



An Improved GPU-Parallelized 2D/3D Elastoplastic-Damage-Fracture Joint Framework for Combined Finite–Discrete-Element Program

Bo-nan Zhang¹ · Bo Han¹ · Qingsong Zhang¹

Received: 6 June 2023 / Accepted: 19 December 2023 / Published online: 21 February 2024
© The Author(s), under exclusive licence to Springer-Verlag GmbH Austria, part of Springer Nature 2024

Abstract

The combined Finite–Discrete-Element Method (FDEM) has demonstrated significant advantages in simulating the fracture and fragmentation process of brittle materials. However, the use of simple elastic-damage model alone is insufficient to meet the requirements for simulating the deformation of materials with ductility, and the computational cost of the discrete-element makes it a bottleneck in further development of FDEM. In this work, the 3D elastoplastic-damage-fracture joint mechanical model is proposed, and a combined algorithm has been implemented in the new FDEM package, INterdisciplinary Finite–Discrete-Element Program (INFIDEP). To improve computational efficiency and scale, a universal General-Purpose Graphics Processing Unit (GPGPU) parallelization INFIDEP code has been developed using the Compute Unified Device Architecture (CUDA) in C/C++. An improved parallel framework has been proposed, and a simplified contact detection algorithm that is more robust for non-uniform elements has been adopted. Using GPGPU-INFIDEP, we simulated Brazil's tensile strength (BTS) tests under quasi-static loading conditions and Taylor bar impact tests. The BTS tests demonstrated consistent stress–strain curves and accurate deformation trends. The speedup ratio stabilized at around 400–500 times on the NVIDIA A6000 GPU platform. The Taylor bar impact tests simulated based on the joint mechanical model well reflect the deformation trend of the specimen head and the impact fracture mode, which cannot be simulated by traditional FDEM and 2D plane-strain models. These results show that GPGPU-INFIDEP offers a valuable and powerful numerical tool for studying continuum–discontinuum deformation problems of brittle and elastoplastic materials in rock engineering.

Highlights

- The framework of 3D elastoplastic-damage-fracture joint mechanical model has been implemented in FDEM.
- An improved GPGPU parallel FDEM framework has been proposed, which improves the performance by a factor of 400 to 500 compared to a single-core CPU.
- The accuracy of the elastic-plastic model based on the FDEM framework has been verified, and GPGPU parallel acceleration has been implemented to improve its performance.
- The FDEM elastoplastic-damage-fracture joint model can better simulate the plastic deformation and cracking of ductile materials.

Keywords Combined finite–discrete-element method (FDEM) · GPU parallelized · Plasticity · Fracture · CUDA

1 Introduction

The combined finite-discrete-element method (FDEM) is a numerical technique that combines the advantages of continuum-based modeling approaches and discrete-element methods to overcome the inability of these methods to capture progressive damage and failure processes. Since its first proposal by Munjiza (1992), FDEM has shown significant

✉ Bo Han
bo.han@sdu.edu.cn

¹ School of Civil Engineering, Shandong University, Jinan, China

advantages in continuum-discontinuity calculations for brittle materials (Mahabadi et al. 2012; Knight et al. 2020). The latest theoretical development of FDEM has further expanded its applications, mainly including new contact detection (Zhao et al. 2018; Liu et al. 2021), contact interaction (Han et al. 2000a, 2000b), fluid–solid interaction (Munjiza et al. 2020; Yang et al. 2019; Xiang et al. 2012), parallel computing (Liu et al. 2022; Lisjak et al. 2018; Fukuda et al. 2020), large deformation (Rougier et al. 2020; Froment et al. 2020), and so on. With the introduction of parallel technology, the calculation efficiency of FDEM has increased by nearly a thousand times (Liu et al. 2021), and it has begun to be applied to simulate engineering-scale structural failures and geological disasters (Liu and Deng 2019; Deng et al. 2020). Multiphysics failure problem has also been successfully attempted (Yan et al. 2023, 2022b, 2022a).

Nevertheless, at present, the application of FDEM is still mainly used in elastic brittle fractured solid materials, such as granite, glass, ceramics, dry soil, etc. However, in practical engineering, many materials have plastic deformation before fracture (Rougier et al. 2020). For example, soft rock in geotechnical engineering has plastic deformation before the macro-shear fracture surface appears. When many metals deform under load, there is also a first stage of plastic deformation of the metal structure. That is, it will irreversibly yield under load before it actually breaks. Therefore, to describe pure plasticity or plastic behavior before fracture, it is necessary to introduce the plastic constitutive model into the framework of FDEM and combine it with the existing damage–fracture model.

In traditional FDEM, the Green–St. Venants strain tensor is used to describe homogeneous deformation (Munjiza 2004). Karantzoulis (2017) implemented the elastoplastic deformation simulation for the first time in the FDEM code "Solidity" and introduced the concept of Lagrangian finite strain tensor. Lei et al. (2020a) introduced novel plasticity modeling based on multiplicative decomposition and implemented it into the FDEM code "HOSS". Rougier et al. (2020) proposed a 2D finite strain-based elastoplastic and fracture framework based on the multiplicative decomposition model, which was applied to the HOSS code. Wu et al. (2023) studied the elastic–plastic deformation and fracture of rock caused by explosion load under the framework of 2D GPU-parallel FDEM program. However, in fact, most of the engineering problems are three-dimensional (3D). However, there is currently a lack of research on 3D elastoplastic–damage–fracture joint FDEM models. In this work, the deformation framework of FDEM has been extended to include 3D plastic deformation in conjunction with fracture and fragmentation. The platform was named as the INterdisciplinary Finite–Discrete-Element Program (INFIDEP).

