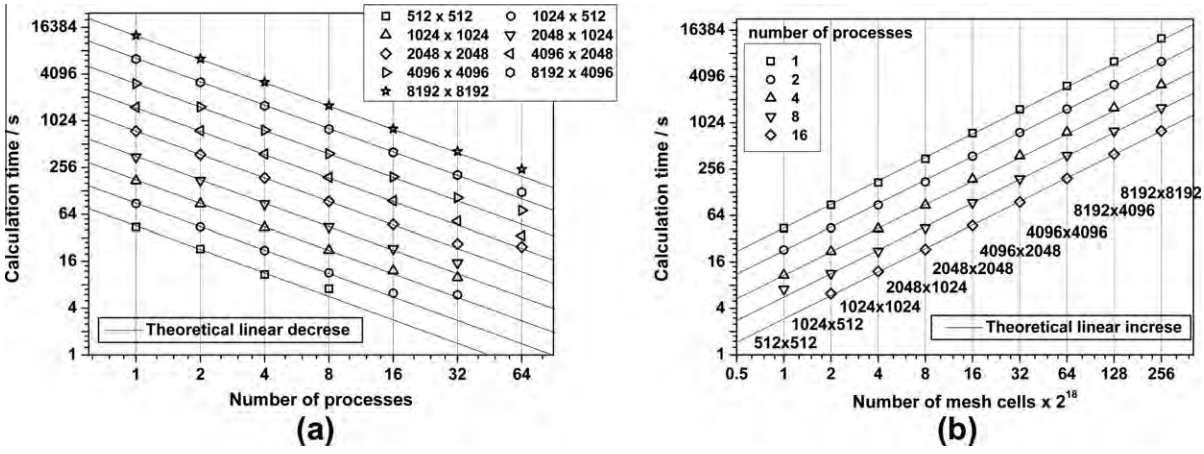

Fig. 5. Parallel efficiency and scalability tests using the proposed method for the isothermal dendrite growth case with the calculation time versus (a) the number of the processes under different sizes of computing domain and (b) the number of mesh cells under different numbers of the parallel processes.

munication, and consequently the parallel efficiency will be improved. However, such improvement cannot be limitlessly increased because the data length during communication will also be increased.

Fig. 5b shows the calculation time against the number of the mesh cells. Here, the "ideal" trend lines were determined by the calculation time under $\#\Omega = 8192 \times 8192 = 2^{26}$ multiplied by $\#\Omega/2^{26}$. The symbol type indicates the process number during the simulation. The parallel scalability of the method is shown to be extremely high. The calculation time increases linearly as the domain size increases from $\#\Omega = 512 \times 512$ to $\#\Omega = 8192 \times 8192$. This situation applies almost ideally when the process number, i.e. N_{proc} is 1, 2 and 4. For the case when N_{proc} reaches 8, there is a deviation of the simulation time (from the linear trend) when $\#\Omega = 512 \times 512$. This is also as expected because using too many processes to simulate a "small" case does not extract the benefit of this particular method.

5. Dendrite evolution in the coupled thermal-solutal condition

For testing the ability of the proposed parallel multigrid approach, a multi-dendrite simulation was performed on a relative large, square domain of 8192×8192 cells by applying 64 processes. An Al-4wt%Cu alloy was simulated using param-

Table 1
Physical properties of the Al–Cu alloy used during simulation [11,33].

Parameter/Physical property	Value
Latent heat of melting, L (J kg^{-1})	3.82×10^5
Specific heat, c_p (J kg^{-1} K^{-1})	1.05×10^3
Density, ρ (kg m^{-3})	2.61×10^3
Liquid solute diffusivity, D (m^2 s^{-1})	3.0×10^{-9}
Lewis number, Le	10^4
Liquidus slope, m (K/wt%)	-2.6
Initial alloy composition, c_∞ (wt%)	4
Equilibrium partition coefficient, k	0.15
Anisotropy strength, ε	0.03
Thermal capillary length, d_0 (m)	6.6×10^{-10}
Gibbs–Thomson coefficient, Γ (K m)	2.4×10^{-7}

