Massively parallel phase field fracture simulations on supercomputers: towards multi-billion degree-of-freedom computations

Zakaria Chafia, Julien Yvonnet, Jérémy Bleyer, Stéphane Vincent, Simon El Ouafa

Z. Chafia et al. *Advanced Modeling and Simulation in Engineering Sciences(2024)11:25*
https://doi.org/10.1186/s40323-024-00280-4

Advanced Modeling and Simulation in Engineering Sciences

RESEARCH ARTICLE Open Access

Massively parallel phase field fracture simulations on supercomputers: towards multi-billion degree-of-freedom computations

Zakaria Chafia[1,2], Julien Yvonnet[1*], Jérémy Bleyer[2], Stéphane Vincent[1] and Simon El Ouafa[1]

*Correspondence:
julien.yvonnet@univ-eiffel.fr
[1]Université Gustave Eiffel, MSME, CNRS, UMR 8208, Marne-la-Vallée, France
[2]Ecole des Ponts ParisTech, NAVIER, CNRS, UMR 8205, Champs-sur-Marne, France

Abstract

An efficient parallel implementation of the phase field method for quasi-brittle crack simulations able to run on supercomputers with a large number of processes is proposed. This framework uses the finite-element method on 3D structured meshes, and was developed for distributed memory machines using the Message Passing Interface (MPI) for workload distribution and data communication between processes. Parallel assembly is carried out to build the matrices associated with the linear systems of equations. In the proposed context, linear systems derived from displacement and damage discretizations are solved using parallel solvers and preconditioners from the PETSc (Portable, Extensible Toolkit for Scientific Computation) library. All additional operations in this implementation are also efficiently parallelized. Performance analysis shows linear acceleration with an efficiency of 97% for a computation on 6400 processes and over 80% for a computation on 10240 processes. The linear systems involved in the simulations with up to 10^{10} degrees of freedom can be solved in a few seconds. The methodology is applied to quasi-brittle simulations, involving alternate solving of large linear systems and incremental evolution. The applications presented to illustrate the parallel framework involve crack initiation and propagation in strongly heterogeneous materials with a detailed description of the microstructure. Large three-dimensional periodic structures and a realistic geometrical model directly obtained by micro-CT imagery are used.

Keywords: High performance computing, Finite-element method, Phase field, Material damage, Composite materials

Introduction

Modeling cracks in heterogeneous media is a central engineering problem for predicting the strength of existing materials based on knowledge of their microstructure, or for designing new materials with improved strength properties. Potential applications range from composites and civil engineering materials to biomaterials and new 3D-printed architectural materials. Modeling fracture in heterogeneous materials is a difficult prob-

lem, facing several scientific challenges. Among these, proposing crack propagation models that take into account the effects of microstructure, such as preferential orientations and the size effects induced by these, remains a delicate task. Another difficulty is the ratio between the characteristic length of heterogeneities and the length of a mechanical component of the structure, which makes direct simulations that take into account all the details of the sample microstructure extremely cumbersome and limited to small volumes within the structure.

Several models and methods have been developed to study crack initiation and propagation [1–6]. In recent years, the phase field method [7–11] for quasi-fragile failure, which is the numerical method used in this work, has established itself as a powerful computational tool for studying the failure behavior of structures. This approach has the ability to model crack nucleation, coalescence and branching simply and efficiently.

This method has become a widely used tool for describing various cracking phenomena, and has proven its effectiveness in modeling material damage under different loading conditions, thus providing a better understanding of the fracture behavior of materials and structures. The phase field method has been widely developed and applied to numerous problems, such as ductile fracture [12], dynamic fracture [13,14], cohesive fracture [15, 16], hydraulic fracture [17], anisotropic damage [18,19], topology optimization for crack resistance [20], fracture in microtomography image-based microstructure models [21–23], among many others.

Despite its effectiveness in simulating crack initiation and propagation, a major drawback of the phase field method is the high computational costs involved when using the finite-element method. Using a finer mesh and smaller loading steps to obtain accurate results leads to very expensive simulations in terms of computing resources. One of the current challenges is to develop methods and models capable of modeling the damage of three-dimensional structures on a macroscopic scale, while taking into account local heterogeneities and a realistic representation of the microstructure.

Several methods have been developed in the literature to meet these challenges, including multi-scale methods [24–28] or computational homogenization methods [29–32], where the prediction of fracture behavior at the macroscopic scale implies the simultaneous simulations on representative volume elements (RVE) at both micro and macro scales. Other approaches have been developed to simulate damage at the structural scale while taking into account the fine-scale features within the framework of the phase field method, including fracture toughness and characteristic length scale calculations [33], inverse approaches [34] or asymptotic homogenization [35].

Another approach to dealing with this problem is to perform direct calculations, using parallel computing. The main idea behind these approaches is to decompose the global problem into smaller, independent sub-problems associated with sub-domains. In [36,37], FETI methods have been used to simulate structural failure using the phase field method. Other methods using parallel solvers with data distribution can be found. In [38], a parallel algorithm on a graphics processing unit (GPU) to simulate dynamic brittle fracture with the phase field method has been proposed. A parallel-adaptive framework for phase-field fracture propagation has been developped in [39–41]. In [42], a parallel matrix-free higher-order finite-element solvers has been developed to simulate the failure of two-dimensional structures using the phase field method. A parallel framework for the matrix-free solution to a monolithic quasi-static phase field fracture model with geometric

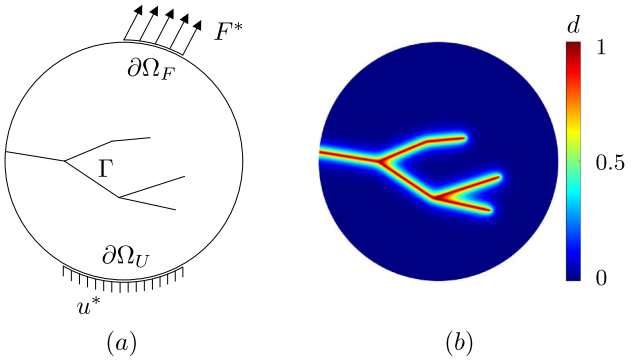

Fig. 1 Regularized representation of a crack: two-dimensional case: **a** sharp crack model; **b** regularized representation through phase field

multigrid methods has been proposed in [43]. In [44], the ParaFEM library, which is a parallel finite-element analysis library, has been used to simulate brittle fracture in three-dimensional polycrystalline structures with the CASUP package under dynamic loading. In [45], a hybrid MPI/OpenMP strategy has been used to simulate damage of three-dimensional structures with the phase field method using the matrix-free framework.

The present work proposes an efficient in-house developed parallel implementation of the phase field method, for modeling fracture behavior in large three-dimensional heterogeneous quasi-brittle structures. The implementation is based on high-performance computing using distributed memory supercomputers. Crack nucleation and propagation are simulated using the phase field method within the framework of the finite-element method using assembled matrices and vectors. In this context, the workload, which is the amount of operations performed by the processes, is well balanced, and data is communicated efficiently between processes using the MPI library. The parallelization is not limited to the linear solvers, but is applied to all operations involved in the alternate solver used for the coupled mechanical and phase field problems. Linear systems of damage and displacement problems are solved using parallel solvers and preconditioners from the PETSc library. This in-house developed parallel implementation is highly optimized in terms of memory allocation, and despite the use of assembled matrices and vectors to solve the linear systems, few memory resources are used.

This paper is organized as follows. The phase field method is briefly reviewed in "Phase field method for brittle fracture" section, the parallel implementation of the phase field method is presented in "Parallel implementation" section , the validation of the proposed implementation and its performance evaluation are carried out in "Validation and performance evaluation" section. Finally, numerical applications of crack initiation and propagation using three-dimensional periodic structures and highly heterogeneous material models obtained by micro-CT imagery are presented in "Numerical applications" section.

Phase field method for brittle fracture

In this work, the phase field method for brittle material fracture is briefly reviewed. Let us consider an open domain $\Omega \subset \mathbb{R}^D$ describing a cracked solid of dimension D and boundary $\partial\Omega$ of dimension $D-1$. Let Γ be the crack surface of dimension $D-1$ and let $\partial\Omega_U$ and $\partial\Omega_F$ be the parts of the boundary on which Dirichlet and Neumann boundary conditions are prescribed, respectively, such that $\partial\Omega = \partial\Omega_U \cup \partial\Omega_F$ and $\partial\Omega_U \cap \partial\Omega_F = \emptyset$

(see Fig. 1a). Considering the variational framework proposed in [7,8], the total energy functional $\mathcal{E}(u, d)$ is given as:

$$\mathcal{E}(u,d) = \int_{\Omega} \psi_e(\varepsilon(u), d)\, \mathrm{d}\Omega + \int_{\Omega} \psi_f(d)\, \mathrm{d}\Omega - \int_{\Omega} f^* \cdot u\, \mathrm{d}\Omega - \int_{\partial\Omega_F} F^* \cdot u\, \mathrm{d}S, \tag{1}$$

where u is the displacement field, $\varepsilon(u) = \frac{1}{2}\left(\nabla(u) + \nabla(u)^T\right)$ is the strain tensor, d is the damage field or the phase field variable. This variable refers to the smeared crack which varies smoothly from $d = 0$ representing the undamaged state to $d = 1$ representing the fully damaged state (see Fig. 1b). Above, $\psi_e(u, d)$ denotes the elastic energy density and $\psi_f(u, d)$ refers to the fracture energy density, and f^* and F^* are body forces and prescribed surface forces on the boundary $\partial\Omega_F$, respectively.

A consistent damage model for brittle fracture, developed by Miehe in [10,11], is considered here. In this model, the elastic energy density is split into positive and negative parts using the spectral decomposition of the strain tensor, which is given as:

$$\psi_e(\varepsilon(u), d) = (g(d) + k)\, \psi_e^+(\varepsilon(u)) + \psi_e^-(\varepsilon(u)), \tag{2}$$

in which $g(d) = (1 - d)^2$ is the degradation function applied only on the positive part of the elastic energy density to take into account, solely, degradation due to tension, k is a small positive parameter introduced to ensure the well-posedness of the system in case of fully damaged state of material, $\psi_e^+(\varepsilon(u))$ and $\psi_e^-(\varepsilon(u))$ represent the positive and negative parts of the elastic energy density expressed as:

$$\psi_e^{\pm}(\varepsilon(u)) = \frac{\lambda}{2}\left(\langle \mathrm{Tr}(\varepsilon)\rangle_{\pm}\right)^2 + \mu \mathrm{Tr}\left((\varepsilon^{\pm})^2\right), \tag{3}$$

where λ and μ are the Lamé's parameters, Tr(.) is the trace operator, ε^+ and ε^- represent the positive and the negative parts of the strain tensors, respectively, which are expressed by:

$$\varepsilon^{\pm} = \sum_{i=1}^{n} \langle \varepsilon^i \rangle_{\pm} n^i \otimes n^i, \quad n = 2,\ 3, \tag{4}$$

where $\langle x \rangle_{\pm} = \frac{1}{2}(x \pm |x|)$, ε^i and n^i are the eigenvalues and eigenvectors of the strain tensor ε.

There are several possible choices for the expression of the fracture energy density. For example, the so-called AT2 model [9] gives:

$$\psi_f(d) = \psi_c \left(d^2 + \ell_0^2 \nabla d \cdot \nabla d\right), \tag{5}$$

where $\nabla(\cdot)$ is the gradient operator, ℓ_0 is the characteristic length that governs the width of the regularization band, ψ_c is a specific fracture energy density, considered here to be a material property and can be related to the critical stress σ_c and the fracture toughness G_c by [10]:

$$\psi_c = \frac{\sigma_c^2}{2E} = \frac{G_c}{2\ell_0}, \tag{6}$$

where E is the Young's modulus.

