mathematics

MDPI

Article

Acceleration and Parallelization of a Linear Equation Solver for Crack Growth Simulation Based on the Phase Field Model

Gaku Ishii [1], Yusaku Yamamoto [1,*] and Takeshi Takaishi [2]

[1] Department of Communication Engineering and Informatics, The University of Electro-Communications, Tokyo 182-8585 , Japan; QZY13306@nifty.com
[2] Department of Mathematical Engineering, Musashino University, Tokyo 135-8181, Japan; taketaka@musashino-u.ac.jp
* Correspondence: yusaku.yamamoto@uec.ac.jp; Tel.: +81-42-443-5360

Abstract: We aim to accelerate the linear equation solver for crack growth simulation based on the phase field model. As a first step, we analyze the properties of the coefficient matrices and prove that they are symmetric positive definite. This justifies the use of the conjugate gradient method with the efficient incomplete Cholesky preconditioner. We then parallelize this preconditioner using so-called block multi-color ordering and evaluate its performance on multicore processors. The experimental results show that our solver scales well and achieves an acceleration of several times over the original solver based on the diagonally scaled CG method.

Keywords: crack growth simulation; phase field model; conjugate gradient method; incomplete Cholesky factorization; parallelization; block red-black ordering; performance evaluation

MSC: 15A06; 35F61; 65F08; 65M60; 74R99; 74S05

Citation: Ishii, G.; Yamamoto, Y.; Takaishi, T. Acceleration and Parallelization of a Linear Equation Solver for Crack Growth Simulation Based on the Phase Field Model. *Mathematics* **2021**, *9*, 2248. https://doi.org/10.3390/math9182248

Academic Editor: Junseok Kim

Received: 31 July 2021
Accepted: 10 September 2021
Published: 13 September 2021

1. Introduction

Crack growth is a ubiquitous phenomenon that affects the strength and functions of materials and structures. Since cracks can grow very quickly, simulation is a useful tool to study their growth process in detail. Simulation can also be used to predict the generation of cracks under given stresses or other conditions. In the conventional method of crack growth simulation, the finite element method (FEM) is used and the mesh is regenerated at every time step so that the mesh boundary conforms to the crack boundary. However, this incurs huge computational cost. Moreover, to determine the direction of crack growth, it is usually necessary to evaluate the total energy for various possible scenarios. This also adds to the computational cost.

To resolve these problems, a crack growth simulation method based on the phase field model has been proposed [1–4]. In this model, a new continuous dependent variable $z(\boldsymbol{x}, t)$ called the *phase field* [5,6] is introduced in addition to the displacement $\boldsymbol{u}(\boldsymbol{x}, t)$. This variable expresses the degree of the crack at each point: $z \simeq 0$ if there is no crack and $z \simeq 1$ if there is a crack. Moreover, a partial differential equation (PDE) describing the time evolution of $z(\boldsymbol{x}, t)$ is also derived along with that of $\boldsymbol{u}(\boldsymbol{x}, t)$. This makes it possible to determine the direction of crack growth without the total energy evaluations. Hence, the method is a promising candidate for real-time three-dimensional crack growth simulation. Takaishi et al. implemented a crack growth simulation program based on this method and showed that it works well in various examples. For evaluation purposes, a two-dimensional FreeFEM code based on the method is also available.

When implementing this method, the time-discretization of the PDEs both for $\boldsymbol{u}(\boldsymbol{x}, t)$ and $z(\boldsymbol{x}, t)$ is usually done with the semi-implicit method to ensure numerical stability. In that case, the time taken to solve the resulting linear simultaneous equations is often dominant in the total computation time. Takaishi et al.'s simulation program uses the

conjugate gradient (CG) method with a diagonal scaling preconditioner for both of the equations for $\boldsymbol{u}(\boldsymbol{x}, t)$ and $z(\boldsymbol{r}, t)$, which is a simple preconditioner applicable to a wide class of matrices but is not very efficient. The reason for using this preconditioner is that the properties of the coefficient matrices have not been fully understood yet.

In this paper, we aim to accelerate this linear equation solver by employing a more powerful preconditioner. This will help to speed up the linear equation solution that accounts for a large part of the computing time and open a way to solve larger scale and more realistic problems. To this end, we make the following contributions. First, we analyze the properties of the coefficient matrices obtained by applying semi-implicit time discretization and space discretization by FEM to the PDEs for $\boldsymbol{u}(\boldsymbol{x}, t)$ and $z(\boldsymbol{x}, t)$. We show that, under appropriate boundary conditions, both of these coefficient matrices become symmetric positive definite (SPD). This justifies the use of the incomplete Cholesky (IC) preconditioner, which is more powerful than diagonal scaling. Second, we show that the IC preconditioner can be parallelized efficiently using the block multi-color ordering proposed by Iwashita et al. [7,8]. In fact, our numerical experiments suggest that the number of CG iterations increases only slightly, if at all, by this parallelization method compared to the sequential case. Finally, we optimize several performance parameters such as the block division scheme and show that the resulting parallel solver is several times faster than the diagonally scaled CG solver on multicore processors. Our results will be applicable to crack growth simulation in a variety of fields in science and engineering, such as the prediction of cracking in buildings and bridges and the analysis of solder cracking in circuit boards [2].

The rest of this paper is structured as follows. In Section 2, we briefly describe the crack simulation method based on the phase field model and its space and time discretization. In Section 3, the properties of the coefficient matrices arising from the discretization are analyzed, and their symmetric positive definiteness is proved under certain conditions. Section 4 describes the parallelization of the IC preconditioner using block multi-color ordering. Numerical results are given in Section 5. Finally, Section 6 concludes the paper.

2. Crack Growth Simulation Based on the Phase Field Model

In this section, we briefly explain the crack growth simulation method based on the phase field model, which was proposed by Takaishi and Kimura [3]. We begin with the two-dimensional case and then proceed to the three-dimensional case.

2.1. The Two-Dimensional Case

Let us consider crack growth in a thin panel as shown in Figure 1. Here, we focus on the so-called *mode 3 crack*, in which the displacement of the panel is in the direction perpendicular to the panel. Thus, we treat the problem as two-dimensional and denote the displacement at a point $\boldsymbol{x} = (x_1, x_2)$ by a scalar variable $u(\boldsymbol{x}, t)$. We denote the region by Ω, its boundary by Γ, and the crack, which is modeled as a curve on Ω, by Σ. Hence, $u(\boldsymbol{x})$ is discontinuous across Σ. In the example shown in Figure 1, the Dirichlet boundary condition is imposed on $\Gamma_D \subset \Gamma$, while the Neumann boundary condition is imposed on $\Gamma_N = \Gamma \backslash \Gamma_D$.

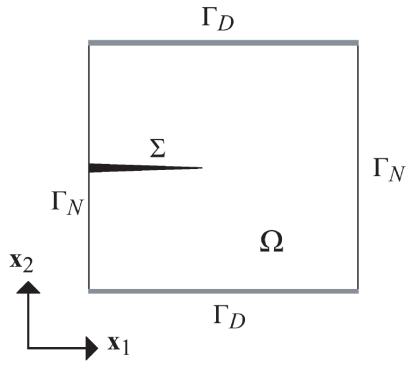


Figure 1. A two-dimensional region Ω and crack Σ.

The basic idea of the crack growth simulation method to be described below goes back to Griffith [9]. He proposed the expression of the total energy of the system as a sum of the elasticity energy E_1 and the surface energy E_2 due to the existence of a crack. Both of these energies depend on the crack Σ, as well as on $u(\boldsymbol{x}, t)$, so we denote them as $E_1[u, \Sigma]$ and $E_2[u, \Sigma]$. Griffith assumed that crack growth occurs if the total energy $E[u, \Sigma] = E_1[u, \Sigma] + E_2[u, \Sigma]$ decreases due to that factor. In the two-dimensional case, E_1 and E_2 can be written as follows [1]:

$$E_1[u, \Sigma] = \frac{\mu}{2} \int_{\Omega \setminus \Sigma} |\nabla u|^2 \, d\boldsymbol{x} \tag{1}$$

$$E_2[u, \Sigma] = \int_{\Sigma} \gamma(\boldsymbol{x}) \, ds, \tag{2}$$

where $\gamma(\boldsymbol{x}) > 0$ is fracture toughness at $\boldsymbol{x}$ and μ is one of Lamé's constants. Equation (1) means that the elasticity energy is expressed as an integral of $\frac{\mu}{2}|\nabla u|^2$ over the entire region Ω, excluding the crack Σ. This is because the difference of u across Σ does not contribute to the elasticity energy. On the other hand, the surface energy (2) is expressed as a line integral along the crack.

While in principle (1) and (2) can be used to study the development of crack Σ, they are not convenient for numerical computation because the regions of the integral depend on Σ and change from step to step. To resolve this problem, Bourdin et al. [10] introduced a phase field variable $z(\boldsymbol{x}, t)$ that expresses the degree of crack at $(\boldsymbol{x}, t)$: $z \simeq 0$ if there is no crack and $z \simeq 1$ if there is a crack. $z(\boldsymbol{x}, t)$ is assumed to be a smooth function of $\boldsymbol{x}$, and the transition between $z = 0$ and $z = 1$ is assumed to occur across a narrow region of width $\simeq \epsilon$, where $\epsilon > 0$ is a regularization parameter [11]. Under these assumptions, Bourdin et al. propose the use of the following regularized total energy functional instead of $E[u, \Sigma]$:

$$\mathcal{E}[u, z; \epsilon] = \frac{\mu}{2} \int_{\Omega} (1 - z)^2 |\nabla u|^2 \, d\boldsymbol{x} + \frac{1}{2} \int_{\Omega} \gamma(\boldsymbol{x}) \left(\epsilon |\nabla z|^2 + \frac{1}{\epsilon} z^2 \right) d\boldsymbol{x}. \tag{3}$$

In this formulation, the region of the integral is the entire region Ω for both the elasticity and surface energies, which greatly simplifies the numerical procedure.