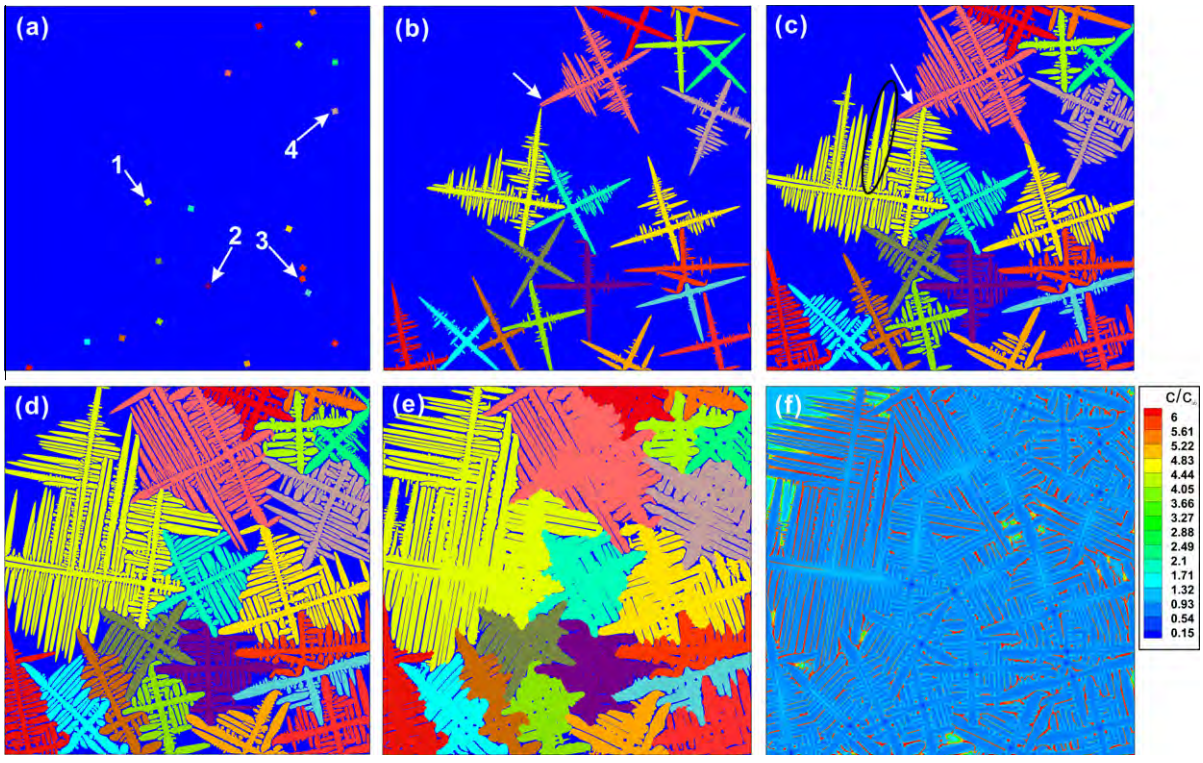

Fig. 6. Simulated dendrite growth in a coupled thermal-solute condition in a domain of 8192×8192 mesh grids with 20 initial random seeds at different time, i.e. tD/d_0^2 = (a) 8191.7, (b) 24575.1, (c) 40958.5, (d) 65533.6, and (e) 81917. The solute concentration for (e) measured in c/c_∞ is shown in (f).

eters shown in Table 1. Other important parameters were: (1) the noise amplitude $\dot{\xi} = 1 \times 10^{-3}$; (2) a constant time step $dt = 0.8 \times (dx)^2/(4 \times \tilde{D})$, 8000 times larger than the restrict limit employed in explicit algorithms because the adaptive time scheme is no longer suitable when the noise term is introduced; (3) the spatial step $dx = 0.8$; and (4) the scaling parameter $\lambda = 5$ was chosen in order to facilitate the development of dendrites. By applying these parameters the final size of the computing domain is 37072 $d_0 \times 37072$ d_0, which represents a physical domain of 25 × 25 μm. Another important parameter is the artificial energy source (or drain) term $\dot{q}$ in Eq. (14) which was set at 1.3×10^{-3} after a number of numerical tests were undertaken to select conditions that mimic cooling of the system and the evolution of realistic dendrite morphology, such as high-order dendrite arm branching and solute segregation. The physical formulation of the heat source (or drain) term can be restored by utilizing the real time and length scales into $\dot{q}_{real} = \frac{\rho L}{\tau_0} \times \dot{q}$, from which the actual value of the heat drain term $\dot{q}_{real} \sim 1.0 \times 10^{-4}$ Wμm^{-3}. The computing domain was initialized with a uniform undercooling of $\Delta\theta = -0.05$ (~18 K) and the liquid alloy domain was then filled with 20 randomly positioned and oriented solid seeds of $R_0 = 25d_0$.