To solve this minimization problem, a robust algorithm based on an alternate minimization or staggered scheme was introduced by Miehe in [9]. In fact, the total energy functional (1) is not convex with respect to both unknowns (u, d), but is convex with respect to each variable separately [46]. Accordingly, an incremental problem is obtained, where two coupled sets of equations are solved sequentially: a damage problem is solved by fixing the displacement field, and then a mechanical problem is solved by fixing the damage field d.

Damage problem

Governing equations

Given the displacement field u, the minimization of the total energy functional (1) with respect to the damage field is expressed as: find $d \in H^1(\Omega)$, such that:

$$D_{\delta d}\mathcal{E}(u,d) = \int_\Omega -2(1-d)\delta d\,\psi_e^+(\varepsilon(u)) + 2\psi_c\left(d\delta d + \ell_0^2\nabla d\cdot\nabla(\delta d)\right)\mathrm{d}\Omega = 0, \tag{7}$$

$\forall \delta d \in H^1(\Omega)$, and where $D_{\delta d}\mathcal{E}(u,d)$ is the directional derivative of $\mathcal{E}(u,d)$ in the direction of δd. The associated Euler-Lagrange equations to Eq. (7) are given by:

$$\begin{cases} (1-d)\psi_e^+ - \psi_c\left(d - {\ell_0}^2\Delta d\right) = 0 \quad \text{in} \quad \Omega, \\ d = 1 \quad \text{on} \quad \Gamma, \\ \nabla d\cdot n = 0 \quad \text{on} \quad \partial\Omega, \end{cases} \tag{8}$$

where $\Delta(\cdot)$ is the Laplacian operator, and n is the outward-pointing normal vector on the boundary $\partial\Omega$.

The weak form of the damage problem is expressed as:

$$\int_\Omega (\psi_e^+ + \psi_c)d\delta d + \psi_c\ell_0^2\nabla d\cdot\nabla(\delta d)\,\mathrm{d}\Omega = \int_\Omega \psi_e^+\delta d\,\mathrm{d}\Omega. \tag{9}$$

To avoid non-physical self-healing phenomena of damage, an irreversibility condition is imposed on the phase field variable by an appropriate choice of the source term (the right-end term in Eq. 9). At every point in Ω, the damage variable is an increasing function of load evolution. Accordingly, a history functional is introduced in [9] which is given as:

$$\mathcal{H} = \max_{\tau\in[0,t]}\left[\left\langle\psi^+(\varepsilon;\,\tau) - \psi_c\right\rangle_+\right], \tag{10}$$

where t is a pseudo time parameter describing the load evolution.

The weak form (9) becomes:

$$\int_\Omega (\mathcal{H} + \psi_c)d\delta d + \psi_c\ell_0^2\nabla d\cdot\nabla(\delta d)\,\mathrm{d}\Omega = \int_\Omega \mathcal{H}\delta d\,\mathrm{d}\Omega. \tag{11}$$

Finite-element discretization

The damage field, the damage gradient and their variations are approximated in one element by:

$$d = N_d d_i \quad ; \quad \delta d = N_d\delta d_i \quad ; \quad \nabla d = B_d d_i \quad ; \quad \nabla(\delta d) = B_d\delta d_i, \tag{12}$$

where d_i are the nodal values of the damage field d, N_d and B_d are vectors and matrices of shape functions and of shape functions derivatives for scalar fields, respectively.

The discretization of the damage problem (11) leads to the following discrete system of equations:

$$K_d d = F_d, \tag{13}$$

in which

$$K_d = \int_\Omega (\mathcal{H} + \psi_c) N_d^T N_d + \psi_c \ell_0^2 B_d^T B_d \, \mathrm{d}\Omega, \quad F_d = \int_\Omega N_d^T \mathcal{H} \, \mathrm{d}\Omega. \tag{14}$$

Mechanical problem

Governing equations

Given the damage field d, the minimization of the total energy functional (1) with respect to the displacement field is expressed as: find $u \in \mathcal{U} = \left\{u \in H^1(\Omega) | u = u^* \quad \text{on} \quad \partial\Omega_U\right\}$ such that

$$D_{\delta u}\mathcal{E}(u, d) = \int_\Omega \sigma\left(\varepsilon(u), d\right) : \varepsilon(\delta u) \, \mathrm{d}\Omega - \int_\Omega f^* \cdot \delta u \, \mathrm{d}\Omega - \int_{\partial\Omega_F} F^* \cdot \delta u \, dS = 0, \tag{15}$$

$\forall \delta u \in \mathcal{U}^0 = \left\{\delta u \in H^1(\Omega) | \delta u = 0 \quad \text{on} \quad \partial\Omega_U\right\}$ and $\sigma\left(\varepsilon(u), d\right)$ denotes the stress tensor, which is expressed as:

$$\sigma\left(\varepsilon(u), d\right) = \left((1-d)^2 + k\right)\sigma^+\left(\varepsilon(u)\right) + \sigma^-\left(\varepsilon(u)\right), \tag{16}$$

where σ^+ and σ^- represent the positive and the negative parts of the stress tensor, respectively, given by:

$$\sigma^{\pm}(\varepsilon(u)) = \frac{\partial \psi_e^{\pm}}{\partial \varepsilon}(\varepsilon(u)) = \lambda \left\langle tr[\varepsilon(u)]\right\rangle_{\pm} I + 2\mu\varepsilon^{\pm}(u), \tag{17}$$

in which I is the second-order identity tensor.

The Euler-Lagrange equations associated to Eq. (15) are given by:

$$\begin{cases} \nabla \cdot \sigma + f = 0 \quad \text{in} \quad \Omega, \\ u = u^* \quad \text{on} \quad \partial\Omega_U, \\ \sigma \cdot n = F^* \quad \text{on} \quad \partial\Omega_F. \end{cases} \tag{18}$$

where $\nabla \cdot (\cdot)$ is the divergence operator.

To avoid the nonlinearity related to the strain tensor decomposition, two shifted strain tensor split algorithms were introduced in [21] and expressed by:

$$\varepsilon_{n+1}^{\pm} \simeq \mathcal{P}_n^{\pm} : \varepsilon_{n+1}, \tag{19}$$

in which

$$\mathcal{P}_n^{\pm} = \frac{\partial \varepsilon_n^{\pm}}{\partial \varepsilon_n}, \tag{20}$$

where $\varepsilon_{n+1}^{\pm}$ and ε_{n+1} are computed at load increment t_{n+1}. The computation of the projector tensors $\mathcal{P}_n^{\pm}$ are performed at the previous load increment t_n using the algorithm proposed in [47].

This allows us to rewrite the stress/strain relationship as follows:

$$\sigma\left(\varepsilon(u), d\right) = \mathcal{C}(d) : \varepsilon(u), \tag{21}$$

where

$$\mathcal{C}(d) = \left((1-d)^2 + k\right)\left[\lambda \mathcal{R}^+[1][1]^T + 2\mu\mathcal{P}^+\right] + \left[\lambda\mathcal{R}^-[1][1]^T + 2\mu\mathcal{P}^-\right], \tag{22}$$

in which $[1] = \{1; 1; 0\}^T$ and

$$\mathcal{R}^+(\varepsilon) = \frac{1}{2}\left(\text{sign}(tr(\varepsilon)) + 1\right), \quad \mathcal{R}^-(\varepsilon) = \frac{1}{2}\left(\text{sign}(-tr(\varepsilon)) + 1\right). \tag{23}$$

where sign is the signum function.

The weak form of the displacement problem is finally expressed as:

$$\int_\Omega \varepsilon(u) : \mathcal{C}(d) : \varepsilon(\delta u)\ \mathrm{d}\Omega = \int_\Omega f^* \cdot \delta u\ \mathrm{d}\Omega + \int_{\partial\Omega_F} F^* \cdot \delta u\ dS. \tag{24}$$

Finite-element discretization

The displacement field, the strain tensor and their variations can be approximated in one element by:

$$u = N_u u_i \quad ; \quad \delta u = N_u \delta u_i \quad ; \quad \varepsilon(u) = B_u u_i \quad ; \quad \varepsilon(\delta u) = B_u \delta u_i, \tag{25}$$

where u_i are the nodal values of the displacement field u, N_u and B_u are matrices of displacement shape functions and of displacement shape functions derivatives, respectively.

The discretization of the mechanical problem (15) leads to the following discrete system of equations:

$$K_u u = F_u, \tag{26}$$

in which

$$K_u = \int_\Omega B_u^T \mathcal{C}(d) B_u d\Omega, \quad F_u = \int_\Omega N_u^T f^* d\Omega + \int_{\partial\Omega_F} N_u^T F^* dS. \tag{27}$$

In this work, the computations are carried out in quasi-static conditions. The damage and displacement problems are solved alternately using the staggered scheme. It should be noted that a single iteration is performed at each load increment as in [9]. In this approach, the accuracy of the results is highly dependent on the loading increments size. When relatively large time steps are used, inaccurate results can be found (see [48]). Considering a discretization of the time interval $[0, T]$, the present phase field algorithm is summarized in Algorithm (1).

Parallel implementation

The phase field method is a powerful tool to simulate material fracture and is widely used by the scientific community. However, within the finite-element context, limitations can be induced due to the fine mesh required to track nucleation and propagation of cracks, especially when dealing with large three-dimensional structures. This drawback can be overcome by massively parallel computing based on domain decomposition of the FEM mesh, as described in the following.

Algorithm 1 Phase field method

Initialization of the strain field $\varepsilon_0(x)$, the phase field $d_0(x)$ and the energy history $\mathcal{H}_0(x)$
for $t_{n+1} \leq T$ **do**
  Given ε_n, d_n and $\mathcal{H}_n$ at load increment t_n
  Compute damage :
    Compute $\mathcal{H}_{n+1}(\varepsilon_n, \mathcal{H}_n)$ by (10)
    Compute and assemble K_d and F_d by (14)
    Compute damage d_{n+1} by solving (13)
  Compute displacement :
    Compute $\mathcal{P}_n^{\pm}(\varepsilon_n)$ and $\mathcal{R}_n^{\pm}(\varepsilon_n)$ by (20), (23)
    Compute and assemble K_u and F_u by (27)
    Compute displacement u_{n+1} by solving (26)
    Compute strain ε_{n+1}
end for

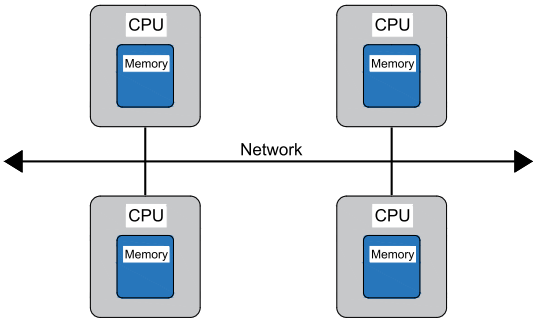

Fig. 2 Distributed-memory machines architecture

Machine architecture and MPI framework

In the field of parallel computing, there are mainly three widely used parallel programming methods; The Message Passing Interface (MPI) [49] is adapted for distributed-memory machines. The Open Multi-Processing (OpenMP) [50] is suitable for shared-memory machines, and the Compute Unified Device Architecture (CUDA) [51] is used for GPU acceleration. In this work, distributed-memory architectures are considered through the Message Passing Interface. This library ensures both point-to-point and collective communications between the distributed processes. This framework refers to parallel computing where multiple processes operate independently and have their own memory (see Fig. 2): each process can only access its local memory. Communication between processes is typically achieved through message passing, where data is explicitly sent from one process memory and received by another one. Our in-house developed implementation belongs to data partitioning parallel computing techniques with the same code acting on various part of data.