For the efficient computation of $u(\boldsymbol{x}, t)$ and $z(\boldsymbol{x}, t)$ based on (3), Takaishi and Kimura [3] proposed the use of the gradient flow

$$\frac{\partial u}{\partial \tau} = -\alpha_1 \frac{\delta \mathcal{E}}{\delta u}, \quad \frac{\partial z}{\partial \tau} = -\alpha_2 \frac{\delta \mathcal{E}}{\delta z}, \tag{4}$$

where τ is a virtual time parameter and α_1 and α_2 are time constants. It can easily be shown that if $u(\boldsymbol{x}, t)$ and $z(\boldsymbol{x}, t)$ evolve according to (4), the total energy functional (3) decreases monotonically. Thus we can expect that the (local) minimum of $\mathcal{E}[u, z; \epsilon]$ is reached as $\tau \to \infty$. Furthermore, if α_1 and α_2 are sufficiently large, $u(\boldsymbol{x}, t)$ and $z(\boldsymbol{x}, t)$

are expected to reach the minimizer of $\mathcal{E}[u, z; \epsilon]$ for given external conditions such as the boundary conditions and external forces (if any) very quickly. Thus, we can regard $u(\boldsymbol{x}, t)$ and $z(\boldsymbol{x}, t)$ determined by (4) as instantaneous reactions to the external conditions and (4) as approximately describing the time evolution of $u(\boldsymbol{x}, t)$ and $z(\boldsymbol{x}, t)$. By computing $\frac{\delta \mathcal{E}}{\delta u}$ and $\frac{\delta \mathcal{E}}{\delta z}$ explicitly, we obtain the following set of PDEs:

$$\begin{cases} \alpha_1 \frac{\partial u}{\partial t} &= \nabla \cdot \left((1-z)^2 \nabla u \right) & (\boldsymbol{x} \in \Omega), \\ \alpha_2 \frac{\partial z}{\partial t} &= \left(\epsilon \nabla \cdot (\gamma(\boldsymbol{x}) \nabla z) - \frac{\gamma(\boldsymbol{x})}{\epsilon} z + |\nabla u|^2 (1-z) \right)_+ & (\boldsymbol{x} \in \Omega), \\ u(\boldsymbol{x}, t) &= g(\boldsymbol{x}, t) & (\boldsymbol{x} \in \Gamma_D), \\ \frac{\partial u}{\partial n} &= 0 & (\boldsymbol{x} \in \Gamma_N), \\ \frac{\partial z}{\partial n} &= 0 & (\boldsymbol{x} \in \Gamma), \\ u(\boldsymbol{x}, 0) &= u_0(\boldsymbol{x}) & (\boldsymbol{x} \in \Omega), \\ z(\boldsymbol{x}, 0) &= z_0(\boldsymbol{x}) \in [0, 1] & (\boldsymbol{x} \in \Omega), \end{cases} \tag{5}$$

where we changed τ to t and set $\mu = 1$ for simplicity. $g(\boldsymbol{r}, t)$ denotes the Dirichlet boundary condition that causes the development of the crack. The symbol $(\cdot)_+$ in the second equation means $\max(\cdot, 0)$, which expresses the fact that a crack does not vanish once it is created [12]. See [13] for the treatment of partial differential equations with such terms. $\frac{\partial}{\partial n}$ denotes the partial derivative in the direction of the outgoing normal vector.

Crack growth simulation based on (5) has the following advantages:

- The direction of crack growth is automatically determined by the PDEs. Hence, total energy evaluation under multiple possible scenarios, which is needed in simulation methods based directly on (3), is not necessary;
- By introducing the phase field variable $z(\boldsymbol{x}, t)$ and the regularization parameter ϵ, the divergence of the stress at the tip of the crack is kept to a level manageable by numerical methods;
- It is not necessary to regenerate the mesh at every time step to conform to the crack boundary.

Our theoretical analysis and numerical experiments in the two-dimensional case are based on these PDEs.

2.2. The Three-Dimensional Case

In the three-dimensional case, the displacement becomes a vector field variable $\boldsymbol{u}(\boldsymbol{x}, t) = (u_1(\boldsymbol{x}, t), u_2(\boldsymbol{x}, t), u_3(\boldsymbol{x}, t))^\top$, where $\boldsymbol{x} = (x_1, x_2, x_3)$. The phase field variable $z(\boldsymbol{x}, t)$ is still a scalar field variable. Using the same idea as in the previous subsection, we can derive the set of PDEs corresponding to (5). In the following, we assume isotropic materials for simplicity. First, let us define the strain tensor $\epsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$ and the stress tensor σ_{ij}. These tensors are connected by Hooke's law, which has the following form in the case of isotropic materials:

$$\sigma_{ij} = \lambda \delta_{ij} \nabla \cdot \boldsymbol{u} + 2\mu \epsilon_{ij} \quad (i, j = 1, 2, 3), \tag{6}$$

where λ and μ are Lamé's constants and δ_{ij} is Kronecker's delta. We also write the stress tensor as

$$\boldsymbol{\sigma} = (\sigma_{ij}) = (\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3). \tag{7}$$

Using these tensors, the elasticity energy density $e(\boldsymbol{u})$, which corresponds to $\frac{\mu}{2}|\nabla u|^2$ in the two-dimensional case, can be defined as follows:

$$e(\boldsymbol{u}) = \frac{1}{2} \sum_{i=1}^{3} \sum_{j=1}^{3} \sigma_{ij} \epsilon_{ij} = \frac{1}{2} \lambda (\nabla \cdot \boldsymbol{u})^2 + \mu \sum_{i=1}^{3} \sum_{j=1}^{3} \epsilon_{ij} \epsilon_{ij}. \tag{8}$$

Using this, the regularized total energy functional is defined as

$$\mathcal{E}[u,z;\epsilon] = \int_\Omega (1-z)^2 e(\boldsymbol{u})\,d\boldsymbol{x} + \frac{1}{2}\int_\Omega \gamma(\boldsymbol{x})\left(\epsilon|\nabla z|^2 + \frac{1}{\epsilon}z^2\right)d\boldsymbol{x}, \tag{9}$$

which has the same form as (3) except that $\frac{\mu}{2}|\nabla u|^2$ is replaced with $e(\boldsymbol{u})$. Note that now Ω is a three-dimensional region and $\int_\Omega \cdot\, d\boldsymbol{x}$ denotes a volume integral. By considering the gradient flow as in (4), we obtain the following set of PDEs after some calculations [2]:

$$\left\{\begin{array}{rcll} \alpha_1 \frac{\partial \boldsymbol{u}}{\partial t} & = & \nabla((\lambda+\mu)(1-z)^2(\nabla\cdot\boldsymbol{u})) + \nabla\cdot(\mu(1-z)^2\nabla\boldsymbol{u}) & (\boldsymbol{x}\in\Omega), \\ \alpha_2 \frac{\partial z}{\partial t} & = & \left(\epsilon\nabla\cdot(\gamma(\boldsymbol{x})\nabla z) - \frac{\gamma(\boldsymbol{x})}{\epsilon}z + 2e(\boldsymbol{u})(1-z)\right)_+ & (\boldsymbol{x}\in\Omega), \\ \boldsymbol{u}(\boldsymbol{x},t) & = & \boldsymbol{g}(\boldsymbol{x},t) & (\boldsymbol{x}\in\Gamma_D), \\ \boldsymbol{s}_j\cdot\boldsymbol{n} & = & 0 \quad (j=1,2,3) & (\boldsymbol{x}\in\Gamma_N), \\ \frac{\partial z}{\partial n} & = & 0 & (\boldsymbol{x}\in\Gamma), \\ u(\boldsymbol{x},0) & = & u_0(\boldsymbol{x}) & (\boldsymbol{x}\in\Omega), \\ z(\boldsymbol{x},0) & = & z_0(\boldsymbol{x})\in[0,1]. & (\boldsymbol{x}\in\Omega). \end{array}\right. \tag{10}$$

Here,

$$\nabla\boldsymbol{u} = \begin{pmatrix} \nabla u_1(\boldsymbol{x},t) \\ \nabla u_2(\boldsymbol{x},t) \\ \nabla u_3(\boldsymbol{x},t) \end{pmatrix} = \begin{pmatrix} \frac{\partial u_1}{\partial x_1} & \frac{\partial u_1}{\partial x_2} & \frac{\partial u_1}{\partial x_3} \\ \frac{\partial u_2}{\partial x_1} & \frac{\partial u_2}{\partial x_2} & \frac{\partial u_2}{\partial x_3} \\ \frac{\partial u_3}{\partial x_1} & \frac{\partial u_3}{\partial x_2} & \frac{\partial u_3}{\partial x_3} \end{pmatrix} \tag{11}$$

and

$$\nabla\cdot(\mu(1-z)^2\nabla\boldsymbol{u}) = \begin{pmatrix} \nabla\cdot(\mu(1-z)^2\nabla u_1) \\ \nabla\cdot(\mu(1-z)^2\nabla u_2) \\ \nabla\cdot(\mu(1-z)^2\nabla u_3) \end{pmatrix}. \tag{12}$$

where $\boldsymbol{n} = (n_1, n_2, n_3)^\top$ denotes the unit normal vector. Note that the first equation of (10) can also be written as

$$\alpha_1 \frac{\partial u_j}{\partial t} = \nabla\cdot\left((1-z)^2\boldsymbol{s}_j\right) \quad (j=1,2,3,\ \boldsymbol{x}\in\Omega). \tag{13}$$

This can be verified directly using the definitions of ϵ_{ij}, σ_{ij} and $\boldsymbol{s}_j$. The fourth equation of (10) is often expressed as

$$\boldsymbol{\sigma}\cdot\boldsymbol{n} = \boldsymbol{0}. \tag{14}$$

and is known as the stress-free boundary condition.

2.3. Temporal Discretization

For the temporal discretization of (5) and (10), we use a semi-implicit method. More specifically, to compute the solution $(u_{t+\Delta t}, z_{t+\Delta_t})$at time $t+\Delta t$ from (u_t, z_t) at time t, we replace u and z in the right-hand side of the equation for $\partial u/\partial t$ by $u_{t+\Delta t}$ and z_t, respectively. Similarly, in the right-hand side of the equation for $\partial z/\partial t$, we replace u and z by u_t and $z_{t+\Delta t}$, respectively. Thus, in the two-dimensional case, we obtain the following set of equations, which are linear both in $u_{t+\Delta t}$ and $z_{t+\Delta t}$:.

$$\left\{\begin{array}{rcll} \alpha_1 \frac{u_{t+\Delta t}-u_t}{\Delta t} & = & \nabla\cdot((1-z_t)^2\nabla u_{t+\Delta t}), & (15) \\ \alpha_2 \frac{\tilde{z}_{t+\Delta t}-z_t}{\Delta t} & = & \epsilon\nabla\cdot(\gamma(\boldsymbol{x})\nabla\tilde{z}_{t+\Delta t}) - \frac{\gamma(\boldsymbol{x})}{\epsilon}\tilde{z}_{t+dt} + |\nabla u_t|^2(1-\tilde{z}_{t+\Delta t}), & (16) \\ z_{t+\Delta t} & = & \max(\tilde{z}_{t+\Delta t}, z_t). & (17) \end{array}\right.$$

Here, (17) corresponds to taking $(\cdot)_+$ In the numerical simulation; we set $\alpha_1 = 0$, which corresponds to assuming that the displacement $u(\boldsymbol{x},t)$ responds to the changes of $z(\boldsymbol{x},t)$ and the boundary conditions instantaneously.