5.1. Dendrite evolution, impingement, solute segregation and heat transfer

Fig. 6a–e show the morphology evolution of the 20 dendrites in the period from $tD/d_0^2 \approx 8191.7$ to $tD/d_0^2 \approx 81917$ during the solidification calculated using 64 processes and taking ~18 h. Colors are used in Fig. 6a–e to distinguish different dendrites. At first, all seeds grow independently and gradually evolve into a four-fold morphology; then as each primary arm marches into the surrounding undercooled melt, its secondary dendrite arms gradually develop and grow. After the main arms from different dendrites meet and impinge, the growth of the main arms is restricted, while the higher order branches gradually grow faster and some of these arms may even overtake the main arm as indicated by the circle in Fig 6c. The simulation also shows that some dendrite tips split during growth (marked by the arrow in Fig. 6c). This dendrite tip splitting phenomenon has been extensively observed in experiment and discussed in [34], however, this is the first time such event is revealed by a PF simulation in the thermal-solutal coupled condition. Fig. 6f shows the solute distribution corresponding to the dendrite morphology shown in Fig. 6e. As expected, during solidification, strong solute segregation exists among the high-order dendrite arms especially at their roots.

Fig. 7a–e show the temperature distributions corresponding to Fig. 6a–e. Fig. 7f shows the thermal histories at the locations marked by 1, 2, 3 and 4 in Fig. 6a. Due to the release of latent heat during solidification, the temperature is always

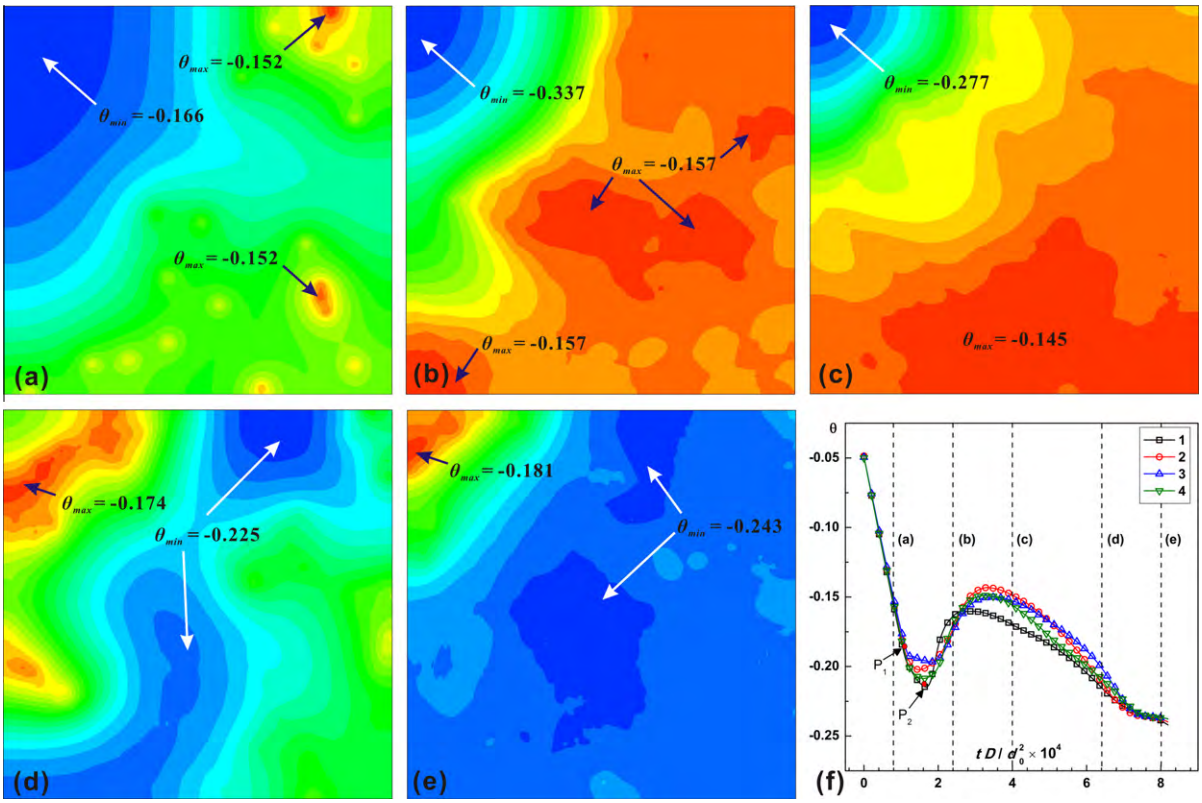

Fig. 7. Temperature contour maps at tD/d_0^2 = (a) 8191.7, (b) 24575.1, (c) 40958.5, (d) 65533.6, and (e) 81917 corresponding to the dendrite morphologies shown in Fig. 6. The thermal histories at position 1–4 indicated in Fig. 6a are shown in (f).