Domain decomposition

Domain decomposition is a key step in running parallel algorithms on a supercomputer. This process is required for decomposing operations into several sub-operations to be processed simultaneously. The main idea of domain decomposition is to divide the spatial domain into several smaller subdomains and assign to each process an independent task on a specific subdomain ensuring a well-balanced workload for efficient computations.

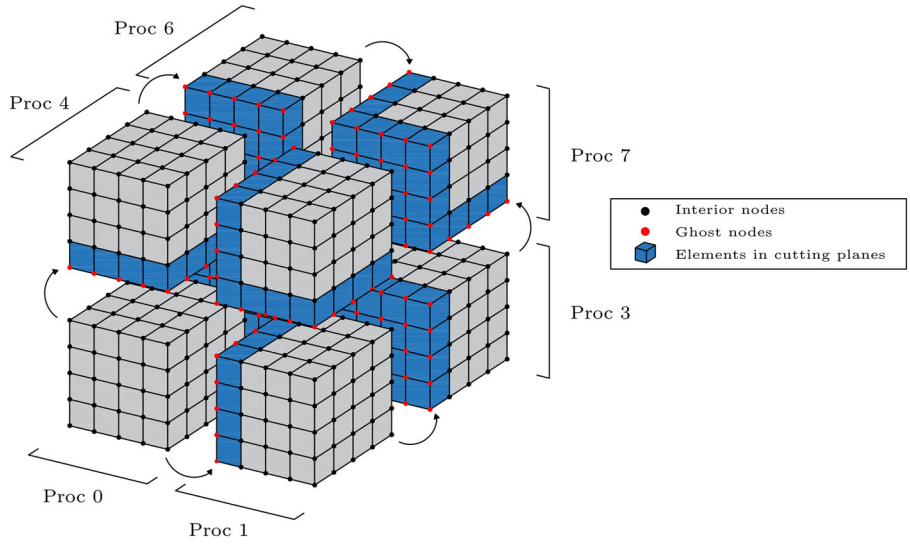

Fig. 3 Three-dimensional domain decomposition

In this work, within the finite-element method, three-dimensional structured meshes with 8-node finite elements are considered. In this context, three-dimensional blocks decomposition is used (see Fig 3). The structured mesh and spatial decomposition are created using the Data Management of Distributed Array tool (DMDA) of PETSc library [52].

In this context, the overall domain is partitioned into non-overlapping subdomains, each subdomain is assigned to a computational process. A topology of processes is created to ensure efficient use of computing resources and optimize communication between processes. The use of the phase field method within the framework of the finite-element method requires the resolution of two linear systems corresponding to the damage (8) and displacement (18) problems. The principle of parallel solving for these systems lies in the distribution of matrices and vectors across all processes. Sets of rows, corresponding to subdomains, are assigned to each process. To compute the global matrices and vectors K_d, F_d (14) and K_u, F_u (27), local operations, such as the computation of elemental stiffness matrices and vectors, are performed independently by all processes. In the case of cutting elements between subdomains, ghost nodes are introduced, representing additional degrees of freedom located on the boundaries of subdomains (see Fig 3). Ghost node values are updated to provide access to informations on interior nodes from adjacent subdomains through data exchange between processes. The global matrices and vectors are assembled in parallel by combining the local contributions, and once they are computed, the resolution of the linear systems (13) and (26) is performed with parallel solvers and preconditioners that are detailed in the next section.

Parallel solvers and preconditioners

The simulation of damage using the phase field method described in "Phase field method for brittle fracture" section within the finite-element method leads to the resolution of large sparse linear systems with a very small number of non-zero values as compared with their size. Most of the computation time is devoted to solving the linear systems

(13) and (26) of assembled matrices and vectors. The operations performed in these resolutions can consume up to 80% of the total computation time, making them the essential part of the numerical simulations. In this context, iterative solvers are here considered. This kind of solvers constructs increasingly accurate approximations to the solution through a sequence of iterations until reaching a convergence criterion. Krylov subspace methods are among widely used solvers, which include several algorithms such as the generalized minimal residual (GMRES) [53], conjugate gradient (CG) [54], BiCGStab [55], and BiCGStab(2) [56,57]. Iterative solvers performs operations such as matrix–vector products, dot products and linear combinations. They are capable of efficiently solving sparse, symmetric positive definite, and unsymmetric linear systems, require less memory allocation and are highly scalable.

In the present parallel computing framework, PETSc library is used to solve damage (8) and displacement (18) problems. The library provides a wide range of parallel solvers, including Krylov subspace methods. In this work, an improved version of the BiCGStab (IBiCGStab) method [58] is employed. This method is well-suited for the solutions of large and sparse linear systems on distributed-memory machines, the type of supercomputer architecture used in this work. It combines elements of numerical stability and parallel algorithm design without increasing the computational costs. In this approach, all inner products of a single iteration step are independent and the communication time required for these operations is overlapped efficiently with the computation time of vector updates, which are highly parallelizable. Therefore, the cost of global communication can be significantly reduced. The algorithm of the IBiCGStab solver is presented in Algorithm (2).

Algorithm 2 The IBiCGStab method [58]

Initialization of vectors and parameters:
$r_0 = b - Ax_0,\ u_0 = Ar_0,\ f_0 = A^T r_0,\ \sigma_0 = r_0^T u_0,\ \rho_0 = \alpha_0 = \omega_0 = 0;$
$q_0 = v_0 = z_0 = \sigma_{-1} = \pi_0 = \phi_0 = \tau_0 = 0;$
for $n = 1, 2, 3, \ldots$ **do**
  $\rho_n = \phi_{n-1} - \omega_{n-1}\sigma_{n-2} + \omega_{n-1}\alpha_{n-1}\pi_{n-1};$
  $\delta_n = \frac{\rho_n}{\rho_{n-1}}\alpha_{n-1},\ \beta_n = \frac{\delta_n}{\omega_{n-1}};$
  $\tau_n = \sigma_{n-1} + \beta_n\tau_{n-1} - \delta_n\pi_{n-1};$
  $v_n = u_{n-1} + \beta_n v_{n-1} - \delta_n q_{n-1};$
  $\alpha_n = \frac{\rho_n}{\tau_n},\ q_n = Av_n;$
  $s_n = r_{n-1} - \alpha_n v_n,\ t_n = u_{n-1} - \alpha_n q_n;$
  $z_n = \alpha_n r_{n-1} + \beta_n z_{n-1} - \alpha_n\delta_n v_{n-1};$
  $\phi_n = r_0^T s_n,\ \pi_n = r_0^T q_n,\ \gamma_n = f_0^T s_n,\ \eta_n = f_0^T t_n;$
  $\theta_n = s_n^T t_n,\ \kappa_n = t_n^T t_n,\ \omega_n = \frac{\theta_n}{\kappa_n};$
  $\sigma_n = \gamma_n - \omega_n\eta_n,\ r_n = s_n - \omega_n t_n;$
  $x_n = x_{n-1} + z_n + \omega_n s_n;$
  if residual norm is small enough **then**
    STOP
  end if
  $u_n = Ar_n;$
end for

Iterative solvers stand as powerful tools for solving large-scale linear systems in parallel, however, they are very sensitive to conditioning. In the case of the mechanical linear system

(26), a large contrast in stiffness is produced due to the crack nucleation and propagation, which makes the system poorly-conditioned [59]. As a result, iterative solvers converge slowly and the computational efficiency decreases. Additionally, when dealing with large-scale problems, the number of iterations of the iterative solvers growths with the mesh refinement [60], which increases the computational time. To tackle these challenges, preconditioning techniques stand as a key ingredient to achieving a high convergence rate. Preconditioning consists of transforming the linear system $Ku = F$ into the equivalent system:

$$M^{-1}Ku = M^{-1}F. \tag{28}$$

Such a strategy is called left preconditioning, where the matrix M is a preconditioner. Right preconditioning could be used:

$$KM^{-1}w = F, \quad Mu = w, \tag{29}$$

as well as bilateral preconditioning:

$$M^{-1}KM'^{-1}w = M^{-1}F, \quad M'u = w, \tag{30}$$

where the matrices M and M' are preconditioners.

To accelerate convergence and improve the efficiency of iterative solvers, PETSc library offers a large number of preconditioners. In the proposed implementation, the Algebraic MultiGrid (AMG) [61,62] preconditioner is considered. The AMG methods construct a multilevel hierarchy of grids based solely on the algebraic properties of the linear system matrix, which is their main advantage over geometric methods. The linear system is solved iteratively on the coarsest grid, typically using a direct solver or an iterative method. The coarse-grid solutions are then interpolated back to finer grids to improve the solution accuracy. At each level of the hierarchy, AMG applies prolongation and restriction operators to transfer information between grids, capturing both low and high-frequency components of the error. This preconditioner can be used with PETSC library as PCGAMG or via the external package HYPRE [63,64] as BoomerAMG. Additionally, integrating a smoothing technique with the AMG preconditioner can improve its performance by reducing the high-frequency error components. In this work, the smoothed aggregation AMG [65] with Chebyshev-Jacobi smoother is adopted as in [43]. Its effectiveness has been proven in solving mechanical problems using the finite-element method [60,66], and using the phase field method for damage simulation in [43]. For more details about the preconditioner AMG with Chebyshev-Jacobi smoother, see [43].

It is worth mentioning that the present methodology cannot be applied directly to unstructured meshes. The first ingredient to be modified would be the meshing algorithm, which can consist of either tetrahedral mesh construction in arbitrary geometries, or refined meshes in structured grids. The second ingredient is mesh and data partitioning, which should be adapted to unstructured meshes, to balance sub-domains while minimizing communication overheads. The rest of the methodology would remain unchanged.

Validation and performance evaluation

In this section, the validation of the proposed implementation and the evaluation of its performance are presented. On one hand, the validation test and the first numerical example in "Numerical applications" section are performed on a medium-sized workstation. This is an HPE ProLiant DL560 Gen10 server equipped with two processors Intel(R) Xeon(R) Platinum 8260 L, each with 24 cores, and a memory capacity of 6 TB. the base frequency of the processors is about 2.4 GHz and the maximum Turbo frequency is 3.90 GHz. On the other hand, the performance analysis and all the remaining numerical examples in "Numerical applications" section are carried out on the Jean Zay HPE SGI 8600 supercomputer from the Institute for Development and Resources in Intensive Scientific Computing (IDRIS-CNRS). This supercomputer is made up of 720 nodes, each node being equipped with 40 cores and a 190 GB memory capacity. The processors installed are Intel(R) Xeon(R) Gold 6248 with a base frequency of 2.5 GHz and a maximum Turbo frequency of 3.90 GHz.

In this work, the relative residual is used for the IBiCGStab solver convergence criterion, where the relative tolerence is chosen as $R_{tol} = 10^{-7}$ for both damage (8) and displacement (18) problems, in all simulations. For the AMG preconditioner, V-Cycle approach is used for all simulations and for both problems. The linear system (13) involved in the damage problem (8) is well-conditioned, allowing the use of a 1-level multigrid algorithm for all simulations. It is enough to obtain a good convergence rate, while optimizing memory usage. For the mechanical problem (18), a 5-level multigrid algorithm is applied for the validation test, the performance analysis, and the first two applications in "Numerical applications" section, while a 6-level multigrid algorithm is applied for the two remaining applications in "Numerical applications" section. All remaining AMG preconditioner parameters are set to the default values provided by PETSc with PCGAMG.