The equations for the three-dimensional case are as follows:

$$\left\{\begin{array}{rcll} \alpha_1 \frac{\boldsymbol{u}_{t+\Delta t}-\boldsymbol{u}_t}{\Delta t} & = & \nabla((\lambda+\mu)(1-z_t)^2\nabla\cdot\boldsymbol{u}_{t+\Delta t}) + \nabla\cdot(\mu(1-z_t)^2\nabla\boldsymbol{u}_{t+\Delta t}), & (18) \\ \alpha_2 \frac{\tilde{z}_{t+\Delta t}-z_t}{\Delta t} & = & \epsilon\nabla\cdot(\gamma(\boldsymbol{x})\nabla\tilde{z}_{t+\Delta t}) - \frac{\gamma(\boldsymbol{x})}{\epsilon}\tilde{z}_{t+\Delta t} + 2e(\boldsymbol{u}_t)(1-\tilde{z}_{t+\Delta t}), & (19) \\ z_{t+\Delta t} & = & \max(\tilde{z}_{t+\Delta t}, z_t). & (20) \end{array}\right.$$

We set $\alpha_1 = 0$ also in this case. Using expression (13), Equation (18) can also be written as

$$\alpha_1 \frac{u_{t+\Delta t,j} - u_{t,j}}{\Delta t} = \nabla \cdot \left((1 - z_t)^2 \boldsymbol{s}_{t+\Delta t,j} \right) \quad (j = 1, 2, 3), \tag{21}$$

where $\boldsymbol{s}_{t+\Delta t,j}$ is the jth column of the stress tensor σ at time $t + \Delta t$.

3. Properties of the Coefficient Matrices Arising from Phase Field-Based Crack Growth Simulation

In this section, we study the properties of the coefficient matrices arising from the finite element discretization of the basic equations for the phase field-based crack growth simulation. In particular, we prove that these matrices become symmetric positive definite under certain assumptions. This is important to be able to apply the efficient incomplete Cholesky preconditioner to these matrices.

3.1. The Two-Dimensional Case

The weak forms

We start with the time-discretized Equation (15) with $\alpha_1 = 0$. This is a Poisson equation for $u_{t+\Delta t}$ with variable coefficient $(1 - z_t)^2$, and it is well known that its weak form can be written as follows:

$$\int_\Omega (1 - z_t)^2 \nabla v \cdot \nabla u_{t+\Delta t} \, d\boldsymbol{x} = 0, \tag{22}$$

where $v(\boldsymbol{x})$ is a test function with $v(\boldsymbol{x}) = 0$ on Γ_D. Finding $u_{t+\Delta t}$ satisfying (15) with $\alpha_1 = 0$ is equivalent to finding $u_{t+\Delta t}$ satisfying (22) for arbitrary test functions $v(\boldsymbol{x})$.

Next, we derive the weak form for (16). By multiplying (16) by a test function $w(\boldsymbol{x})$, integrating over Ω, using a slight extension of Green's identity (see Lemma 1),

$$\int_\Omega \phi \nabla \cdot (f \nabla \psi) \, d\boldsymbol{x} = \int_\Gamma \phi f \nabla \psi \cdot \boldsymbol{n} \, ds - \int_\Omega f \nabla \phi \cdot \nabla \psi \, d\boldsymbol{x} \tag{23}$$

with $\phi = w$, $\psi = \tilde{z}_{t+\Delta t}$ and $f = \gamma(\boldsymbol{x})$, and noting that $\nabla \tilde{z}_{r+\Delta t} \cdot \boldsymbol{n} = 0$ on $\Gamma_N = \Gamma$, we obtain

$$\begin{aligned} &\int_\Omega \left\{ \alpha_2 w \tilde{z}_{t+\Delta t} + \Delta t \left(\epsilon \gamma(\boldsymbol{x}) \nabla w \cdot \nabla \tilde{z}_{t+\Delta t} + w \frac{\gamma(\boldsymbol{x})}{\epsilon} \tilde{z}_{t+\Delta t} + w |\nabla u_t|^2 \tilde{z}_{t+\Delta t} \right) \right\} d\boldsymbol{x} \\ &\quad - \int_\Omega w(\alpha_2 z_t + \Delta t |\nabla u_t|^2) d\boldsymbol{x} = 0. \end{aligned} \tag{24}$$

Equations (17), (22) and (24) constitute the weak forms for the two-dimensional case.

Properties of the coefficient matrix for $u_{t+\Delta t}$

Now, we consider the properties of the matrix obtained by applying the finite element discretization to the weak form (22). Let $\phi_0(\boldsymbol{x})$ be a function satisfying the boundary condition $\phi_0(\boldsymbol{x}, t) = g(\boldsymbol{x}, t)$ $(\boldsymbol{x} \in \Gamma_D)$ and $\{\phi_j\}_{j=1}^m$ be basis functions that are zero on Γ_D. We approximate $u_{t+\Delta t}$ by $\hat{u}_{t+\Delta t}(\boldsymbol{x})$, which is a linear combination of these functions:

$$\hat{u}_{t+\Delta t} = \phi_0 + \sum_{j=1}^m a_j \phi_j. \tag{25}$$

Inserting this into (22) and choosing the test function as $v = \phi_i$, we obtain

$$\sum_{j=1}^m \left\{ \int_\Omega (1 - z_t)^2 \nabla \phi_i \cdot \nabla \phi_j \, dx \right\} a_j = - \int_\Omega (1 - z_t)^2 \nabla \phi_i \cdot \nabla \phi_0 \, dx. \tag{26}$$

By defining the matrix $C = (c_{ij}) \in \mathbb{R}^{m\times m}$ and the vector $\boldsymbol{d} = (d_i) \in \mathbb{R}^m$ by

$$\begin{cases} c_{ij} = \int_\Omega (1-z_t)^2 \nabla\phi_i \cdot \nabla\phi_j \, d\boldsymbol{x}, & (27) \\ d_i = -\int_\Omega (1-z_t)^2 \nabla\phi_i \cdot \nabla\phi_0 \, d\boldsymbol{x} & (28) \end{cases}$$

and letting $\boldsymbol{a} = (a_1, a_2, \ldots, a_m)^\top$, (26) can be written as follows.

$$C\boldsymbol{a} = \boldsymbol{d}. \tag{29}$$

Since $c_{ij} = c_{ji}$ from (27), it is clear that C is a symmetric matrix. Moreover, for an arbitrary nonzero vector $\boldsymbol{p} = (p_1, p_2, \ldots, p_m)^\top$, we have

$$\begin{aligned} \boldsymbol{p}^\top C\boldsymbol{p} &= \sum_{i=1}^m \sum_{j=1}^m p_j p_j \int_\Omega (1-z_t)^2 \nabla\phi_i \cdot \nabla\phi_j \, d\boldsymbol{x} \\ &= \int_\Omega (1-z_t)^2 \left(\sum_{i=1}^m p_i \nabla\phi_i \right) \cdot \left(\sum_{j=1}^m p_j \nabla\phi_j \right) d\boldsymbol{x} \\ &= \int_\Omega (1-z_t)^2 \left| \nabla \sum_{i=1}^m p_i \phi_i \right|^2 d\boldsymbol{x}. \end{aligned} \tag{30}$$

Noting that $\nabla \sum_{i=1}^m p_i\phi_i$ is not identically zero from the linear independence of $\{\phi_i\}$, we know that the integral in the right-hand side is positive as long as $\forall \boldsymbol{x}$, $0 \le z_t(\boldsymbol{x}) < 1$. Hence, C is positive definite, and we obtain the following theorem.

Theorem 1. *If $\forall \boldsymbol{x}$, $0 \le z_t(\boldsymbol{x}) < 1$, then the coefficient matrix C of the equation for $u_{t+\Delta t}$ is symmetric positive definite.*

Properties of the coefficient matrix for $\tilde{z}_{t+\Delta t}$

For the phase field variable $\tilde{z}_{t+dt}$, the boundary condition consists of only the Neumann boundary condition. We therefore choose the basis functions $\{\psi_j\}_{j=1}^m$ and approximate $\tilde{z}_{t+dt}$ by the following function:

$$\hat{z}_{t+dt} = \sum_{j=1}^m b_j \psi_j. \tag{31}$$

Inserting this into (24) and choosing the test function as $w = \psi_i$ gives

$$\begin{aligned} &\sum_{i=1}^m \left[\int_\Omega \left\{ \alpha_2 \psi_i \psi_j + \Delta t \left(\epsilon\gamma(\boldsymbol{x}) \nabla\psi_i \cdot \nabla\psi_j + \psi_i \frac{\gamma(\boldsymbol{x})}{\epsilon} \psi_j + \psi_i |\nabla u_t|^2 \psi_j \right) \right\} d\boldsymbol{x} \right] b_j \\ &= \int_\Omega \psi_i (\alpha_2 z_t + dt |\nabla u_t|^2) d\boldsymbol{x}. \end{aligned} \tag{32}$$

By defining the matrix $E = (e_{ij}) \in \mathbb{R}^{m\times m}$ and the vector $\boldsymbol{f} = (f_i) \in \mathbb{R}^m$ by

$$\begin{cases} e_{ij} = \int_\Omega \left\{ \alpha_2 \psi_i \psi_j + \Delta t \left(\epsilon\gamma(\boldsymbol{x}) \nabla\psi_i \cdot \nabla\psi_j + \psi_i \frac{\gamma(\boldsymbol{x})}{\epsilon} \psi_j + \psi_i |\nabla u_t|^2 \psi_j \right) \right\} d\boldsymbol{x}, & (33) \\ f_i = \int_\Omega \psi_i (\alpha_2 z_t + \Delta t |\nabla u_t|^2) d\boldsymbol{x} & (34) \end{cases}$$

and letting $\boldsymbol{b} = (b_1, b_2, \ldots, b_m)^\top$, we have the following linear simultaneous equation.

$$E\boldsymbol{b} = \boldsymbol{f}. \tag{35}$$

Since $e_{ij} = e_{ji}$ from (33), E is symmetric. Furthermore, the first term of e_{ij}, which is $\int_\Omega \alpha_2 \psi_i \psi_j \, dx$, is a Gram matrix and is therefore positive definite if $\{\psi_i\}$ is linearly independent. It is also clear that the remaining parts of e_{ij} are also symmetric positive semidefinite. Thus, we arrive at the following theorem.