higher at the growing interface of the dendrite, especially in locations involving the growth of higher order arms. The different stages corresponding to Fig. 7a–e are also indicated in Fig. 7f by (a) to (e). The overall evolution of temperature at different locations is broadly similar: the temperature starts from an initial value ($\theta = -0.05$) and reduces quickly until it reaches a minimum which is followed by a gradual increase (as latent heat is evolved more quickly), and then drops slowly after reaching a maximum. The turning point in the temperature profile, as indicated in Fig. 7f by P_1, denotes the moment when the secondary branches start growing out from the main arms since the heat release associated with large solid/liquid area associated with secondary arms decreases the resulting cooling rate of system. The local minimum, as indicated in Fig. 7f by P_2 (corresponding to $tD/d_0^2 \approx 16{,}000$) indicates the moment when a local equilibrium between the cooling and the heating (resulting from the release of heat) is reached.

5.2. Effect of growth parameters

Simulations were performed for the different growth parameters listed in Table 2 (Case 1 is shown in Fig. 6), and investigation focused on the magnitude of the cooling term $\dot{q}$ (Case 2), the scaling parameter λ (Case 3), the anisotropy strength ε (Case 4), and the radius of the initial seed R_0 (Case 5), some of which (as discussed in Section 4) were shown to have a strong

Table 2
Simulation parameters and time for case studies with 64 processes.[a]

Case (#)	λ	ε	R_0 (d_0)	υ	Simulation time (hour)[b]
1	5	0.03	25	1.3×10^{-3}	17.3
2	5	0.03	25	1.8×10^{-3}	17.4
3	10	0.03	25	1.3×10^{-3}	16.6
4	10	0.01	25	1.3×10^{-3}	12.3
5	10	0.03	180	1.3×10^{-3}	12.7

[a] The rest parameters are the same as Table 1.
[b] For Cases 4 and 5, simulation is terminated at $t/\tau_0 = 612.7$.