Validation with the single notched structure

The proposed solving procedure is first validated on an academic example [9]. In this context, a cubic domain containing an initial crack with dimensions $L \times H \times l = 1 \times 1 \times 1$ mm^3 is considered (see Fig. 4). The bottom face of the structure is fixed in all directions to simulate a rigid support, and a uniform displacement $\overline{U}$ is applied on the top face of the structure in the z-direction. The simulation test is performed on a domain composed of 155^3 elements and the mesh size is $h = 0.0065$ mm. The characteristic length scale is chosen as $\ell_0 = 0.015$ mm where $2\,h < \ell_0$. The structure is homogeneous isotropic with the properties $\lambda = 121.15$ kN/mm^2, $\mu = 80.77$ kN/mm^2, and $G_c = 0.0027$ kN/mm for the fracture toughness. Since structured meshes are used in this work, notches and voids are modelled by the properties $\lambda = 0$ kN/mm^2 and $\mu = 0.001$ kN/mm^2. The specific fracture energy is chosen to be at least 10 times less than the minimum of the specific fracture energies among all phases within the structure. The computation is carried out with constant displacement increments of $\Delta u = 10^{-5}$ mm for the first 500 time steps and $\Delta u = 10^{-6}$ mm for the remaining time steps.

Figure 5a-c illustrate the evolution of the crack propagation for displacements $\overline{U} = 57.10^{-4}$ mm, $\overline{U} = 60.10^{-4}$ mm, and $\overline{U} = 62.10^{-4}$ mm, respectively. The crack is depicted by selecting damage values in the mesh larger than 0.97. To validate the proposed in-house parallel implementation, a comparison of force-displacement curves is made with

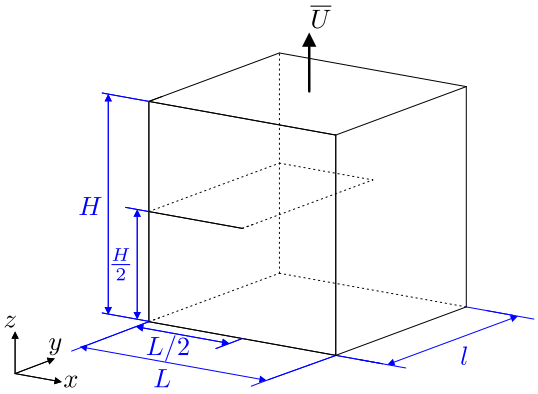

Fig. 4 Validation test geometry

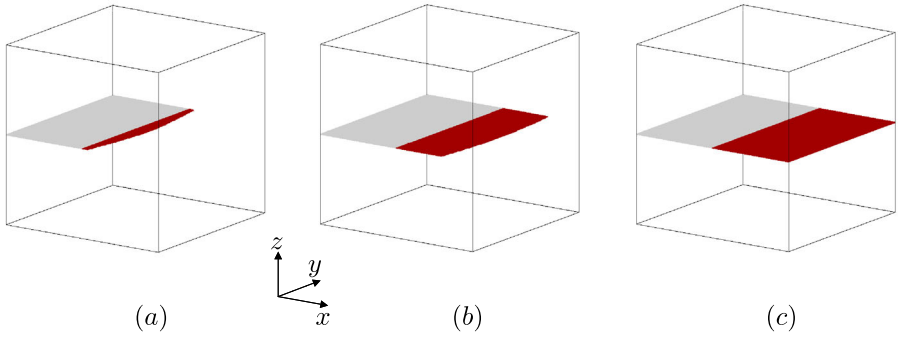

Fig. 5 Phase field $d(x)$ distribution during crack evolution for the validation test for **a** $\overline{U} = 57.10^{-4}$ mm; **b** $\overline{U} = 60.10^{-4}$ mm; **c** $\overline{U} = 62.10^{-4}$ mm

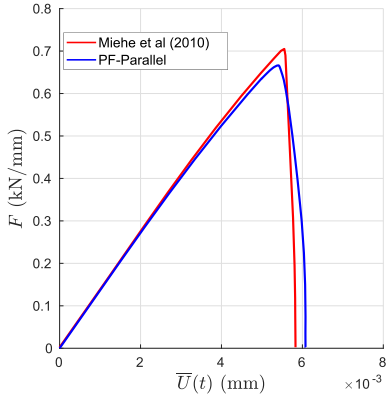

Fig. 6 Force-displacement curve of the validation test

the solution provided in [9]. In Fig. 6, a good agreement is observed with the reference solution, which validates the proposed implementation. The slight variations may come from the fact that the reference solution is performed in 2D plane strain assumption, while our solution is fully 3D.

Performance evaluation

The performance of the proposed in-house developed parallel implementation is assessed by measuring its scalability, which indicates the ability to increase computing speed with

the number of processes. Two parameters are widely employed for this purpose, speed-up and efficiency. The speed-up can be defined in two situations, strong scalability and weak scalability, defined as follows.

- Strong scalability: refers to the ability of a parallel algorithm to improve its performance when the problem size remains constant but the number of processes is increased. As more resources are added, the computation time decreases. The speed-up is in this defined as the ratio of the computation time t_{ref} obtained using a reference number of processes p_{ref} with the computation time t_p obtained using a specified number of processes p.

$$S_s(p) = \frac{t_{ref}}{t_p} p_{ref}. \tag{31}$$

When very large meshes are employed, solving the systems (13) and (26) involve a very large number of unknowns. Then, using a single process requires an enormous amount of computing time. For this reason, the reference computations are performed with n processes instead of using a single process for sequential computations ($t_{ref} \neq t_{seq}$).

- Weak scalability: is the ability of a parallel algorithm to improve its performance as both the problem size and the number of processes are increased proportionally. In this case, the workload assigned to each process remains constant as the problem size increases. In the ideal case, the computation time then remains constant. The speed-up is defined as:

$$S_w(p) = \frac{t_{ref}}{t_p} p. \tag{32}$$

Efficiency is defined for both scalability cases and evaluates how computational resources are used effectively in a parallel algorithm. It is expressed as the ratio of the speed-up with the number of processes:

$$E_{s,w}(p) = \frac{S_{s,w}(p)}{p}. \tag{33}$$

It is important to mention that, in this work, the performance analysis does not take into account the preliminary operations such that mesh construction and partitioning, data import and memory allocation. For the analysis, all the operations involved in the load increments described in Algorithm (1) are considered:

- Computations on the elements (local stiffness matrices and vectors).
- Matrices and vectors assembly.
- Boundary conditions application for the displacement problem (18).
- Problems solving (13) and (26).
- Data exchanges between the processes.

To investigate the scalability of this framework, the notched homogeneous structure in Fig. 8 submitted to tensile loading is considered. The weak scalability computations are carried out on domains ranging from 1.3×10^6 to 3.5×10^9 elements, with the number of elements per process being fixed to 70^3, the number of processes starting from 40 and progressively increasing up to 10240 processes. The strong scalability computations are performed in a domain with 219×10^6 elements. The number of processes is progressively increased from 640 to 10240. Both scalability analysis are performed over 5 loading increments with 20 iterations of the displacement problem (26) and 1 iteration of the damage

Table 1 Configurations used in weak scalability computations

N° El/dir	N° DOFs/d	N° DOFs/u	N° Procs	Workload (w^3)	Time (s)
239	13 824 000	41 472 000	40	69.88	297.15
302	27 818 127	83 454 381	80	70.09	302.69
380	55 306 341	165 919 023	160	70.00	301.38
479	110 592 000	331 776 000	320	70.03	300.72
603	220 348 864	661 046 592	640	69.97	304.11
760	440 711 081	1 322 133 243	1280	70.00	302.97
957	879 217 912	2 637 653 736	2560	69.96	302.54
1206	1 758 416 743	5 275 250 229	5120	69.97	307.16
1520	3 518 743 761	10 556 231 283	10240	70.00	337.01

Table 2 Configurations used in strong scalability computations

N° El/dir	N° DOFs/d	N° DOFs/u	N° Procs	Workload (w^3)	Time (s)
			640	69.97	304.11
			1280	55.54	153.64
			2560	44.08	77.58
603	220 348 864	661 046 592	3840	38.51	52.29
			5120	34.99	40.25
			6400	32.48	32.66
			10240	27.77	23.06

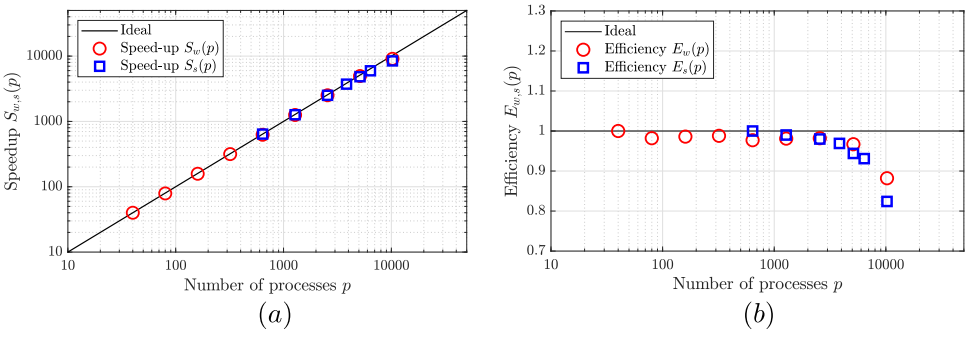

Fig. 7 **a** Speed-up; **b** Efficiency

problem (13) (the first loading increments are used where no damage has yet occurred). Tables 1 and 2 present the configurations used and computation times for weak and strong scalability analysis, respectively, where N° El/dir is the number of elements per direction, and N° DOFs/d and N° DOFs/u are numbers of degrees of freedom of the damage and displacement vectors d and u in (13) and (26), respectively.

The weak and strong scalability are shown in Fig. 7a and b, which present the effectiveness of the proposed phase field method parallel implementation. Regarding strong scalability, the present implementation shows very good scaling performance. The achieved speed-up closely approximates the ideal one. The efficiency is 97% for a computation on 6400 processes for 33^3 elements per process and greater than 80% for a computation on 10240 processes for 23^3 elements per process. This result is due to the well-balanced workload distribution and optimized data exchanges between processes. The weak scalability analysis reveals that the proposed framework maintains a consistent level of performance when the problem size is increased. A linear speed-up can be noticed regardless of the processes number and the problem size.

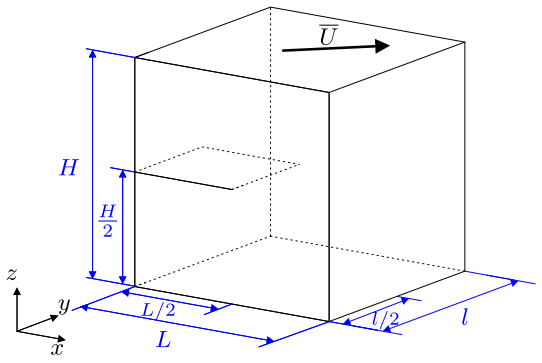

Fig. 8 Geometry of the notched homogeneous structure

In terms of memory allocation, the proposed parallel implementation makes optimal use of resources. A domain with more than 12×10^6 elements can be simulated with less than 190 GB of memory capacity, considering that solving the linear systems of damage (13) and displacement (26) problems is carried out with assembled matrices within the finite-element method. This optimization of memory management indicates that large-scale simulations can be carried out using fewer resources [45], thereby reducing the costs associated with computational infrastructures.

These results confirm that the proposed implementation scales efficiently with the problem size and the available computing resources, and that the scalability improvement is not limited solely to parallel solvers. It encompasses all operations performed within the loading increment loop mentioned above, which validates the effectiveness of the parallel implementation and shows its potential for tackling larger computational problems with improved efficiency.