Theorem 2. *The coefficient matrix E of the equation for $\tilde{z}_{t+\Delta t}$ is symmetric positive definite.*

Theorems 1 and 2 ensure that the CG method preconditioned by the incomplete Cholesky decomposition can be used to solve (29) and (35).

3.2. The Three-Dimensional Case

We next consider the three-dimensional case described by (18) through (21). First, we prepare the following lemma.

Lemma 1. *Let Ω be a bounded region in the three-dimensional space, Γ its boundary, and $\boldsymbol{n}$ the outward normal vector on Γ. Additionally, let $\phi(\boldsymbol{x})$, $\psi(\boldsymbol{x})$ and $f(\boldsymbol{x})$ be scalar fields and $\boldsymbol{w}(\boldsymbol{x})$ be a vector field defined in a region containing Ω. Then, the following equations hold:*

$$\int_\Omega \phi \nabla \cdot (f\boldsymbol{w}) d\boldsymbol{x} = \int_\Gamma \phi f \boldsymbol{w} \cdot \boldsymbol{n} \, dS - \int_\Omega f \nabla \phi \cdot \boldsymbol{w} \, d\boldsymbol{x}, \tag{36}$$

$$\int_\Omega \phi \nabla \cdot (f \nabla \psi) d\boldsymbol{x} = \int_\Gamma \phi f \nabla \psi \cdot \boldsymbol{n} \, dS - \int_\Omega f \nabla \phi \cdot \nabla \psi \, d\boldsymbol{x}, \tag{37}$$

where $\int_\Gamma \cdot \, dS$ denotes the surface integral on Γ.

Proof. First, we integrate both sides of the identity

$$\nabla \cdot (\phi f \boldsymbol{u}) = \phi \nabla \cdot (f \boldsymbol{u}) + f \nabla \phi \cdot \boldsymbol{u} \tag{38}$$

over Ω and apply Gauss's theorem to the left-hand side to transform it to $\int_\Gamma \phi f \boldsymbol{w} \cdot \boldsymbol{n} \, dS$. By moving the terms, we have (36). Then, we obtain (37) by letting $\boldsymbol{w} = \nabla \psi$. □

The weak forms

We derive the weak form for $\boldsymbol{u}_{t+\Delta t}$ by considering a vector test function $\boldsymbol{v}(\boldsymbol{x})$ that becomes $\boldsymbol{0}$ on Γ_D, computing its inner product with both sides of (21), assuming $\alpha_1 = 0$, and integrating the results over Ω (this is equivalent to deriving a weak form for each component of (18) by multiplying it by a scalar test function and integrating the result over Ω. This is because if we choose a special vector test function with its y and z components identical to zero, we obtain the same result as if we multiply the x component of (18) by a

scalar test function). By letting $\phi = v_q$, $f = (1 - z_t)^2$ and $\boldsymbol{w} = \boldsymbol{s}_{t+\Delta t,q}$ $(q = 1, 2, 3)$ in (36) and summing both sides over q, we have

$$\begin{aligned}
0 &= \sum_{q=1}^{3} \int_\Omega v_q \nabla \cdot \left((1 - z_t)^2 \boldsymbol{s}_{t+\Delta t,q} \right) d\boldsymbol{x} \\
&= \sum_{q=1}^{3} \int_\Gamma v_q (1 - z_t)^2 \boldsymbol{s}_{t+\Delta t,q} \cdot \boldsymbol{n} \, dS - \sum_{q=1}^{3} \int_\Omega (1 - z_t)^2 \nabla v_q \cdot \boldsymbol{s}_{t+\Delta t,q} \, d\boldsymbol{x} \\
&= -\sum_{j=1}^{3} \sum_{q=1}^{3} \int_\Omega (1 - z_t)^2 \frac{\partial v_q}{\partial x_j} (\sigma_{t+\Delta t})_{jq} \, d\boldsymbol{x} \\
&= -\sum_{j=1}^{3} \sum_{q=1}^{3} \int_\Omega (1 - z_t)^2 \frac{\partial v_q}{\partial x_j} \left(\lambda \delta_{jq} \nabla \cdot \boldsymbol{u}_{t+\Delta t} + 2\mu (\epsilon_{t+\Delta t})_{jq} \right) d\boldsymbol{x} \\
&= -\int_\Omega \lambda (1 - z_t)^2 (\nabla \cdot \boldsymbol{v})(\nabla \cdot \boldsymbol{u}_{t+\Delta t}) \, d\boldsymbol{x} \\
&\qquad -2 \sum_{j=1}^{3} \sum_{q=1}^{3} \int_\Omega \mu (1 - z_t)^2 \frac{\partial v_q}{\partial x_j} (\epsilon_{t+\Delta t})_{jq} \, d\boldsymbol{x},
\end{aligned} \tag{39}$$

where, in the second equality, we use the fact that $v_q = 0$ on Γ_D and $\boldsymbol{s}_{t+\Delta t,q} \cdot \boldsymbol{n} = 0$ on Γ_N and therefore the surface integral on Γ vanishes. By equating the right-hand side of (39) to zero, we obtain the weak form for $\boldsymbol{u}_{t+\Delta t}$.

The weak form for $\tilde{z}_{t+\Delta t}$ is exactly the same as (24) for the two-dimensional case, except that $|\nabla u_t|^2$ is replaced with $2e(\boldsymbol{u}_t)$.

Properties of the coefficient matrix for $u_{t+\Delta t}$

For finite element discretization of the weak form for $\boldsymbol{u}_{t+\Delta t}$, we approximate each component $u_{t+dt,j}$ $(j = 1, 2, 3)$ of $\boldsymbol{u}_{t+dt}$ as a linear combination of a function $\phi_{j,0}$ satisfying the boundary condition on Γ_D and basis functions $\{\phi_{j,\ell}\}_{\ell=1}^{m}$ that become zero on Γ_D:

$$\hat{u}_{t+dt,j} = \phi_{j,0} + \sum_{\ell=1}^{m} a_{j,\ell} \phi_{j,\ell} \quad (k = 1, 2, 3). \tag{40}$$

Now, we choose as the test function $\boldsymbol{v}$ a function whose ith element is $\phi_{i,k}$ and whose other elements are 0. Thus, $v_q = \delta_{qi} \phi_{i,k}$. Inserting this along with (40) and the definition of ϵ_{ij} into the weak form for $\boldsymbol{u}_{t+\Delta t}$ gives

$$\begin{aligned}
0 &= -\int_\Omega \lambda (1 - z_t)^2 \frac{\partial \phi_{i,k}}{\partial x_i} \sum_{j=1}^{3} \left(\frac{\partial \phi_{j,0}}{\partial x_j} + \sum_{\ell=1}^{m} a_{j,\ell} \frac{\partial \phi_{j,\ell}}{\partial x_j} \right) d\boldsymbol{x} \\
&\quad - \sum_{j=1}^{3} \int_\Omega \mu (1 - z_t)^2 \frac{\partial \phi_{i,k}}{\partial x_j} \left(\frac{\partial \phi_{j,0}}{\partial x_i} + \sum_{\ell=1}^{m} a_{j,\ell} \frac{\partial \phi_{j,\ell}}{\partial x_i} + \frac{\partial \phi_{i,0}}{\partial x_j} + \sum_{\ell=1}^{m} a_{i,\ell} \frac{\partial \phi_{i,\ell}}{\partial x_j} \right) d\boldsymbol{x}.
\end{aligned}$$

By moving the terms containing the unknowns $\{a_{j,\ell}\}$ to the left-hand side and the other terms to the right-hand side, we have

$$\begin{aligned}
&\int_\Omega \lambda(1-z_t)^2 \frac{\partial \phi_{i,k}}{\partial x_i} \sum_{j=1}^{3}\sum_{\ell=1}^{m} a_{j,\ell}\frac{\partial \phi_{j,\ell}}{\partial x_j}\,d\boldsymbol{x} \\
&\quad + \sum_{j=1}^{3}\int_\Omega \mu(1-z_t)^2 \frac{\partial \phi_{i,k}}{\partial x_j}\left(\sum_{\ell=1}^{m} a_{j,\ell}\frac{\partial \phi_{j,\ell}}{\partial x_i} + \sum_{\ell=1}^{m} a_{i,\ell}\frac{\partial \phi_{i,\ell}}{\partial x_j}\right) d\boldsymbol{x} \\
&= -\int_\Omega \lambda(1-z_t)^2 \frac{\partial \phi_{i,k}}{\partial x_i} \sum_{j=1}^{3}\frac{\partial \phi_{j,0}}{\partial x_j}\,d\boldsymbol{x} \\
&\quad - \sum_{j=1}^{3}\int_\Omega \mu(1-z_t)^2 \frac{\partial \phi_{i,k}}{\partial x_j}\left(\frac{\partial \phi_{j,0}}{\partial x_i} + \frac{\partial \phi_{i,0}}{\partial x_j}\right) d\boldsymbol{x}
\end{aligned} \tag{41}$$

Equations (39) for $i = 1, 2, 3$ and $k = 1, 2, \ldots, m$ constitute linear simultaneous equations of order $3m$ in $\{a_{j,\ell}\}$. Let us write this equation as

$$C\boldsymbol{a} = \boldsymbol{d}, \tag{42}$$

where C is a $3m \times 3m$ coefficient matrix, $\boldsymbol{a}$ is a $3m$-dimensional unknown vector and $\boldsymbol{d}$ is a $3m$-dimensional right-hand side vector. To investigate the positive definiteness of C, we compute $\boldsymbol{p}^\top C\boldsymbol{p}$, where $\boldsymbol{p}$ is a nonzero $3m$-dimensional vector. To this end, we replace $a_{j,\ell}$ with $p_{j,\ell}$ in the left-hand side of (41), multiply the result with $p_{i,k}$ and sum over i and k. Then, after some calculations, we obtain

$$\begin{aligned}
\boldsymbol{p}^\top C\boldsymbol{p} &= \int_\Omega \lambda(1-z_t)^2 \left(\sum_{i=1}^{3}\sum_{k=1}^{m} p_{i,k}\frac{\partial \phi_{i,k}}{\partial x_i}\right)\left(\sum_{j=1}^{3}\sum_{\ell=1}^{m} p_{j,\ell}\frac{\partial \phi_{j,\ell}}{\partial x_j}\right) d\boldsymbol{x} \\
&\quad + \frac{1}{2}\sum_{i=1}^{3}\sum_{j=1}^{3}\int_\Omega \mu(1-z_t)^2 \left(\sum_{k=1}^{m} p_{i,k}\frac{\partial \phi_{i,k}}{\partial x_j} + \sum_{k=1}^{m} p_{j,k}\frac{\partial \phi_{j,k}}{\partial x_i}\right) \\
&\qquad \times \left(\sum_{\ell=1}^{m} p_{i,\ell}\frac{\partial \phi_{i,\ell}}{\partial x_j} + \sum_{\ell=1}^{m} p_{j,\ell}\frac{\partial \phi_{j,\ell}}{\partial x_i}\right) d\boldsymbol{x} \\
&= \int_\Omega \lambda(1-z_t)^2 \left(\sum_{i=1}^{3}\sum_{k=1}^{m} p_{i,k}\frac{\partial \phi_{i,k}}{\partial x_i}\right)^2 d\boldsymbol{x} \\
&\quad + \frac{1}{2}\sum_{i=1}^{3}\sum_{j=1}^{3}\int_\Omega \mu(1-z_t)^2 \left(\sum_{k=1}^{m} p_{i,k}\frac{\partial \phi_{i,k}}{\partial x_j} + \sum_{k=1}^{m} p_{j,k}\frac{\partial \phi_{j,k}}{\partial x_i}\right)^2 d\boldsymbol{x}.
\end{aligned} \tag{43}$$