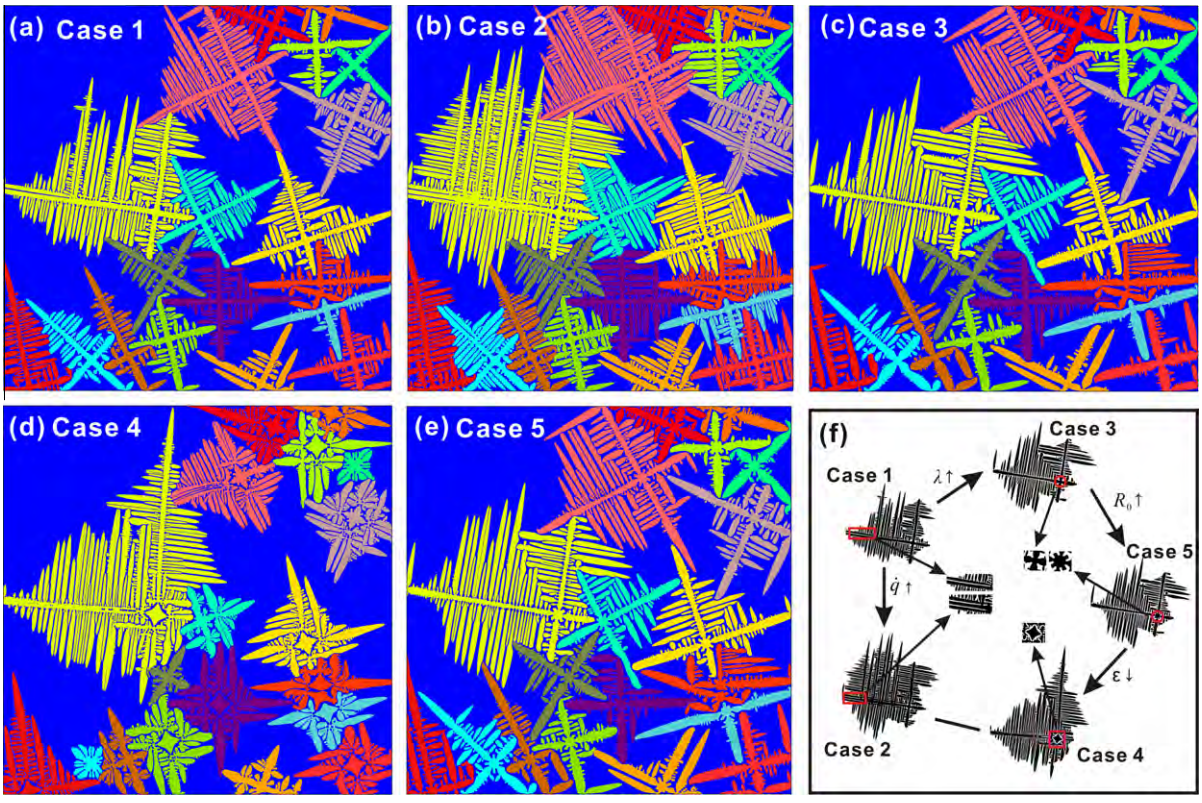

Fig. 8. Simulated dendrite morphologies at $t/\tau_0 = 408.5$ using simulation parameters listed in Table 1; (a)–(e) are corresponding to Case 1 to Case 5 in Table 2, and (f) shows the comparison of dendrite morphology evolved from Seed 1 in more details.

influences on dendrite morphology in some cases. Fig. 8a–e show the related dendrite morphologies calculated at $t/\tau_0 = 408.5$ using the parameters corresponding to Cases 1–5 in Table 2. In order to further clarify the influences of these parameters, the growth of a single dendrite, evolving from seed 1 (indicated by 1 in Fig. 6a) is retrieved and shown in more detail in Fig. 8f.

As shown from Fig. 8a and b, an increase in the cooling term $\dot{q}$ from 1.3×10^{-3} to 1.8×10^{-3} leads to a more developed and mature dendrite morphology. Comparison of Case 3 and Case 5 in Fig. 8f shows that as the initial seed radius increases, extra branches grow between the main arms, as already observed and discussed in Section 4. But this feature, as highlighted by the zoom-in insert, only survives locally around the seed center. The subsequent growth of the dendrite gives an overall very similar final dendrite morphology. The effect of the anisotropy strength, (comparing Case 5 and Case 4 in Fig. 8f), is more dramatic than other parameters. Due to the low amplitude of the anisotropy in Case 4, the dendrite tends to grow in all directions with much more tip splitting, resulting in a more seaweed-like morphology. Another prominent feature according to Case 4, as can also be seen from Fig. 8d and f, is that a four-fold structure exists at each dendrite center, which was evolved from the initial seed. The tip of such structure then grows, acting more like a newly formed seed, into a developed dendrite morphology.

In theory, by applying the proposed parallel multigrid approach the computing domain of the phase field problem can be further enlarged (i.e. $>8192 \times 8192$). The overall computing time can be maintained or even reduced as long as the process number is increased. As mentioned, 64 cores were applied during these tests, which compared with the capacity of the supercomputers nowadays is rather a smaller number, which is usually approaching an order of magnitude of 10^4–10^5. In this respect, the capacity of the simulation for the two dimensional case could include dendrites in a number of 10^3–10^4. Nevertheless, the physics inside would stay the same.

6. Conclusions

An implicit, parallel multigrid computing scheme has been developed and successfully applied to solve phase field equations for the simulation of multi-dendrite shape evolution under coupled thermal-solute conditions. Through detailed parametric studies, convergence evaluation and computation efficiency analysis, the following conclusions can be drawn:

(1) The parallel multigrid approach is highly efficient and scalable with a well-behaved monotonous convergence characteristic. For metal alloys in the coupled thermal-solute situation with a Lewis number approaching reality i.e. $\sim 10^4$, a time marching step of 3–4 orders of magnitude larger than the limit imposed in explicit algorithms has been realized without compromising accuracy.
(2) The parallel multigrid approach can efficiently utilize supercomputing resources to simulate many dendrites under realistic solidification conditions with much less computing time than previous work, including multiple dendrite growth, impingement, high-order side branching, and solute segregation, overcoming a major restriction of the phase field approach.
(3) Parametric studies concerning multi-dendrite growth reveal that the dendrite morphology is highly dependent on the assumed crystal anisotropy strength and the cooling rate, but not on the assumed seed radius and the scaling parameter λ. Under some circumstances, dendrite growth behavior such as tip splitting not captured in previous simulations was predicted.