Numerical applications

In this section, numerical applications of crack initiation and propagation using the proposed parallel implementation are presented.

Notched homogeneous structure

As a first example, the crack propagation in a cubic notched structure (see Fig. 8) submitted to shear loading is considered. The problem is simulated using 48 processes with the medium-sized workstation. The structure is homogeneous isotropic having the properties $\lambda = 30\,\text{kN/mm}^2$, $\mu = 20\,\text{kN/mm}^2$, and $G_c = 2.6.10^{-7}\,\text{kN/mm}$ for the fracture toughness. The domain is composed of 155^3 (3.72×10^6) elements, with dimensions $L \times H \times l = 1 \times 1 \times 1\ \text{mm}^3$. The elements size is $h = 0.0065$ mm. The characteristic length scale is chosen as $\ell_0 = 2h$. As depicted in Fig. 8, The notch occupies only half the length of the structure in both x- and y-directions. Concerning the boundary conditions for the displacement problem, the bottom face of the structure is fixed in all directions to simulate a rigid support and all the structure faces are blocked along the z-direction. A uniform displacement is applied to the top face of the structure according to $\overline{U} = j\Delta u\,(e_1 + e_2)$, where e_1 and e_2 are the normal unit vectors along x- and y-directions and j is the load increment number. The computation is carried out with constant displacement increments of $\Delta u = 4.6.10^{-6}$ mm.

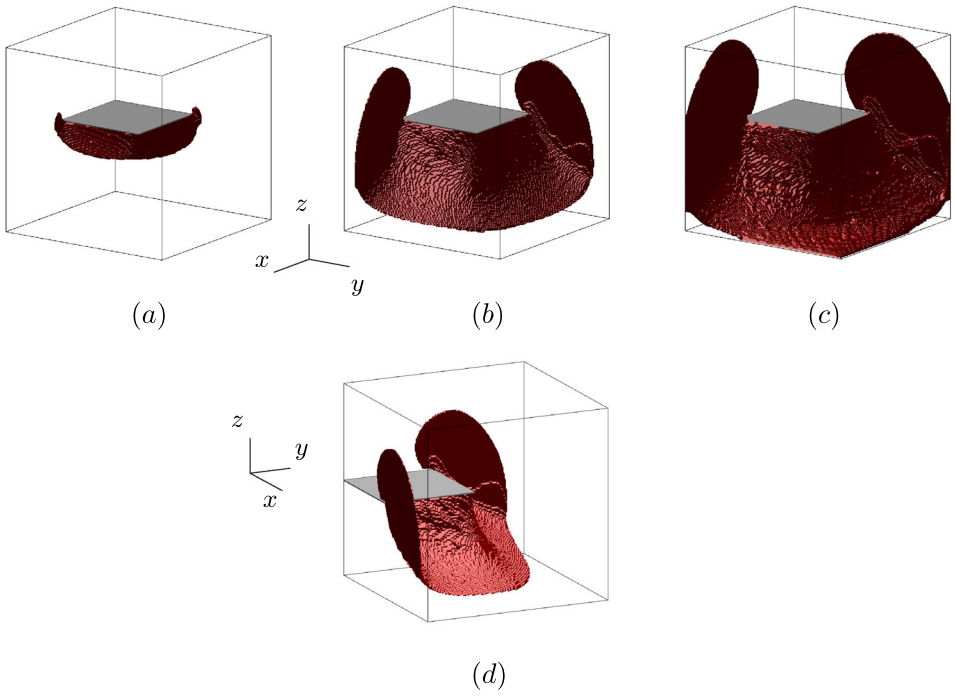

Fig. 9 Damage evolution of the notched homogenous structure at time steps **a** $\overline{U} = 39.10^{-5}$ mm; **b** and **d** $\overline{U} = 56.10^{-5}$ mm; **c** $\overline{U} = 60.10^{-5}$ mm

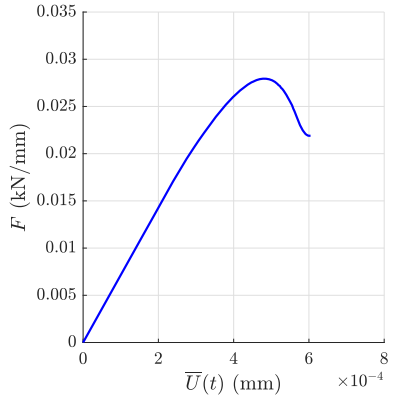

Fig. 10 Force-displacement curve of the notched homogeneous structure

Figure 9a-d shows crack evolution (in red) at displacements $\overline{U} = 39.10^{-5}$ mm, $\overline{U} = 56.10^{-5}$ mm, and $\overline{U} = 60.10^{-5}$ mm, respectively. The force-displacement curve is shown in Fig 10. Figure 11a shows the number of iterations of each loading increment for the damage and displacement solvers. Three phases can be noticed: the first concerns the undamaged structure where the force-displacement curve is linear. The damage problem takes a single iteration to converge while the displacement problem takes 7 iterations. In the second phase, the number of iterations of the damage problem increases, while the displacement problem keeps the same number of iterations. This is due to the low values of the damage. The third phase is characterized by the crack propagation, the number of iterations of the displacement problem increases significantly due to the discontinuity of the structure stiffness caused by the crack. Figure 11b shows the resolution times of both

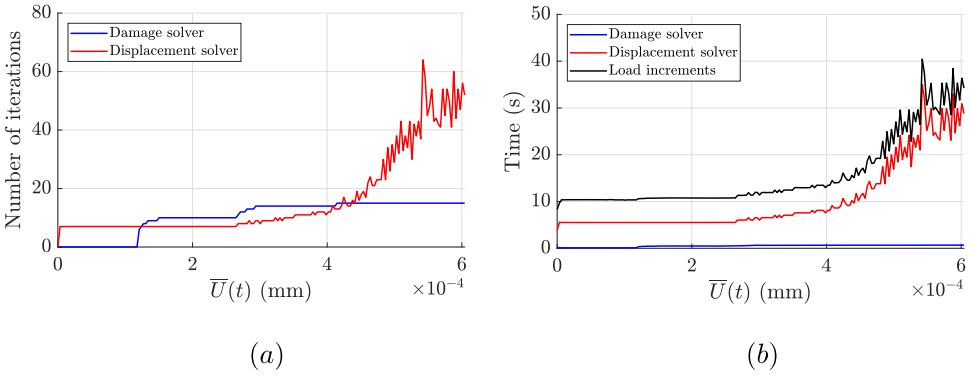

Fig. 11 **a** Iterations number of the damage and displacement problems resolution; **b** Times of the damage and displacement problems resolution and time of loading increments of the notched homogeneous structure

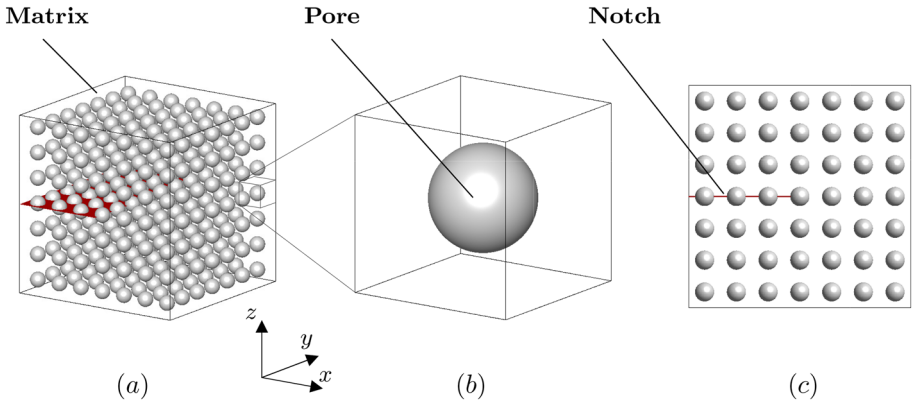

Fig. 12 Geometry of the notched periodic structure with pores: **a** Three-dimensional structure; **b** RVE; **c** Slice of the structure at the position $y = \frac{l}{2}$

damage and displacement linear systems for each time step, as well as the execution time of all the operations in each load increment. The time needed to solve the damage problem is very short compared with the displacement problem resolution. The evolution of the load increments time looks the same as the resolution of the displacement problem. This proves that solving the displacement problem is the dominant part of the simulation and the most costly in terms of computations. The cumulative times for computing and assembling the global matrix and vector K_d, F_d (14) and solving the damage linear systems (13) over all load increments are 69s (1.15min) and 93s (1.55min), respectively. The cumulative times for computing and assembling the global matrice and vector K_u, F_u (27) and solving the mechanical linear systems (26) are 689s (11min) and 1918s (32min), respectively. The total execution time for all loading increments is 2893s (48min), compared to nearly 38h with a sequential execution. Then, we can observe that the total time devoted to solve the mechanical problems takes 66% of the total time.

Notched periodic porous structure

In the second example, the crack propagation of a notched periodic structure with pores submitted to tensile loading is simulated (see Fig. 12). The structure dimensions are $L \times H \times l = 1 \times 1 \times 1$ mm^3 and the pores radius is $r = 43.10^{-3}$ mm. The position of the notch is similar to that of the validation test in "Validation and performance evaluation"

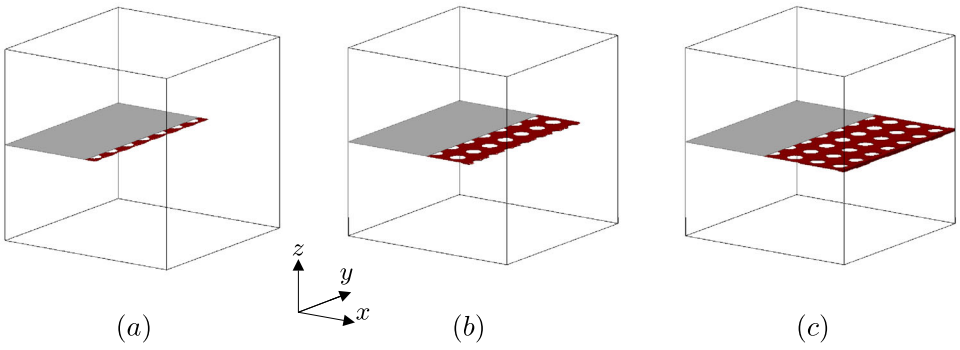

Fig. 13 Damage evolution of the notched periodic structure with pores at time steps **a** $\overline{U} = 15.10^{-5}$ mm; **b** $\overline{U} = 18.10^{-5}$ mm; **c** $\overline{U} = 21.10^{-5}$ mm