It is clear from the expression in the middle that C is symmetric, since interchanging (i,k) and (j,ℓ) leaves it invariant. Furthermore, if $\forall \boldsymbol{x}$, $0 \le z_t(\boldsymbol{x}) < 1$, we have $\boldsymbol{p}^\top C\boldsymbol{p} \ge 0$ from the last expression, so C is also positive semidefinite.

Finally, we investigate whether $\boldsymbol{p}^\top C\boldsymbol{p} = 0$ can occur for some $\boldsymbol{p} \neq \boldsymbol{0}$. By writing $w_i = \sum_{k=1}^{m} p_{i,k}\phi_{i,k}$ and $\boldsymbol{w} = (w_1, w_2, w_3)^\top$, we can rewrite (43) as

$$\boldsymbol{p}^\top C\boldsymbol{p} = \int_\Omega \lambda(1-z_t)^2(\nabla\cdot\boldsymbol{w})^2 d\boldsymbol{x} + \frac{1}{2}\sum_{i=1}^{3}\sum_{j=1}^{3}\int_\Omega \mu(1-z_t)^2\left(\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}\right)^2 d\boldsymbol{x}. \tag{44}$$

For the right-hand side to be zero, under the assumption that $\forall \boldsymbol{x}$, $0 \le z_t(\boldsymbol{x}) < 1$, $\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}$ must be identically zero for $i, j = 1, 2, 3$. This means that the strain tensor computed from $\boldsymbol{w}$ can have only a rotational component. However, if the target solid is fixed at three or more points, no such rotation is possible, and therefore the vector field $\boldsymbol{w}$ must be identical to zero, implying that $\boldsymbol{p} = \boldsymbol{0}$. Thus, we arrive at the following theorem.

Theorem 3. *If* $\forall \boldsymbol{x}$, $0 \leq z_t(\boldsymbol{x}) < 1$, *then the coefficient matrix C of the equation for* $u_{t+\Delta t}$ *is symmetric positive definite.*

Properties of the coefficient matrix for $\tilde{z}_{t+\Delta t}$

The weak form for $\tilde{z}_{t+\Delta t}$ in the three-dimensional case is identical to that of the two-dimensional case except that $|\nabla u_t|^2$ is replaced with $2e(\boldsymbol{u}_t)$. Since $e(\boldsymbol{u}_t)$ is nonnegative as well as $|\nabla u_t|^2$, by repeating the same argument that led to Theorem 2, we obtain the following theorem:

Theorem 4. *The coefficient matrix of the equation for* $\tilde{z}_{t+\Delta t}$ *is symmetric positive definite.*

So far, we have assumed that both λ and μ are constants over Ω. However, a close examination of the derivation of Theorems 3 and 4 reveals that they are valid even when λ and μ are continuous functions of $\boldsymbol{x}$. In some applications, $\lambda(\boldsymbol{x})$ and $\mu(\boldsymbol{x})$ might have discontinuities. In such a case, we can approximate them with smooth functions of $\boldsymbol{x}$, by using sufficiently fine mesh around the discontinuities.

4. Application of the Incomplete Cholesky Preconditioner and Its Parallelization

Now that we have established that the coefficient matrices are symmetric positive definite, we can apply the IC preconditioner, which is more effective than diagonal scaling. In this section, we first describe the IC conditioner briefly and then explain how to parallelize it using the block multicolor ordering proposed by Iwashita et al.

4.1. The Incomplete Cholesky Preconditioner

Let $A = (a_{ij}) \in \mathbb{R}^{n\times n}$ be a sparse symmetric positive definite matrix and consider solving a linear simultaneous equation $A\boldsymbol{x} = \boldsymbol{b}$ by the conjugate gradient (CG) method. In the *preconditioned* conjugate gradient method, one applies the CG method to the modified equation $(K^{-1}AK^{-\top})(K^{\top}\boldsymbol{x}) = K^{-1}\boldsymbol{b}$, whose coefficient matrix $K^{-1}AK^{-\top}$ is again SPD. Here, $K \in \mathbb{R}^{n\times n}$ is a nonsingular matrix designed so that $K^{-1}AK^{-\top}$ has a smaller condition number than A. Thus, the convergence of the CG method is accelerated. The diagonal scaling preconditioner uses $\mathrm{diag}(\sqrt{a_{11}}, \sqrt{a_{22}}, \ldots, \sqrt{a_{nn}})$ as K. This preconditioner is simple and applicable to a wide class of matrices but not as effective in reducing the number of iterations of the CG method.

In this study, we use a more powerful preconditioner based on the incomplete Cholesky decomposition without fill-ins (the IC(0) decomposition). In this decomposition, we compute the Cholesky decomposition of A approximately by allowing the element $\tilde{l}_{ij}$ of the approximate Cholesky factor $\tilde{L}$ to be nonzero only when $a_{ij} \neq 0$. Thus, fill-ins in the Cholesky decomposition are suppressed, and the computational cost and the memory requirement are reduced. The algorithm of the IC(0) decomposition is shown as Algorithm 1. Here, in the sums $\sum_{k=1}^{j-1} \tilde{l}_{jk}^2$ and $\sum_{k=1}^{j-1} \tilde{l}_{ik}\tilde{l}_{jk}$, zero terms are skipped to reduce the computational work.

Algorithm 1: IC(0) decomposition

1: **for** $j = 1$ to n **do**
2: $\quad \tilde{l}_{jj} = (a_{jj} - \sum_{k=1}^{j-1} \tilde{l}_{jk}^2)^{1/2}$
3: $\quad$ **for** $i = j+1$ to n if $a_{ij} \neq 0$ **do**
4: $\quad\quad \tilde{l}_{ij} = (a_{ij} - \sum_{k=1}^{j-1} \tilde{l}_{ik}\tilde{l}_{jk})/l_{jj}$
5: $\quad$ **end for**
6: **end for**

In the incomplete Cholesky preconditioner, $\tilde{L}$ computed by Algorithm 1 is used as K. While $\tilde{L}$ satisfies only an approximate relation $A \simeq \tilde{L}\tilde{L}^{\top}$, it is often a sufficiently good

approximation to the true Cholesky factor to make $K^{-1}AK^{-\top}$ much better conditioned than A.

For a sparse matrix A arising in the finite element method, a simplified IC(0) decomposition is sometimes used instead of Algorithm 1. In this variant, the off-diagonal elements are not updated and the fourth line of Algorithm 1 is replaced with $\tilde{l}_{ij} = a_{ij}/l_{jj}$. We use this variant in this study.

4.2. Parallelization by the Block Multi-Color Ordering

The IC(0) decomposition algorithm inherits the sequential nature of the original Cholesky decomposition. Suppose that $i < j$ and $a_{ij} \neq 0$. Then, since $\tilde{l}_{ii}$ depends on $\tilde{l}_{ij}$ by line 2 of Algorithm 1 and $\tilde{l}_{ij}$ depends on $\tilde{l}_{jj}$ by line 4, $\tilde{l}_{ii}$ depends on $\tilde{l}_{jj}$. In the finite element method using triangular (or tetrahedral) elements and piecewise linear basis functions, the matrix element a_{ij} is nonzero if and only if the nodes i and j belongs to the same element. The dependency thus caused gives rise to a difficulty in parallelizing the IC(0) decomposition.

One of the techniques to resolve this problem is *multi-color ordering* [14]. In this ordering strategy, we assign colors $1, 2, \ldots, c$ to the nodes in such a way that nodes belonging to the same element have different colors and the number of required colors is minimal. Then, we renumber the nodes so that the nodes with color 1 are numbered first, those with color 2 are numbered next, and so on. Then, if nodes i and j have the same color, they do not belong to the same element, and therefore $a_{ij} = 0$. Thus, the computation of $\tilde{l}_{ii}$ and $\tilde{l}_{jj}$ can be done in parallel.

However, it has been pointed out that this reordering can degrade the quality of the IC(0) preconditioner, thereby increasing the number of CG iterations. Consult [15] for more details about this phenomenon. As a remedy, Iwashita et al. proposed *block multi-color ordering* [7,8], which partitions the set of nodes into blocks and applies the multi-color ordering to the blocks rather than to the individual nodes. It is known that this modification is effective in retaining the quality of the IC(0) preconditioner, since the ordering within each block can be made the same as the natural ordering. The block multi-color ordering applied to a two-dimensional triangular mesh and the nonzero pattern of the resulting matrix are depicted in Figures 2 and 3, respectively. Here, $c = 4$, and thus the matrix has a 4×4 block structure. Each of the four diagonal blocks has a 2×2 block diagonal structure, reflecting the fact that there are two blocks of each color. Thus, the IC(0) decomposition of these two (small) diagonal blocks can be performed in parallel. We use this ordering strategy in our numerical experiments.

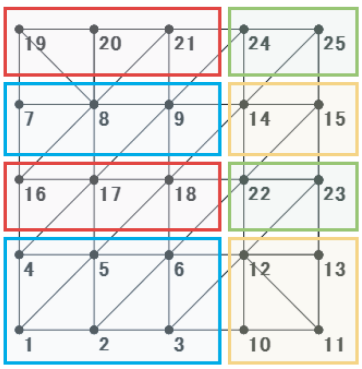


Figure 2. Block multi-color ordering for a two-dimensional triangular mesh.