The parallel multigrid approach has been shown to be robust and suitable for solving the two dimensional phase field problem. However, there are two main points that the current model lacks:

(1) No adaptive mesh was employed during the multi-dendrite simulations. The advantage of the adaptive mesh is that the mesh is refined only in the interface area, which can lead to an overall improvement in computing performance. However, the magnitude of the computing residual error is usually increased in adaptive schemes, which in turn is affected by the noise applied during the simulations. Secondly, a computational load imbalance may arise when a parallel scheme is also applied because an adaptive mesh can destroy the load balance of the data with respect to each parallel process.
(2) Important process physics such as convection, buoyancy, etc were not considered here. Also, the current model does not consider growth of the grain boundaries at the last stage of the solidification. However, the parallel multigrid approach should provide a robust and efficient starting point for incorporating these additional physics.

Acknowledgements

The authors would like to thank the UK Royal Academy of Engineering through the Newton International Fellowship Scheme for financial support, the Oxford Supercomputer Centre for access to supercomputing facilities and support on parallel programming.

Appendix A

The related terms in Eqs. (18)–(20) are given as:

$$P_{S,\phi} = \left(\frac{1}{Le} + Mc_\infty[1 + (1-k)U]\right)A(\psi_{i,j}^n)^2\phi_{i,j}^{n+1} - 8p_{0,\phi}\phi_{i,j}^{n+1} - b\Delta t\left[\phi_{i,j}^{n+1}\left(1 - (\phi_{i,j}^{n+1})^2\right) - \lambda\left(1 - (\phi_{i,j}^{n+1})^2\right)^2(\theta + Mc_\infty U)(1 + \dot{\xi})\right] \tag{29}$$

$$p_{0,\phi} = -\frac{b\cdot\Delta t}{3\cdot\Delta x^2}A(\psi_{i,j}^n)^2 \tag{30}$$

$$p_{x,\phi} = \frac{b\cdot\Delta t}{2\cdot\Delta x}\left(2\omega\varepsilon_4\sin(\omega\psi_{i,j}^n)A(\psi_{i,j}^n)\frac{\partial\psi_{i,j}^n}{\partial x} + \omega^2\varepsilon_4(\cos(\omega\psi_{i,j}^n) + \varepsilon_4\cos(2\omega\psi_{i,j}^n))\frac{\partial\psi_{i,j}^n}{\partial y}\right) \tag{31}$$

$$p_{y,\phi} = \frac{b\cdot\Delta t}{2\cdot\Delta x}\left(2\omega\varepsilon_4\sin(\omega\psi_{i,j}^n)A(\psi_{i,j}^n)\frac{\partial\psi_{i,j}^n}{\partial y} - \omega^2\varepsilon_4(\cos(\omega\psi_{i,j}^n) + \varepsilon_4\cos(2\omega\psi_{i,j}^n))\frac{\partial\psi_{i,j}^n}{\partial x}\right) \tag{32}$$

$$R_\phi = -a_0\phi_{i,j}^n - a_1\phi_{i,j}^{n-1} \tag{33}$$

$$p_{S,U} = \frac{1+k}{2} - \frac{1-k}{2}\phi_{i,j}^{n+1} - b\cdot\Delta t\cdot(1-k)p_{C,U} + \frac{4b\Delta t}{3\Delta x^2}\left(1 - \phi_{i,j}^{n+1}\right)\tilde{D} \tag{34}$$

$$p_{0,U} = -\frac{b\Delta t\left(1 - \phi_{i,j}^{n+1}\right)\tilde{D}}{6\Delta x^2} \tag{35}$$

$$p_{x,U} = \frac{b\Delta t}{4\Delta x}\left[-\tilde{D}\frac{\phi_{i+1,j}^{n+1} - \phi_{i-1,j}^{n+1}}{2\Delta x} + \frac{1-k}{\sqrt{2}}\frac{\phi_{i,j}^{n+1} - \phi_{i,j}^n}{\Delta t}\sin(\psi_{i,j}^n)\right] \tag{36}$$