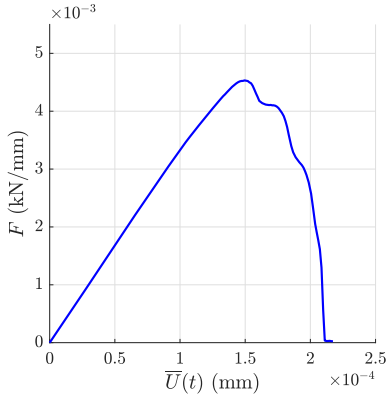

Fig. 14 Force-displacement curve of the notched periodic structure with pores

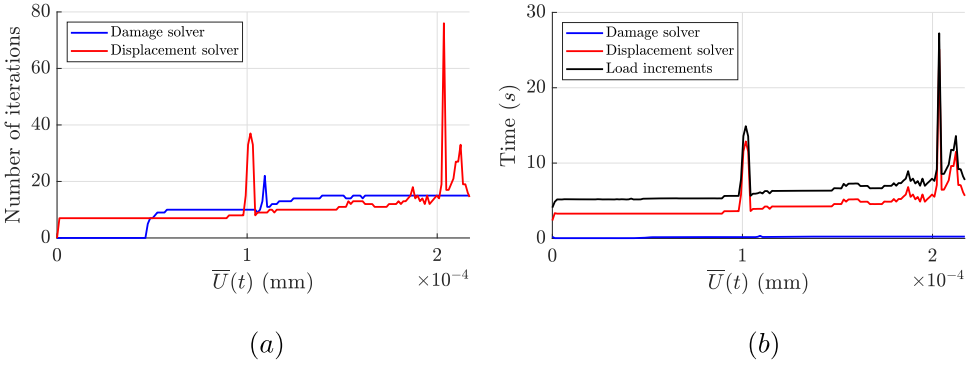

Fig. 15 **a** Iterations number of the damage and displacement problems resolution; **b** Times of the damage and displacement problems resolution and time of loading increments of the notched periodic structure with pores

section. The matrix (see Fig. 12) has the properties $\lambda = 30$ kN/mm^2, $\mu = 20$ kN/mm^2, and $G_c = 1.32.10^{-7}$ kN/mm for the fracture toughness. The structure is composed of 301^3 (27.27×10^6) elements. The elements size is $h = 0.0033$ mm and the characteristic length scale is $\ell_0 = 2h$. For the displacement problem, the bottom face of the structure is fixed in all directions to simulate a rigid support, all the structure faces are blocked along the x- and y-directions, and a distributed load $\overline{U}$ is applied to the top face of the structure in the z-direction. The simulation is performed using Jean Zay supercomputer with 1200

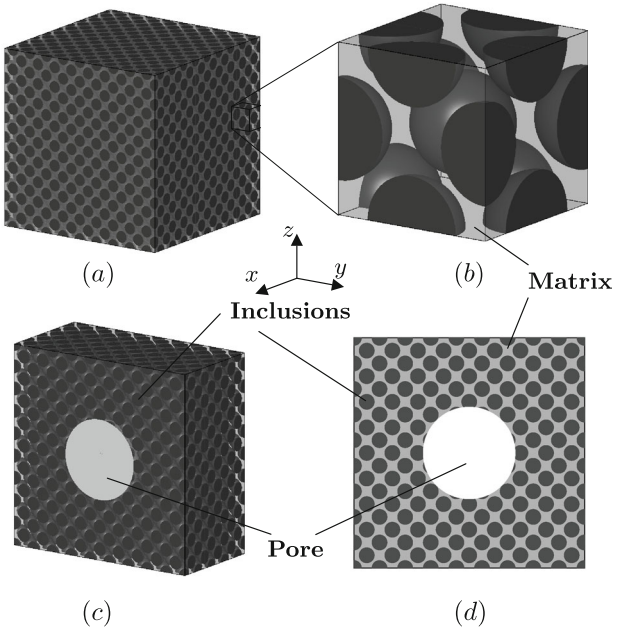

Fig. 16 Geometry of the periodic structure with pore and inclusions: **a** three-dimensional structure; **b** RVE; **c** a half of the structure ($x \in [0,\ \frac{L}{2}]$); **d** slice of the structure in the position $x = \frac{L}{2}$

processes. The computation is carried out with constant displacement increments of $\Delta u = 1.25.10^{-6}\ mm$.

Figure 13a-c show the damage evolution at displacements $\overline{U} = 15.10^{-5}$ mm, $\overline{U} = 18.10^{-5}$ mm, and $\overline{U} = 21.10^{-5}$ mm, respectively. The force-displacement curve is shown in Fig 14. The proposed parallel implementation can simulate the crack propagation jumping from pores to pores in this example. As in the previous example, Fig. 15a shows the same phases of evolution in the number of iterations. However, in this case, a pic in the number of iterations is observed when the crack reaches the boundary of the sample, leading to the full damage of the structure. In some cases, the displacement problem solver for (26) has difficulties converging. Figure 15b shows the times to solve the damage (13) and displacement (26) problems and the time of the execution of all the operations for each load increment. The cumulated times for the damage and displacement problems resolution are 30s (0.5min) and 784s (13min), respectively. The time of all loading increments execution is 1137s (19min), instead of about 15 days with a sequential execution. The total time devoted to solve the mechanical problems takes 68% of the total time.

Periodic structure with pore and inclusions

For the third example, the crack initiation and propagation of a periodic structure with a centered pore and inclusions submitted to tensile loading is performed (see Fig. 16). The structure matrix have the properties $\lambda_m = 30$ kN/mm^2, $\mu_m = 20$ kN/mm^2, and $G_c^m = 10^{-7}$ kN/mm for the fracture toughness. For the inclusions, $\lambda_i = 300$ kN/mm^2, $\mu_i = 200$ kN/mm^2, and $G_c^i = 10^{-5}$ kN/mm. The structure is composed of 405^3 (66.43×10^6) elements, with dimensions $L \times H \times l = 1 \times 1 \times 1$ mm^3. Pore and inclusions radii are $R = 0.2$ mm and $r = 43.10^{-3}$ mm, respectively. The elements size is $h = 0.0025$ mm and the characteristic length scale is $\ell_0 = 2h$. Boundary conditions for the displacement

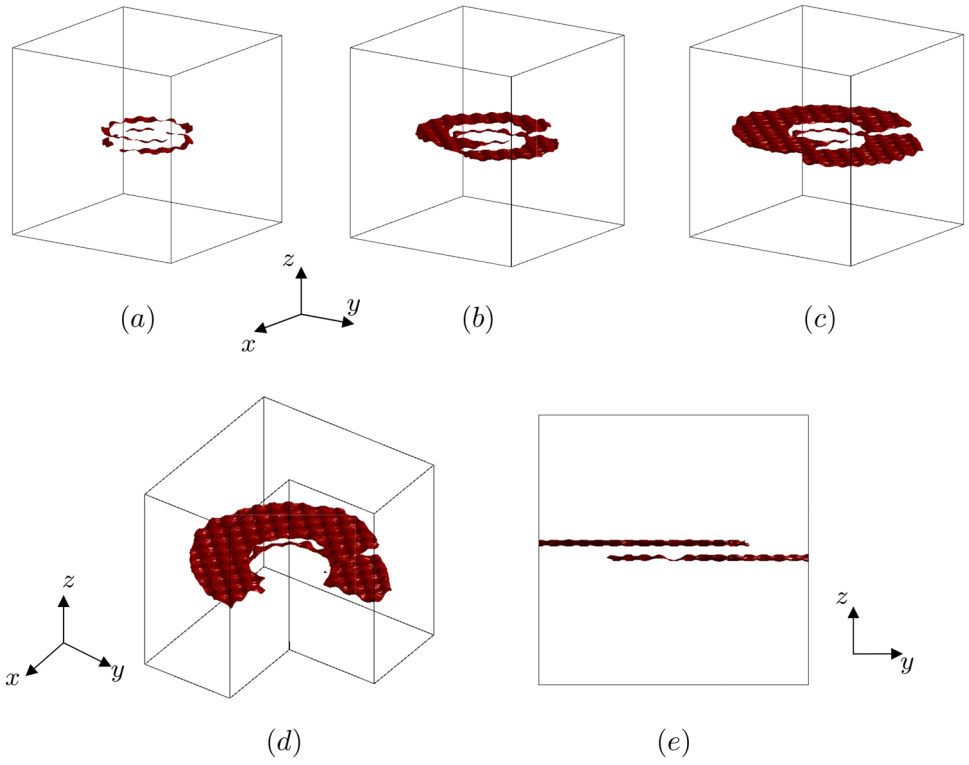

Fig. 17 Damage evolution of the notched periodic structure with pores at time steps **a** $\overline{U} = 173.10^{-6}$ mm; **b** $\overline{U} = 177.10^{-6}$ mm; **c** $\overline{U} = 180.10^{-6}$ mm; **d** and **e** Slices of the damage state (**c**)

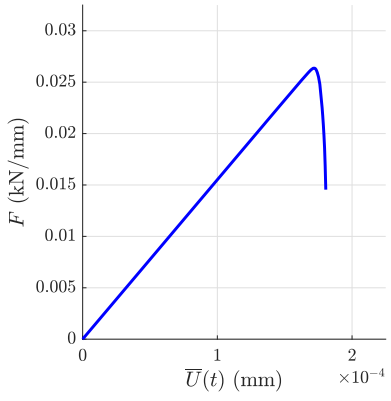

Fig. 18 Force-Displacement curve of the notched periodic structure with pores

problem are similar than in the previous example. The computations are carried out with constant displacement increments of $\Delta u = 45.10^{-8}$ mm. The simulation is performed using Jean-Zay supercomputer with 3200 processes.

Figure 17a-c illustrate the crack evolution (in red) at the time steps $\overline{U} = 173.10^{-6}$ mm, $\overline{U} = 177.10^{-6}$ mm, and $\overline{U} = 180.10^{-6}$ mm, respectively. Figures 17d, e show slices of the Fig. 17c. The curve force-displacement is shown in Fig 18. This example shows that the proposed implementation is capable of modelling crack initiation and propagation in complex heterogeneous structures and materials. It can be noticed that two circular cracks are generated from the central pore, and have a wavy shape due to their interaction with

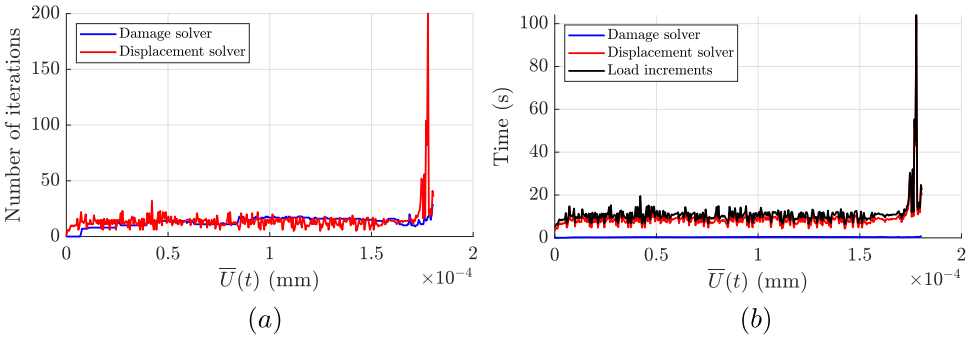

Fig. 19 **a** Iterations number of the damage and displacement problems resolution; **b** times of the damage and displacement problems resolution and time of loading increments of the notched periodic structure with pores

the inclusions. Figure 19a shows the number of iterations of each loading increment for the damage (13) and displacement (26) linear systems resolution. In this case, unlike the previous example, the number of iterations fluctuates along the whole simulations, which might be due to the interaction between the crack propagation and the heterogeneities within the structure. When the cracks reach the boundary of the sample, the number of iterations jump and the solver converges very slowly. The search for better convergence of the iterative solver with hole simulation in such problems deserves to be studied in future work. Figure 19b shows the resolution times of damage (13) and displacement problems (26) resolution and the time of the execution of all the operations for each load increment. The cumulative times for solving the damage and displacement problems are 142s (2.3min) and 3618s (60.3min), respectively. The complete execution of all loading increments took 4443s (74min), instead of approximately 5 months with a sequential execution. The total time devoted to solve the mechanical problems takes 81% of the total time.