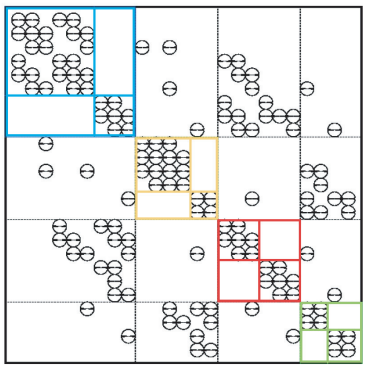


Figure 3. The coefficient matrix ordered by block multi-color ordering.

5. Numerical Results

In this section, we apply the conjugate gradient method with the IC(0) preconditioner to the linear simultaneous equations to compute the time evolution of $\boldsymbol{u}(\boldsymbol{x}, t)$ and $z(\boldsymbol{x}, t)$ in the phase field-based crack growth simulation. We parallelize the IC(0) preconditioner using the block red-black ordering and evaluate its parallel performance, as well as the convergence acceleration effect compared with the diagonal scaling preconditioner. Both two and three-dimensional problems are used as test problems.

5.1. The Two-Dimensional Case

We used the 2-D phase field-based crack growth simulation code developed by Takaishi et al. as a basis and replaced its linear equation solver, which uses the CG method with diagonal scaling, with our solver. Our solver is based on the CG method with IC(0) preconditioning and is thread-parallelized using multi-color ordering. We used four colors, as in Figure 2, and set the number of blocks equal to four times the number of threads. The program was written in C and OpenMP, and all computations were performed in double precision arithmetic. In the numerical experiments in this subsection, we used an Intel Xeon processor E5-2660 v2, which has 10 cores, and Intel C Compiler Ver. 16.0.0.109 with the -O3 option.

The computational region Ω used in the numerical experiments is as shown in Figure 1. It is a square region (panel) with the initial crack Σ running from the left edge to the center. The Dirichlet boundary conditions, which represent the forces to widen the crack, are applied to the upper and lower edges. More specifically, the problem is defined as follows.

- Computational region: $\Omega = [-1, 1] \times [-1, 1]$, $\Gamma = \Gamma_D + \Gamma_N$.
- Dirichlet boundary: $\Gamma_D = \{(x_1, x_2) \,|\, x_1 \in [-1, 1], x_2 = \pm 1\}$.
- Neumann boundary: $\Gamma_N = \{(x_1, x_2) \,|\, x_1 = \pm 1, x_2 \in [-1, 1]\}$.
- Time step: $\Delta t = 0.05$.
- Parameters: $\alpha_1 = 0, \alpha_2 = 10^{-3}, \gamma = 0.5, \varepsilon = 10^{-3}$.
- Initial conditions: $\boldsymbol{u}(\boldsymbol{x}, 0) = \boldsymbol{0}$, $z(\boldsymbol{x}, 0) = \xi(x_1 + 0.5, x_2)$, where $\xi(x_1, x_2) = \exp(-(x_2/\delta)^2)/(1 + \exp(x_1/\delta))$.
- Convergence criterion of the CG method: relative residual $\leq 10^{-10}$.

We first checked the positive definiteness of the coefficient matrix C for $\boldsymbol{u}(\boldsymbol{x}, t)$ (see (27)) using a small problem with 11×11 to 50×50 mesh. Note that the matrix E for $z(\boldsymbol{x}, t)$ (see (33)) is obviously positive definite because it is an $O(\Delta t)$ perturbation of the positive definite Gram matrix. It was confirmed by the numerical experiment that the smallest eigenvalue of C is always positive, and thus the matrix C is positive definite. The smallest eigenvalue $\lambda_{\min}$ and the largest eigenvalue $\lambda_{\max}$ of C for each mesh before, during and after crack growth are shown in Table 1. We also show the change of the condition number of C as the crack grows in Figure 4. It can be seen that the condition number leaps up suddenly around $t = 35$, where the crack grows rapidly. It seems that this is because the

area of $z(\boldsymbol{x},t) \simeq 1$ is widened, causing the near-singularity of C, as can be inferred from (30).

Table 1. The smallest and the largest eigenvalues of C for each mesh.

Point	11×11		30×30		50×50	
	$\lambda_{\min}$	$\lambda_{\max}$	$\lambda_{\min}$	$\lambda_{\max}$	$\lambda_{\min}$	$\lambda_{\max}$
Before growth	1.000	493.486	0.182	520.190	0.063	522.691
During growth	0.544	424.596	0.070	428.934	0.015	504.204
After growth	0.376	424.558	0.065	428.934	0.014	504.204

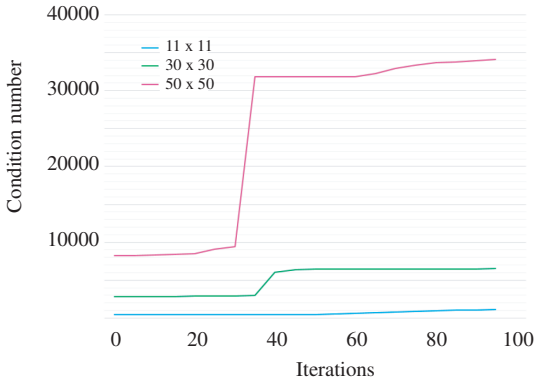
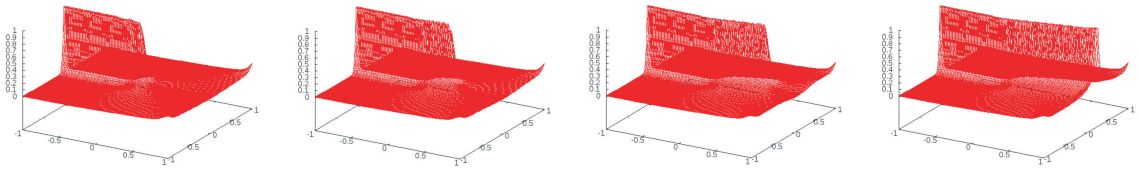


Figure 4. Time evolution of the condition number of C.

The time evolution of $z(\boldsymbol{x},t)$ for this problem is shown in Figure 5. Here, the mesh size is 101×101 and the period of simulation is from $t = 0$ to $t = 34.5$. Until $t = 28.5$, $z(\boldsymbol{x},t)$ does not change significantly and $z(\boldsymbol{x},t) \simeq 1$ only along the line connecting $(-1,0)$ and $(0,0)$, showing that the crack exists only in this region. As time passes, the region of $z(\boldsymbol{x},t) \simeq 1$ extends to the right edge of the region, meaning that the panel has broken into two pieces. The evolution of $z(\boldsymbol{x},t)$ for other initial conditions, which was computed using a FreeFEM code, is shown in Appendix A.



(a) t = 28.5 (b) t = 30.5 (c) t = 32.5 (d) t = 34.5

Figure 5. Time evolution of the phase-field variable $z(\boldsymbol{x},t)$.

We next evaluated the parallel acceleration of our linear equation solver by varying the number of threads from 1 to 10. The results for 101×101 and 201×201 meshes are shown in Figures 6 and 7, respectively. The number of blocks in the x_1 and x_2 directions, which we denote by nbx and nby, are shown in Table 2. These numbers were determined experimentally to achieve the best performance in each case, under the condition that $nbx \times nby$ equals four times the number of threads. It can be seen that the solution of the linear simultaneous equations for $\boldsymbol{u}$ and z achieves an acceleration of up to 5 and 4 times, respectively, using 10 threads.

Table 2. Optimal combinations of *nbx* and *nby*.

Number of Threads	101 × 101	201 × 201
1	(1, 4)	(1, 4)
2	(2, 4)	(2, 4)
3	(2, 6)	(2, 6)
4	(2, 8)	(2, 8)
5	(2, 10)	(2, 10)
6	(4, 6)	(4, 6)
7	(14, 2)	(14, 2)
8	(4, 8)	(4, 8)
9	(6, 6)	(6, 6)
10	(4, 10)	(4, 10)

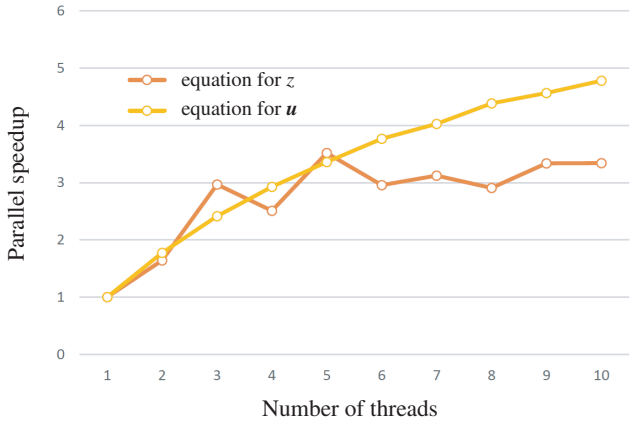


Figure 6. Parallel acceleration for the 101 × 101 mesh.

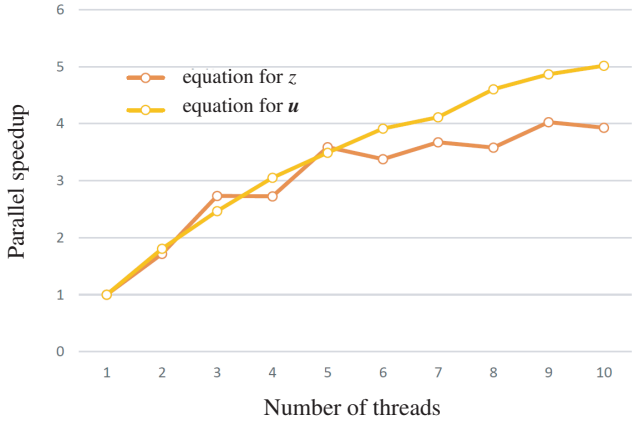


Figure 7. Parallel acceleration for the 201 × 201 mesh.

Finally, we compare the execution time of our solver with that of the original solver using the diagonal scaling preconditioner. The results are shown in Figures 8 and 9. Compared with the original solver, our solver achieved an acceleration of 7.2 and 7.4 times for the 101 × 101 and 201 × 201 meshes, respectively.