$$p_{y,U} = \frac{b\Delta t}{4\Delta x}\left[-\tilde{D}\frac{\phi_{i,j+1}^{n+1} - \phi_{i,j-1}^{n+1}}{2\Delta x} + \frac{1-k}{\sqrt{2}}\frac{\phi_{i,j}^{n+1} - \phi_{i,j}^n}{\Delta t}\cos(\psi_{i,j}^n)\right] \tag{37}$$

$$R_U = b\Delta t P_{C,U} - \left(\frac{1+k}{2} - \frac{1-k}{2}\phi_{i,j}^{n+1}\right)\left(a_0U_{i,j}^n + a_1U_{i,j}^{n-1}\right) \tag{38}$$

$$p_{C,U} = \frac{1}{2\sqrt{2}}\frac{\phi_{i,j}^{n+1} - \phi_{i,j}^n}{\Delta t}\left(-\sin\left(\psi_{i,j}^{n+1}\right)\frac{\partial\psi_{i,j}^{n+1}}{\partial x} + \cos\left(\psi_{i,j}^{n+1}\right)\frac{\partial\psi_{i,j}^{n+1}}{\partial y}\right) + \frac{1}{2\sqrt{2}}\left(\cos(\psi_{i,j}^{n+1})\frac{\partial^2\psi_{i,j}^{n+1}}{\partial x\partial t} + \sin(\psi_{i,j}^{n+1})\frac{\partial^2\psi_{i,j}^{n+1}}{\partial y\partial t}\right) + \frac{1}{2}\frac{\phi_{i,j}^{n+1} - \phi_{i,j}^n}{\Delta t} \tag{39}$$

$$p_{S,\theta} = 1 + \frac{8b\Delta t\tilde{\alpha}}{3\Delta x^2} \tag{40}$$

$$p_{0,\theta} = -\frac{b\Delta t\tilde{\alpha}}{3\Delta x^2} \tag{41}$$

$$R_\theta = -a_0\theta_{i,j}^n - a_1\theta_{i,j}^{n-1} + 0.5\cdot b\left(\phi_{i,j}^{n+1} - \phi_{i,j}^n\right) + b\Delta t\dot{q} \tag{42}$$

The derivatives of angle ψ with respect to axis are:

$$\frac{\partial\psi}{\partial x} = \frac{\phi_x\phi_{xy} - \phi_y\phi_{xx}}{|\nabla\phi|^2} \tag{43}$$

$$\frac{\partial\psi}{\partial y} = \frac{\phi_x\phi_{yy} - \phi_y\phi_{xy}}{|\nabla\phi|^2} \tag{44}$$

where

$$\phi^n_{xy} = \frac{\partial^2 \phi}{\partial x \partial y} = \frac{\phi^n_{i+1,j+1} - \phi^n_{i-1,j+1} + \phi^n_{i-1,j-1} - \phi^n_{i+1,j-1}}{4\Delta x^2} \tag{45}$$