Experimental image-based microstructure obtained from XR-μCT

In the last example, the crack initiation and propagation of a realistic geometrical model of a microstructure directly obtained from X-Ray micro computed tomography (XR-μCT) of lightweight concrete is simulated (see Fig. 20). For more details about the numerical model, see [23]. A centered pore of a radius $R = 0.15$ mm is added to force the crack nucleation. The phase properties are chosen as follow: $\lambda_m = 30$ kN/mm^2, $\mu_m = 20$ kN/mm^2, and $G_c^m = 8.10^{-8}$ kN/mm for the matrix, and $\lambda_i = 300$ kN/mm^2, $\mu_i = 200$ kN/mm^2, and $G_c^i = 8.10^{-6}$ for the aggregates. The structure is composed of 500^3 (125×10^6) elements, with dimensions $L \times H \times l = 1 \times 1 \times 1$ mm^3. The elements size is $h = 0.0002$ mm and the characteristic length scale is $\ell_0 = 2h$. The boundary conditions for the displacement problem are similar to the previous example. The simulation is performed using Jean-Zay supercomputer with 6000 processes. the computation performed with constant displacement increments of $\Delta u = 4.10^{-7}$ mm.

Figure 21a-c illustrate the structure damage states at the time steps $\overline{U} = 157.10^{-6}$ mm, $\overline{U} = 178.10^{-6}$ mm, and $\overline{U} = 200.10^{-6}$ mm, respectively. Figures 21d, e show slices of the Fig. 21c. The Force-Displacement curve is shown in Fig. 22. This example shows that the proposed implementation is capable of modelling crack initiation and propagation in highly heterogeneous structures, in particular, real structures obtained using experimental imagery techniques. The framework can manage the nucleation of several cracks in

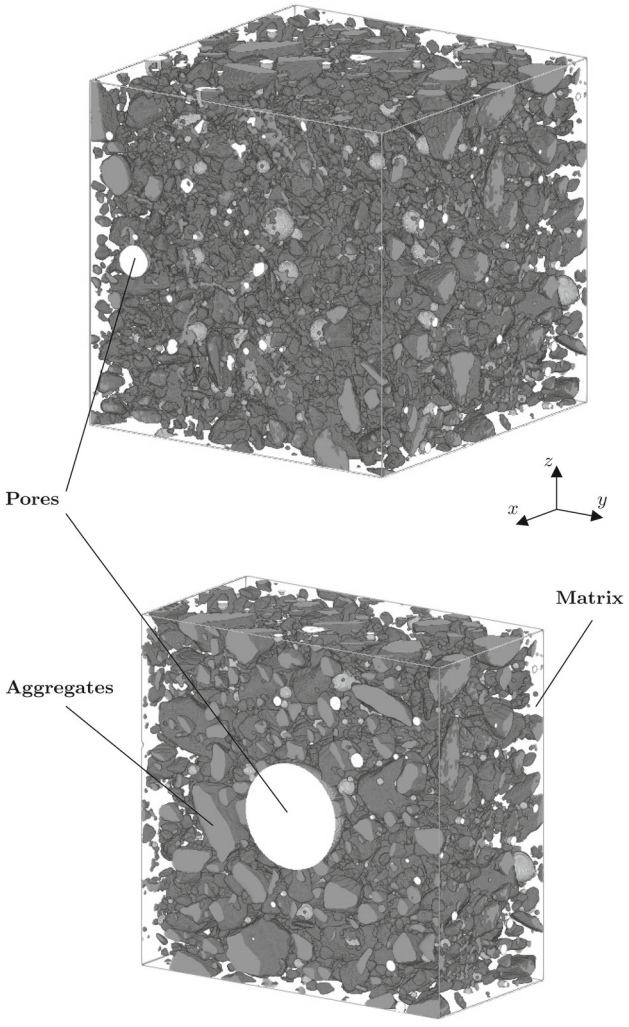

Fig. 20 Geometry of the realistic microstructure obtained from experimental XR-μCT: **a** Three-dimensional structure; **b** a half of the structure ($x \in [0, \frac{L}{2}]$)

different locations and their connections. Figure 23a shows the number of iterations of each loading increment for the damage and displacement solvers. Figure 23b shows the resolution times of damage and displacement problems resolution and the time of the execution of all the operations for each load increment. The cumulative times for solving the damage and displacement problems are 102s (1.7min) and 3220s (53.7min), respectively. The time of all loading increments execution is 4150s (69.1min), instead of about 10 months with a sequential execution. The total time devoted to solve the mechanical problems takes 77% of the total time.

Conclusions

In this work, an efficientin- parallel implementation of the phase field method has been proposed to simulate damage of quasi-brittle heterogeneous materials on supercomputers. This implementation, developed in the context of the finite-element method, is capable of modelling damage in very large heterogeneous three-dimensional structures with

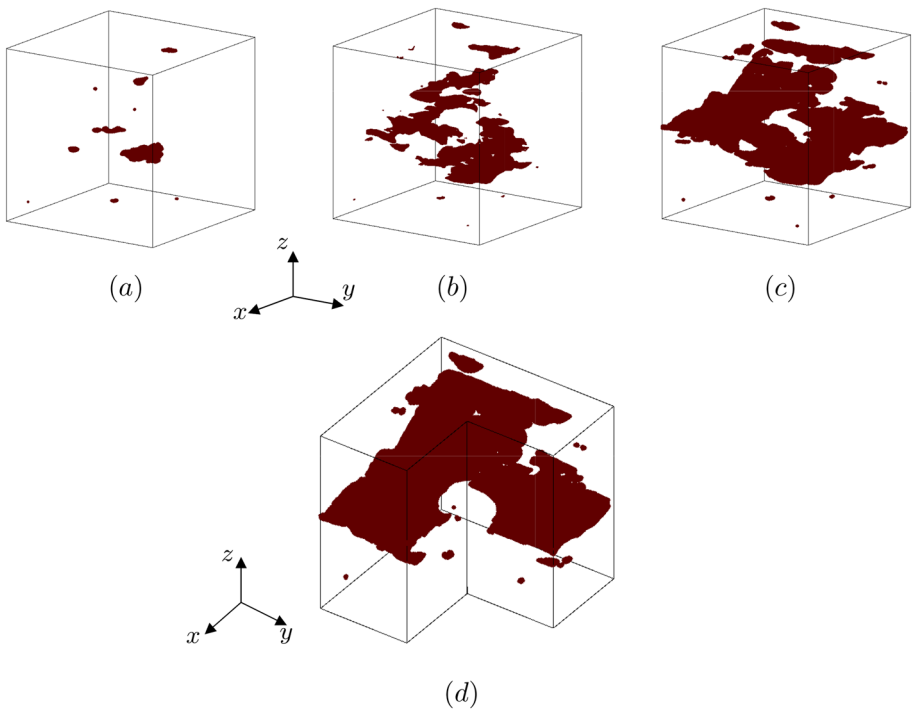

Fig. 21 Damage evolution of the real structure at time steps **a** $\overline{U} = 157.10^{-6}$ mm; **b** $\overline{U} = 178.10^{-6}$ mm; **c** $\overline{U} = 200.10^{-6}$ mm; **d** Slice of the damage state (**c**)

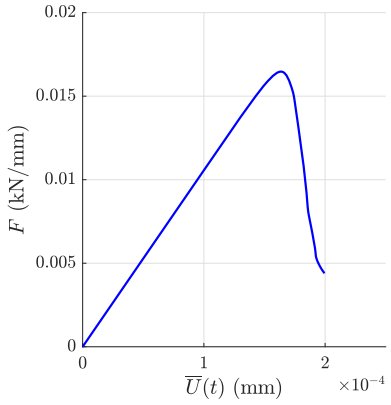

Fig. 22 Force-Displacement curve of the real structure

improved efficiency and using fewer resources than other available techniques. The parallel implementation of the phase field method is based on domain decomposition, each process performs computations on a subdomain. The MPI library is used to ensure data distribution and communication between processes. To solve the damage and displacement problems, global matrices and vectors are computed and assembled in parallel. The resolution of the linear systems is performed in parallel using the IBiCGStab solver and Smoothed Aggragation AMG preconditioner. The performance analysis has been carried out using the Jean Zay supercomputer from IDRIS. The efficiency has been 97% for a computation on 6400 processes and greater than 80% for a computation on 10240 processes, which makes this proposed implementation fast and extremely efficient. The linear

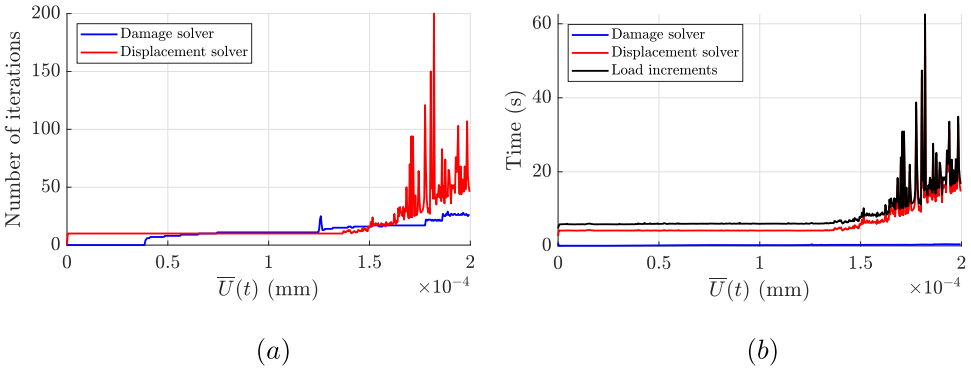

Fig. 23 **a** Iterations number of the damage and displacement problems resolution; **b** times of the damage and displacement problems resolution and time of loading increments of the real structure

systems involved in the simulations with up to 10^{10} degrees of freedom can be solved in a few seconds. Numerical applications of crack initiation and propagation using three-dimensional periodic structureshave been presented. A complete damage simulation of a realistic geometrical model of lightweight concrete obtained by micro-CT imaging with 375×10^6 degrees of freedom was carried out with 500 time steps. Using 6000 processes, the simulation was performed in just 1 hour and 9 minutes instead of around 10 months employing a sequential implementation. These results demonstrate the effectiveness of the proposed implementation.

Acknowledgements
This work was granted access to the HPC resources of IDRIS under the allocation 2023-AD010914842 made by GENCI. We would like to express our gratitude to the technical teams of IDRIS center for their kind and efficient help.

Author contributions
Conceptualization, Methodology, Formal analysis: ZC, JY, JB, SV, SEO Writing–original draft: ZC, JY, JB Investigation and Validation: ZC, JY, JB Software: ZC Writing–review & editing: ZC, JY, JB Supervision, Project administration, and Funding acquisition: JY, JB All authors read and approved the final manuscript.

Funding
The authors gratefully acknowledge the financial support from the LABEX Multi-Scale Modeling & Experimentation of Materials for Sustainable Construction (MMCD) through ANR Investments for the Future program ANR- 585 11-LABX-022-01.

Data availability
Data can be made available on request.

Declarations

Competing interests
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Received: 29 October 2024 Accepted: 2 December 2024
Published online: 24 December 2024

References

1. Griffith AA. Vi. the phenomena of rupture and flow in solids. Philos Trans A Math Phys Eng Sci. 1921;221(582–593):163–98.
2. Irwin GR. Analysis of stresses and strains near the end of a crack traversing a plate. J Appl Mech. 1957;24(3):361–4.
3. Hillerborg A, Modéer M, Petersson P-E. Analysis of crack formation and crack growth in concrete by means of fracture mechanics and finite elements. Cem Concr Res. 1976;6(6):773–81.
4. Peerlings RH, Borst R, Brekelmans W, Geers MG. Gradient-enhanced damage modelling of concrete fracture. Mech Cohes Frict Mater. 1998;3(4):323–42.
5. Moës N, Dolbow J, Belytschko T. A finite element method for crack growth without remeshing. Int J Numer Meth Eng. 1999;46(1):131–50.