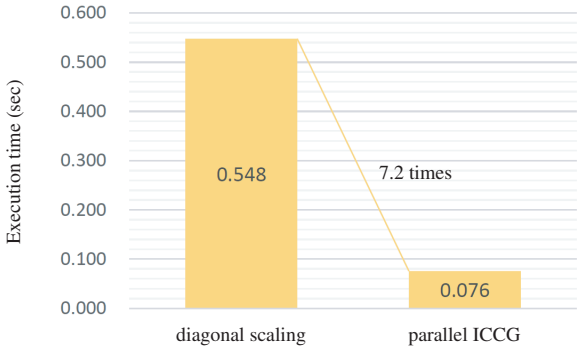


Figure 8. Execution time for the 101 × 101 mesh.

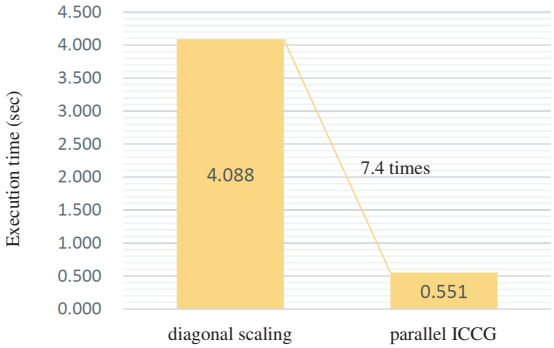


Figure 9. Execution time for the 201 × 201 mesh.

5.2. The Three-Dimensional Case

For the three-dimensional case, we again used Takaishi et al.'s code and replaced its linear equation solver with our parallel CG solver with IC(0) preconditioning and multi-color ordering. We used a semi-structured mesh as shown in Figure 10, which is unstructured in the (x_1, x_2) plane and is structured in the x_3 direction, sliced it horizontally into 2× (number of threads) panels, and colored them in an alternating manner using two colors. Then, each pair of panels was allocated to one thread for parallel execution. In the numerical experiments in this subsection, we used an Intel Xeon Gold 6148 Processor with 20 cores (1 node of the Grand Chariot supercomputer at the Hokkaido University Information Initiative Center) and Intel C Compiler Ver. 18.0.3 with the -O3 option.

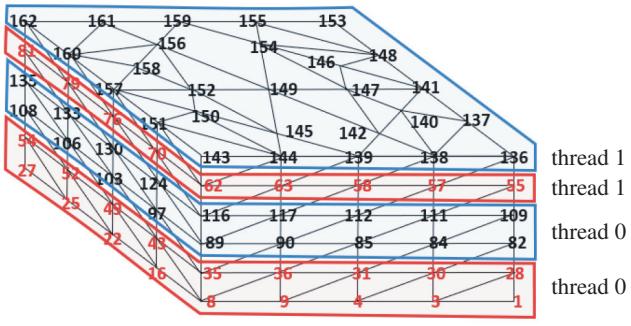


Figure 10. Semi-structured mesh for the 3-D simulation and its ordering.

The test problem is crack growth in a rectangular parallelepiped region, defined as follows.

- Computational region: $\Omega = [-1,1] \times [-1,1] \times [-0.5,0.5]$, $\Gamma = \Gamma_D^{(1)} + \Gamma_D^{(2)} + \Gamma_N$;
- Dirichlet boundary: $\Gamma_D^{(1)} = \{(x_1,x_2,x_3) \,|\, x_1 \in [-1,1], x_2 \in [-1,1], x_3 = \pm 0.5\}$;
- Dirichlet boundary: $\Gamma_D^{(2)} = \{(x_1,x_2,x_3) \,|\, x_1 \in [-1,1], x_2 = \pm 1, x_3 \in [-0.5,0.5]\}$;
- Neumann boundary: $\Gamma_N = \{(x_1,x_2,x_3) \,|\, x_1 = \pm 1, x_2 \in [-1,1], x_3 \in [-0.5,0.5]\}$;

- Time step: $\Delta t = 0.05$;
- Parameters: $\alpha_1 = 0, \alpha_2 = 10^{-3}, \gamma = 0.5, \varepsilon = 10^{-3}$;
- Initial conditions: $\boldsymbol{u}(\boldsymbol{x}, 0) = \boldsymbol{0}$, $z(\boldsymbol{x}, 0) = \xi(x_1 + 0.5, x_2)$, where $\xi(x_1, x_2) = \exp(-(x_2/\delta)^2)/(1 + \exp(x_1/\delta))$;
- Convergence criterion of the CG method: relative residual $\leq 10^{-10}$.

The parallel acceleration of our solver for the $51 \times 51 \times 52$ and $101 \times 101 \times 102$ meshes is shown in Figures 11 and 12, respectively. For the latter mesh, the solution of the linear system for $\boldsymbol{u}$ and z was accelerated by up to 6.7 and 4.4 times, respectively. We also show the acceleration of each component of our solver and the breakdown of the execution time in Figures 13–16. Our solver consists of a matrix-vector product, forward and backward substitutions corresponding to the application of $\tilde{L}^{-1}$ and $\tilde{L}^{-\top}$, respectively, and other parts such as vector additions, dot products and computation of norms. Figures 13 and 15 show that the matrix-vector product and the forward/backward substitutions are reasonably well accelerated, but Figures 14 and 16 reveal that the other parts are not accelerated at all. The latter are difficult to parallelize efficiently because they are vector operations with small computational work and a relatively large amount of data transfer. If this part could be improved, our solver could achieve further acceleration.

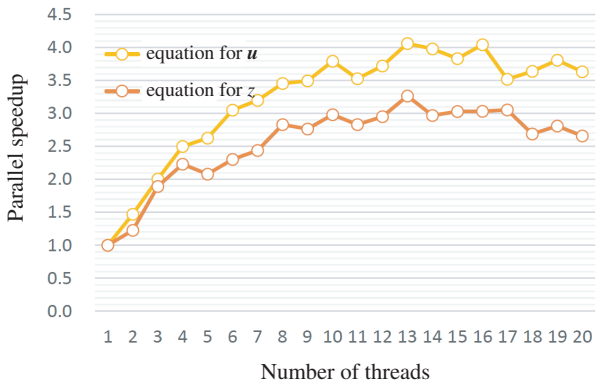


Figure 11. Parallel acceleration for the $51 \times 51 \times 52$ mesh.

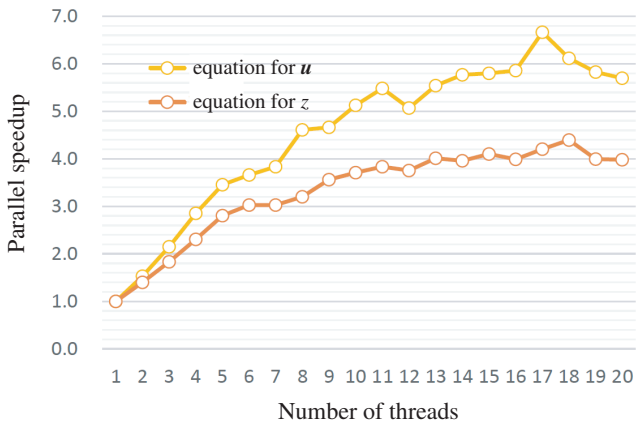


Figure 12. Parallel acceleration for the $101 \times 101 \times 102$ mesh.

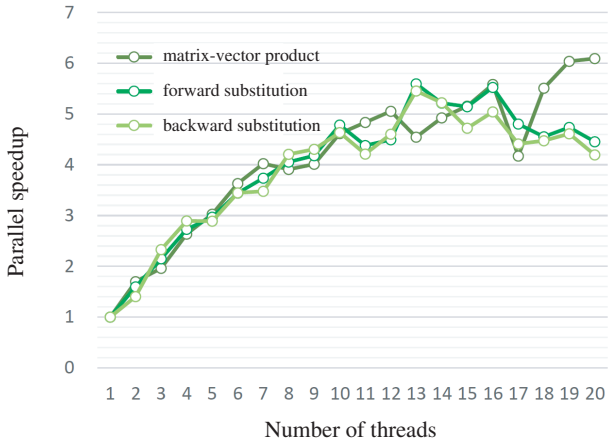


Figure 13. Parallel acceleration of each component for the 51 × 51 × 52 mesh.

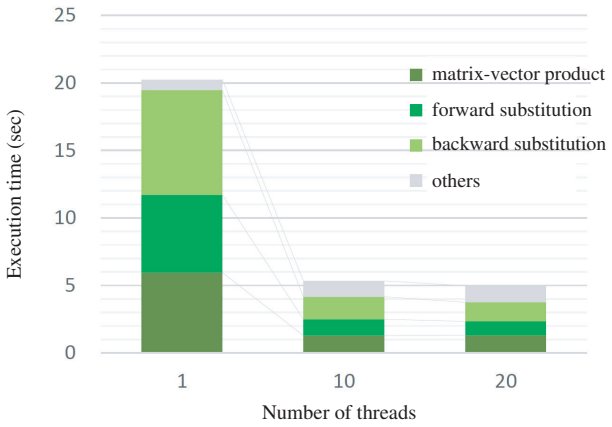


Figure 14. Breakdown of the execution time for the 51 × 51 × 52 mesh.

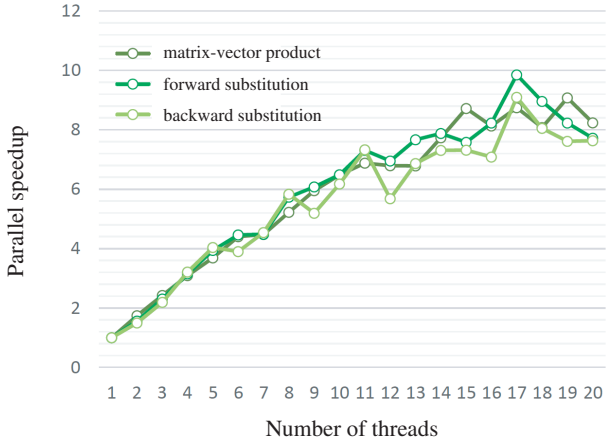


Figure 15. Parallel acceleration of each component for the 101 × 101 × 102 mesh.

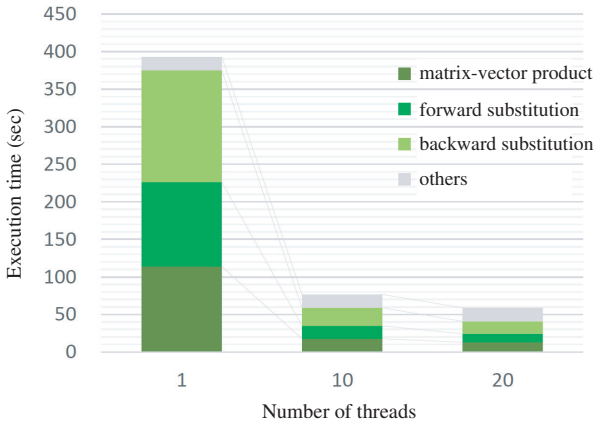


Figure 16. Breakdown of the execution time for the $101 \times 101 \times 102$ mesh.