References

[1] R. Kobayashi, A numerical approach to three-dimensional dendritic solidification, Exp. Math. 3 (1994) 59–81.
[2] W.J. Boettinger, J.A. Warren, C. Beckermann, A. Karma, Phase-field simulation of solidification, Ann. Rev. Mater. Res. 32 (2002) 163–194.
[3] L.-Q. Chen, Phase-field models for microstructure evolution, Ann. Rev. Mater. Res. 32 (2002) 113–140.
[4] L. Granasy, T. Pusztai, T. Borzsonyi, J.A. Warren, J.F. Douglas, A general mechanism of polycrystalline growth, Nat. Mater. 3 (2004) 645–650.
[5] L. Granasy, T. Pusztai, J.A. Warren, Modelling polycrystalline solidification using phase field theory, J. Phys. Condens. Matter 16 (2004) R1205–R1235.
[6] T. Haxhimali, A. Karma, F. Gonzales, M. Rappaz, Orientation selection in dendritic evolution, Nat. Mater. 5 (2006) 660–664.
[7] N. Moelans, B. Blanpain, P. Wollants, An introduction to phase-field modeling of microstructure evolution, Calphad 32 (2008) 268–294.
[8] A. Karma, Phase-field formulation for quantitative modeling of alloy solidification, Phys. Rev. Lett. 87 (2001) 115701.
[9] L. Gránásy, T. Börzsönyi, T. Pusztai, Nucleation and bulk crystallization in binary phase field theory, Phys. Rev. Lett. 88 (2002) 206105.
[10] J.J. Li, J.C. Wang, Q. Xu, G.C. Yang, Comparison of Johnson–Mehl–Avrami–Kologoromov (JMAK) kinetics with a phase field simulation for polycrystalline solidification, Acta Mater. 55 (2007) 825–832.
[11] A. Badillo, C. Beckermann, Phase-field simulation of the columnar-to-equiaxed transition in alloy solidification, Acta Mater. 54 (2006) 2015–2026.
[12] C.W. Lan, C.J. Shih, M.H. Lee, Quantitative phase field simulation of deep cells in directional solidification of an alloy, Acta Mater. 53 (2005) 2285–2294.
[13] A. Karma, W.-J. Rappel, Quantitative phase-field modeling of dendritic growth in two and three dimensions, Phys. Rev. E 57 (1998) 4323.
[14] A. Karma, W.-J. Rappel, Phase-field model of dendritic sidebranching with thermal noise, Phys. Rev. E 60 (1999) 3614–3625.
[15] J.C. Ramirez, C. Beckermann, A. Karma, H.J. Diepers, Phase-field modeling of binary alloy solidification with coupled heat and solute diffusion, Phys. Rev. E 69 (2004) 051607.
[16] J. Rosam, P.K. Jimack, A. Mullis, A fully implicit, fully adaptive time and space discretisation method for phase-field simulation of binary alloy solidification, J. Comput. Phys. 225 (2007) 1271–1287.
[17] J. Rosam, P.K. Jimack, A.M. Mullis, An adaptive, fully implicit multigrid phase-field model for the quantitative simulation of non-isothermal binary alloy solidification, Acta Mater. 56 (2008) 4559–4569.
[18] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numnerical recipes – the art of science computing, 3rd ed., Cambridge University Press, 2007.
[19] A.M. Mullis, R.F. Cochrane, A phase field model for spontaneous grain refinement in deeply undercooled metallic melts, Acta Mater. 49 (2001) 2205–2214.
[20] R. Kobayashi, J.A. Warren, W.C. Carter, Vector-valued phase field model for crystallization and grain boundary formation, Physica D 119 (1998) 415–423.
[21] R. Kobayashi, J.A. Warren, W. Craig Carter, A continuum model of grain boundaries, Physica D 140 (2000) 141–150.
[22] J.A. Warren, R. Kobayashi, A.E. Lobkovsky, W.C. Carter, Extending phase field models of solidification to polycrystalline materials, Acta Mater. 51 (2003) 6035–6058.
[23] L. Beltran-Sanchez, D. Stefanescu, A quantitative dendrite growth model and analysis of stability concepts, Metall. Mater. Trans. A. 35 (2004) 2471–2485.
[24] M.F. Zhu, D.M. Stefanescu, Virtual front tracking model for the quantitative modeling of dendritic growth in solidification of alloys, Acta Mater. 55 (2007) 1741–1755.
[25] U. Trottenberg, C. Oosterlee, A. Schuller, Multigrid, Academic Press, Harcourt place, 32 Jamestown Road, London NW1 7BY, UK, 2001.
[26] A. Brandt, Multi-grid techniques: 1984 guide with applications to fluid dynamics, GMD-Studien, Augustin, Germany, 1984.
[27] A. Brandt, Multilevel computations of integral transforms and particle interactions with oscillatory kernels, Comp. Phys. Commun. 65 (1991) 24–38.
[28] A. Grama, A. Gupta, G. Karypis, V. Kumar, Introduction to Parallel Computing, second ed., Addison-Wesley, London, 2003.
[29] W. Gropp, E. Lusk, A. Skjellum, Using MPI: Portable Parallel Programming with the Message-Passing Interface, second ed., MIT Press, Cambridge, 1999.
[30] B. Chapman, G. Jost, R. van der Pas, Using OpenMP Portable Shared Memory Parallel Programming, MIT Press, Cambridge, 2007.
[31] L.A. Smith, Mixed Mode MPI/OpenMP Programming, in, UKHEC, Edinburgh: Edinburgh Parallel Computing Center, 2001.
[32] J.C. Ramirez, C. Beckermann, Examination of binary alloy free dendritic growth theories with a phase-field model, Acta Mater. 53 (2005) 1721–1736.
[33] W. Kurz, D.J. Fisher, Fundamentals of Solidification, Fourth ed., Trans Tech Publications Ltd, 1998.
[34] E. Ben-Jacob, P. Garik, The formation of patterns in non-equilibrium growth, Nature 343 (1990) 523–530.