6. Moës N, Stolz C, Bernard P-E, Chevaugeon N. A level set based model for damage growth: the thick level set approach. Int J Numer Meth Eng. 2011;86(3):358–80.
7. Bourdin B. Numerical implementation of the variational formulation for quasi-static brittle fracture. Interf Free Boundar. 2007;9(3):411–30.
8. Bourdin B, Francfort GA, Marigo J-J. Numerical experiments in revisited brittle fracture. J Mech Phys Solids. 2000;48(4):797–826.
9. Miehe C, Welschinger F, Hofacker M. Thermodynamically consistent phase-field models of fracture: variational principles and multi-field fe implementations. Int J Numer Meth Eng. 2010;83(10):1273–311.
10. Miehe C, Schaenzel L-M, Ulmer H. Phase field modeling of fracture in multi-physics problems. Part I. Balance of crack surface and failure criteria for brittle crack propagation in thermo-elastic solids. Comput Methods Appl Mech Eng. 2015;294:449–85.
11. Miehe C, Hofacker M, Schänzel L-M, Aldakheel F. Phase field modeling of fracture in multi-physics problems. Part II. Coupled brittle-to-ductile failure criteria and crack propagation in thermo-elastic-plastic solids. Comput Methods Appl Mech Eng. 2015;294:486–522.
12. Ambati M, Gerasimov T, De Lorenzis L. Phase-field modeling of ductile fracture. Comput Mech. 2015;55:1017–40.
13. Borden MJ, Verhoosel CV, Scott MA, Hughes TJ, Landis CM. A phase-field description of dynamic brittle fracture. Comput Methods Appl Mech Eng. 2012;217:77–95.
14. Bleyer J, Roux-Langlois C, Molinari J-F. Dynamic crack propagation with a variational phase-field model: limiting speed, crack branching and velocity-toughening mechanisms. Int J Fract. 2017;204(1):79–100.
15. Wu J-Y. A unified phase-field theory for the mechanics of damage and quasi-brittle failure. J Mech Phys Solids. 2017;103:72–99.
16. Chen L, Borst R. Phase-field modelling of cohesive fracture. Eur J Mech-A/Solids. 2021;90: 104343.
17. Yvonnet J, Xia L, Ghabezloo S. Phase field modeling of hydraulic fracturing with interfacial damage in highly heterogeneous fluid-saturated porous media. In: 14th US National Congress on Computational Mechanics 2017
18. Nguyen T-T, Réthoré J, Yvonnet J, Baietto M-C. Multi-phase-field modeling of anisotropic crack propagation for polycrystalline materials. Comput Mech. 2017;60:289–314.
19. Bleyer J, Alessi R. Phase-field modeling of anisotropic brittle fracture including several damage mechanisms. Comput Methods Appl Mech Eng. 2018;336:213–36.
20. Li P, Wu Y, Yvonnet J. A simp-phase field topology optimization framework to maximize quasi-brittle fracture resistance of 2d and 3d composites. Theoret Appl Fract Mech. 2021;114: 102919.
21. Nguyen TT, Yvonnet J, Zhu Q-Z, Bornert M, Chateau C. A phase field method to simulate crack nucleation and propagation in strongly heterogeneous materials from direct imaging of their microstructure. Eng Fract Mech. 2015;139:18–39.
22. Nguyen TT, Yvonnet J, Bornert M, Chateau C. Initiation and propagation of complex 3d networks of cracks in heterogeneous quasi-brittle materials: Direct comparison between in situ testing-microct experiments and phase field simulations. J Mech Phys Solids. 2016;95:320–50.
23. Nguyen TT, Yvonnet J, Bornert M, Chateau C, Bilteryst F, Steib E. Large-scale simulations of quasi-brittle microcracking in realistic highly heterogeneous microstructures obtained from micro ct imaging. Extr Mech Lett. 2017;17:50–5.
24. Patil R, Mishra B, Singh IV. A new multiscale XFEM for the elastic properties evaluation of heterogeneous materials. Int J Mech Sci. 2017;122:277–87.
25. Patil R, Mishra B, Singh I. An adaptive multiscale phase field method for brittle fracture. Comput Methods Appl Mech Eng. 2018;329:254–88.
26. Patil R, Mishra B, Singh IV. A multiscale framework based on phase field method and XFEM to simulate fracture in highly heterogeneous materials. Theoret Appl Fract Mech. 2019;100:390–415.
27. Nguyen LH, Schillinger D. The multiscale finite element method for nonlinear continuum localization problems at full fine-scale fidelity, illustrated through phase-field fracture and plasticity. J Comput Phys. 2019;396:129–60.
28. Triantafyllou SP, Kakouris EG. A generalized phase field multiscale finite element method for brittle fracture. Int J Numer Meth Eng. 2020;121(9):1915–45.
29. Feyel F. Multiscale FE^2 elastoviscoplastic analysis of composite structure. Comput Mater Sci. 1999;16(1–4):433–54.
30. Oliver J, Caicedo M, Huespe AE, Hernaįndez JA, Roubin E. Reduced order modeling strategies for computational multiscale fracture. Comput Methods Appl Mech Eng 2017;313:560–595
31. Benaimeche MA, Yvonnet J, Bary B, He Q-C. A k-means clustering machine learning-based multiscale method for anelastic heterogeneous structures with internal variables. Int J Numer Meth Eng. 2022;123(9):2012–41.
32. Bosco E, Kouznetsova VG, Geers MGD. Multi-scale computational homogenization-localization for propagating discontinuities using X-FEM. Comput Methods Appl Mech Eng. 2015;102(3–4):496–527.
33. Bharali R, Larsson F, Jänicke R. Computational homogenisation of phase-field fracture. Eur J Mech-A/Solids. 2021;88: 104247.
34. Nguyen N, Yvonnet J, Réthoré J, Tran AB. Identification of fracture models based on phase field for crack propagation in heterogeneous lattices in a context of non-separated scales. Comput Mech. 2019;63:1047–68.
35. Yuan J, He S, Chen C, Wang L. Phase-field fracture analysis of heterogeneous materials based on homogenization method. Acta Mech 2023;1–25
36. Hao S, Shen Y. An efficient parallel solution scheme for the phase field approach to dynamic fracture based on a domain decomposition method. Int J Num Methods Eng 2023;7405
37. Rannou J, Bovet C. Domain decomposition methods and acceleration techniques for the phase field fracture staggered solver. Int J Num Methods Eng 2023
38. Ziaei-Rad V, Shen Y. Massive parallelization of the phase field formulation for crack propagation with time adaptivity. Comput Methods Appl Mech Eng. 2016;312:224–53.
39. Heister T, Wheeler MF, Wick T. A primal-dual active set method and predictor-corrector mesh adaptivity for computing fracture propagation using a phase-field approach. Comput Methods Appl Mech Eng. 2015;290:466–95.
40. Heister T, Wick T. Parallel solution, adaptivity, computational convergence, and open-source code of 2d and 3d pressurized phase-field fracture problems. Pamm. 2018;18(1):201800353.

41. Heister T, Wick T. PFM-cracks: a parallel-adaptive framework for phase-field fracture propagation. Softw Impacts. 2020;6: 100045.
42. Jodlbauer D, Langer U, Wick T. Parallel matrix-free higher-order finite element solvers for phase-field fracture problems. Math Comput Appl. 2020;25(3):40.
43. Jodlbauer D, Langer U, Wick T. Matrix-free multigrid solvers for phase-field fracture problems. Comput Methods Appl Mech Eng. 2020;372: 113431.
44. Hewitt S, Margetts L, Shterenlikht A, Revell A. A massively parallel multiscale cafe framework for the modelling of fracture in heterogeneous materials under dynamic loading. Adv Eng Softw. 2020;139: 102737.
45. Liu X, Réthoré J, Lubrecht AA. An efficient matrix-free preconditioned conjugate gradient based multigrid method for phase field modeling of fracture in heterogeneous materials from 3d images. Comput Methods Appl Mech Eng. 2022;388: 114266.
46. Wu J-Y, Huang Y, Nguyen VP. On the BFGS monolithic algorithm for the unified phase field damage theory. Comput Methods Appl Mech Eng. 2020;360: 112704.
47. Miehe C, Lambrecht M. Algorithms for computation of stresses and elasticity moduli in terms of Seth-hill's family of generalized strain tensors. Commun Numer Methods Eng. 2001;17(5):337–53.
48. Ambati M, Gerasimov T, De Lorenzis L. A review on phase-field models of brittle fracture and a new fast hybrid formulation. Comput Mech. 2015;55:383–405.
49. Message Passing Interface Forum: MPI: A Message-Passing Interface Standard Version 4.1. 2023. https://www.mpi-forum.org/docs/mpi-4.1/mpi41-report.pdf
50. OpenMP Application Programming Interface: OpenMP Technical Report 12:Version 6.0 Preview 2. 2023. https://www.openmp.org/wp-content/uploads/openmp-TR12.pdf
51. NVIDIA, Vingelmann P, Fitzek FHP. CUDA, release: 10.2.89 2020. https://developer.nvidia.com/cuda-toolkit
52. Balay S, Gropp W, McInnes LC, Smith BF. Petsc, the portable, extensible toolkit for scientific computation. Argonne Natl Lab. 1998;2:17.
53. Saad Y, Schultz MH. Gmres: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J Sci Stat Comput. 1986;7(3):856–69.
54. Stiefel E. Methods of conjugate gradients for solving linear systems. J Res Nat Bur Standards. 1952;49:409–35.
55. Vorst HA. Bi-cgstab: a fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems. SIAM J Sci Stat Comput. 1992;13(2):631–44.
56. Dongarra JJ, Duff IS, Sorensen DC, Vorst HA. numerical linear algebra for high-performance computers. SIAM 1998
57. El Ouafa S, Vincent S, Le Chenadec V, Trouette B. Fully-coupled parallel solver for the simulation of two-phase incompressible flows. Comput Fluids. 2023;265: 105995.
58. Yang LT, Brent RP. The improved bicgstab method for large and sparse unsymmetric linear systems on parallel distributed memory architectures. In: Fifth International Conference on Algorithms and Architectures for Parallel Processing, 2002. Proceedings. IEEE 2002;pp. 324–328 .
59. Farrell P, Maurini C. Linear and nonlinear solvers for variational phase-field models of brittle fracture. Int J Numer Meth Eng. 2017;109(5):648–67.
60. Richardson CN, Sime N, Wells GN. Scalable computation of thermomechanical turbomachinery problems. Finite Elem Anal Des. 2019;155:32–42.
61. Stüben K. Algebraic multigrid (AMG): experiences and comparisons. Appl Math Comput. 1983;13(3–4):419–51.
62. Brandt A. Algebraic multigrid theory: the symmetric case. Appl Math Comput. 1986;19(1–4):23–56.
63. Falgout RD, Jones JE, Yang UM. The design and implementation of hypre, a library of parallel high performance preconditioners. In: Numerical solution of partial differential equations on parallel computers, Springer 2006;pp. 267–294 .
64. Falgout RD, Jones JE, Yang UM. Conceptual interfaces in hypre. Futur Gener Comput Syst. 2006;22(1–2):239–51.
65. Vanek P, Mandel J, Brezina M. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. Computing. 1996;56(3):179–96.
66. Adams MF, Bayraktar HH, Keaveny TM, Papadopoulos P. Ultrascalable implicit finite element analyses in solid mechanics with over a half a billion degrees of freedom. In: SC'04: Proceedings of the 2004 ACM/IEEE Conference on Supercomputing, IEEE 2004;pp. 34–34 .

Publisher's Note