Finally, we compare the execution time of our solver with that of the original solver in Figures 17 and 18. Thanks to the use of the IC(0) preconditioner with multi-color ordering, our solver attains an acceleration of 6.1 and 8.3 for the $51 \times 51 \times 52$ and $101 \times 101 \times 102$ meshes, respectively.

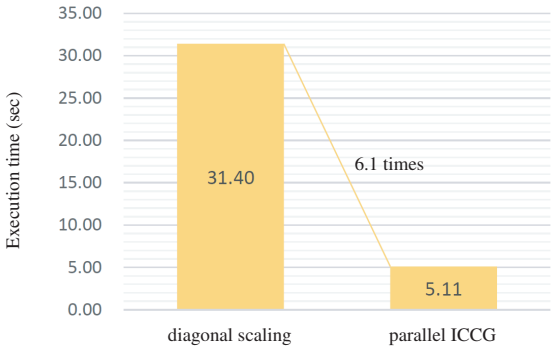


Figure 17. Execution time for the $51 \times 51 \times 52$ mesh.

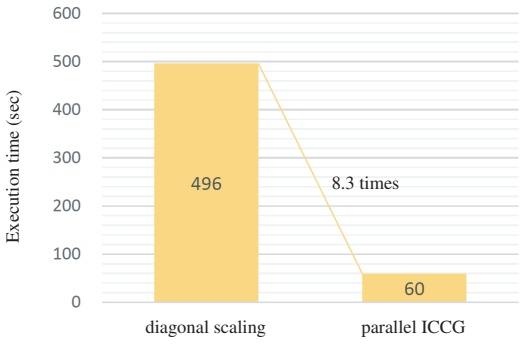


Figure 18. Execution time for the $101 \times 101 \times 102$ mesh.

6. Conclusions

In this paper, we accelerated the linear equation solution part in phase field-based crack growth simulation using the conjugate gradient method with IC(0) preconditioning. To this end, we first analyzed the properties of the coefficient matrices both for the displacement $\boldsymbol{u}(\boldsymbol{x}, t)$ and the phase field $z(\boldsymbol{x}, t)$ and proved that they are symmetric positive definite under certain conditions. Thus, the use of the IC(0) preconditioning is justified. Then, we parallelized the IC(0) preconditioner using the block multi-color ordering and evaluated its performance on multicore processors. The experimental results show that our solver scales well both for the two and three-dimensional problems and achieves an acceleration of several times over the original solver based on the diagonally scaled CG

method. Our future work will include the distributed-memory parallelization of our solver and its application to real-world crack growth problems.

Author Contributions: Conceptualization, T.T. and Y.Y.; theoretical analysis, G.I. and Y.Y.; coding, G.I. and T.T.; parallelization and optimization, G.I.; numerical experiments, G.I. All authors have read and agreed to the published version of the manuscript.

Funding: This study is partially supported by JSPS KAKENHI Grant Numbers 17H02828, 17K19966 and 19KK0255.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The parallel ICCG solver developed in this work, as well as the FreeFEM code used in the Appendix, is available from the authors upon request. Some of these programs are also downloadable from the following URL: https://github.com/yusakuyamamoto/Crack-growth-simulation, accessed on 20 August 2021 .

Acknowledgments: The authors thank Prof. Takaharu Yaguchi of Kobe University and Prof. Shuhei Kudo of The University of Electro-Communications for valuable discussion. They are also grateful to Prof. Takeshi Fukaya of Hokkaido University for providing computational environments for the three-dimensional problem. Part of our numerical experiments were performed on the Grand Chariot supercomputer at Hokkaido University Information Initiative Center.

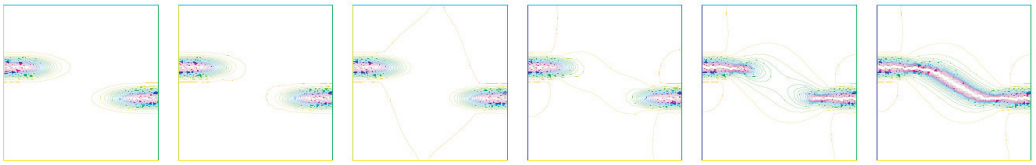
Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Two-Dimensional Crack Growth Simulation for Various Initial Conditions

Here, we present the results of the two-dimensional crack growth simulation for various initial conditions. The computational region is the same as that used in Section 5.1, and a FreeFEM [16] code was used for the simulation. The initial conditions used are as follows:

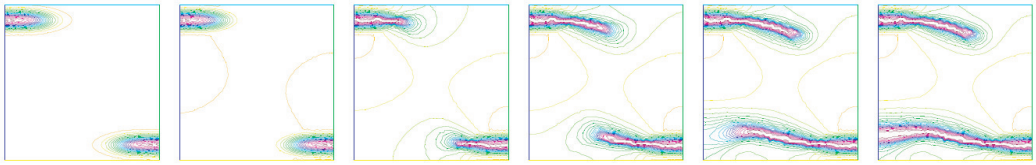
1. $z(\boldsymbol{x}, 0) = \xi(-x_1 + 0.5, x_2 + 0.2) + \xi(x_1 + 0.5, x_2 - 0.2)$
2. $z(\boldsymbol{x}, 0) = \xi(-x_1 + 0.5, x_2 + 0.8) + \xi(x_1 + 0.5, x_2 - 0.8)$
3. $z(\boldsymbol{x}, 0) = \xi(-x_1 + 0.5, x_2 + 0.2) + \xi(x_1 + 0.5, x_2 - 0.2) + \xi(-x_1 + 0.5, x_2)$

Here, the function $\xi(x)$ is as defined in Section 5.1 and the initial condition for u is the same as that used in Section 5.1. Cases 1 and 2 correspond to the case of two initial cracks: one at the left and another at the right. The vertical distance between the cracks is small for case 1 and large for case 2. Case 3 corresponds to the case of three initial cracks: two at the left and one at the right. The time evolution of $z(\boldsymbol{x}, t)$ for these cases is shown in Figures A1–A3 as contour maps. We used a 50×50 mesh for cases 1 and 2. For case 3, it turned out that this mesh is too coarse, so we used a 100×100 mesh. It can be seen that all of the simulations give physically plausible results.



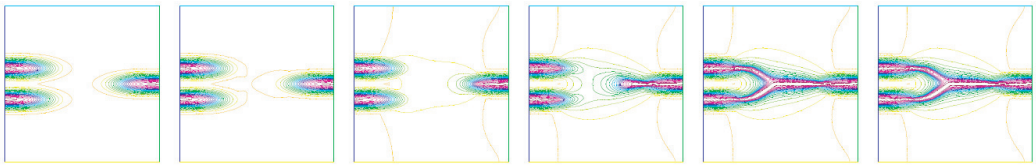
(a) t = 0 (b) t = 20 (c) t = 30 (d) t = 40 (e) t = 50 (f) t = 60

Figure A1. Time evolution of the phase-field variable $z(\boldsymbol{x}, t)$ (Case 1).



(a) t = 0 (b) t = 30 (c) t = 60 (d) t = 70 (e) t = 80 (f) t = 90

Figure A2. Time evolution of the phase-field variable $z(\boldsymbol{x}, t)$ (Case 2).



(a) t = 0 (b) t = 20 (c) t = 40 (d) t = 50 (e) t = 60 (f) t = 70

Figure A3. Time evolution of the phase-field variable $z(\boldsymbol{x}, t)$ (Case 3).

References

1. Francfort, G.A.; Marigo, J.-J. Revisiting Brittle Fracture as an Energy Minimization Problem. *J. Mech. Phys. Solids* **1998**, *46*, 1319–1342.
2. Kimura, M.; Takaishi, T.; Alfat, S.; Nakano, T.; Tanaka, Y. Irreversible phase field models for crack growth in industrial applications: Thermal stress, viscoelasticity, hydrogen embrittlement. *J. Fract. Mech.* **2021**, *3*, 781
3. Takaishi, T. and Kimura, M. Phase field model for mode III crack growth. *Kybernetika* **2009**, *45*, 605–614.
4. Takaishi, T. Numerical simulations of a phase field model for mode III crack growth. *Trans. Jpn. Soc. Ind. Appl. Math.* **2009**, *19*, 351–369. (In Japanese)
5. Kobayashi, R. Modeling and numerical simulations of dendritic crystal growth. *Phys. D* **1998**, *63*, 410–423.
6. Provatas, N.; Elder, K. *Phase-Field Methods in Materials Science and Engineering*; Wiley-VCH: Weinheim Germany 2010.
7. Iwashita, T., Nakashima, H. and Takahashi, Y. Algebraic block multi-color ordering method for parallel multi-threaded sparse triangular solver in ICCG method. In Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium, Shanghai, China, 21 May 2012 ; pp. 474–483.
8. Iwashita, T.; Li, S.; Fukaya, T. Hierarchical block multi-color ordering: A new parallel ordering method for vectorization and parallelization of the sparse triangular solver in the ICCG method. *CCF Trans. High. Perform. Comput.* **2020**, *2*, 84–97.
9. Griffith, A.A. The Phenomena of Rupture and Flow in Solids. *Phil. Trans. R. Soc. Lond.* **1921**, *A221*, 163–198.
10. Bourdin, B.; Francfort, G.A.; Marigo, J.-J. Numerical Experiments in Revisited Brittle Fracture. *J. Mech. Phys. Solids* **2000**, *48*, 797–826.
11. Ambrosio, L.; Tortorelli, V.M. On the approximation of free discountinuity problems. *Boll. Un. Mat. Ital.* **1992**, *7*, 105–123.
12. Akagi, G.; Kimura, M. Unidirectional evolution equations of diffusion type. *J. Differ. Equ.* **2019**, *266*, 1–43.
13. Visintin, A. *Models of Phase Transitions*; Birkhaeuser: Basle Switzerland 1996.
14. Saad, Y. *Iterative Methods for Sparse Linear Systems*; SIAM: Philadelphia, PA, USA , 2003.
15. Iwashita, T., Nakanishi, Y. and Shimasaki, M. Comparison criteria for parallel orderings in ILU preconditioning. *SIAM J. Sci. Comput.* **2005**, *26*, 1234–1260.
16. FreeFEM. Available online: https://freefem.org/ (accessed on 20 August 2021).