Similar to the traditional FDEM, INFIDEP is limited by the computational cost after introducing the plastic deformation framework. To accelerate the FDEM simulation process, Liu et al. (2020) designed a GPU-parallel 2D FDEM. Lisjak et al. (2018) and Fukuda et al. (2020) proposed the GPGPU (general purpose graphic processing unit)-parallelized 3D FDEM versions based on OpenCL and CUDA (Compute Unified Device Architecture), respectively, which have been applied in commercial software Irazu. Yan et al. (2023) developed the GPU-parallel multi-physics coupling simulation software MultiFracS. Liu et al. (2021) proposed an improved GPGPU-parallelized 3D contact detection algorithm including neighbor search and fine search. Although a high speedup ratio can be obtained through GPU-parallel FDEM programs, there are few GPU-parallel studies on 3D FDEM with plasticity. The effect of GPU parallelism on 3D FDEM elastic–plastic models is unclear. Therefore, further research is needed on the GPU-parallel algorithm under the joint elastoplastic–damage–fracture framework. In this work, an improved GPU-parallel framework has been innovated, and the 2D/3D INFIDEP program based on the elastoplastic–damage–fracture joint framework was successfully implemented in GPU-parallel computing.

The paper is organized as follows: Sect. 2 introduces the FDEM theoretical details of INFIDEP. In particular, a simplified 2D/3D contact detection algorithm suitable for GPU parallelism is adopted. Section 3 introduces the plastic theory adopted in this work, and realized the coupling between the plasticity and fracture mechanisms. Section 4 presents INFIDEP's GPU-parallel framework and CUDA implementation, and investigates the performance of INFIDEP in a series of tests considering pure damage–fracture, pure plastic, and elastoplastic–damage–fracture models. Conclusions are given in Sect. 5.

2 Fundamental Theory of INFIDEP

The main idea of FDEM is to divide the continuous solid domain into several discrete elements by inserting no-thickness cohesion elements in the solid domain. At the same time, under the action of cohesive force, adjacent discrete elements can maintain continuous deformation. When the deformation between adjacent elements reaches the critical distance, the cohesive force elements fail to form cracks. Then, the adjacent elements are separated, and based on Newton's second law, the motion state of the discrete elements is solved by the contact algorithm.

2.1 Governing Equation

The motion of the element is controlled by the forces on the element nodes, and its governing equation is (Tatone and Grasselli 2015; Geomechanica 2019)

$$\mathbf{M} \frac{\partial^2 \mathbf{x}}{\partial t^2} + \mathbf{C} \frac{\partial \mathbf{x}}{\partial t} + \mathbf{F}_{int}(\mathbf{x}) - \mathbf{F}_{ext}(\mathbf{x}) - \mathbf{F}_c(\mathbf{x}) = \mathbf{0}, \tag{1}$$

where \mathbf{M} and \mathbf{C} are the nodal mass matrix and viscous damping matrix, respectively. \mathbf{x} is the node displacement vector. \mathbf{F}_{int} , \mathbf{F}_{ext} , and \mathbf{F}_c are node internal force vector, external force vector, and contact force vector respectively.

2.2 Solid Deformation in INFIDEP

The main difference between FDEM and the DEM is whether to consider the deformation of the material (Lei et al. 2016). For the deformation of finite elements, Munjiza et al. (2015) proposed the unified constitutive approach to store internal forces for the generalized material element inside the material package. Under this framework, both hyper-deformation and hypo-deformation-based constitutive law formulations become identical and converge to one so-called unified constitutive approach. In INFIDEP, aiming at material plastic deformation, based on the FDEM framework, the elastic-plastic constitutive model is introduced, and the elastoplastic-damage joint mechanical model is established.

For a homogeneous isotropic elastic material, based on the original Y-code (Munjiza 2004), the constitutive relation satisfied is

$$\mathbf{T} = \frac{1}{(\det \mathbf{F})^{2/3}} \left[\frac{E}{(1 + \nu)} \tilde{\mathbf{E}}_d + \frac{E}{(1 - 2\nu)} \tilde{\mathbf{E}}_s + 2\bar{\mu} \mathbf{D} \right], \tag{2}$$

where E and ν are the Young's modulus and Poisson's ratio, and $\bar{\mu}$ is a damping coefficient. $\tilde{\mathbf{E}}_d$ and $\tilde{\mathbf{E}}_s$ are the Green–St.Venant strain tensors due to shape change and volume change, respectively. \mathbf{F} is the deformation gradient, and \mathbf{D} is the rate of deformation. For homogeneous isotropic elastoplastic materials, the strain increment method is used in INFIDEP to solve the deformation. The cracking of the material adopts the cohesive force damage model proposed by Munjiza et al. (Munjiza et al. 1999, Lei, Zang and Munjiza 2010, Guo 2014), and it will be further described in Sect. 3.

2.3 Contact Model in INFIDEP

2.3.1 Contact Detection Algorithm

FDEM discretizes the solid domain into discrete elements; first of all, it needs to solve the problems of contact detection and contact force calculation. In the traditional Y-FDEM program, the No Binary Search (NBS) (Munjiza 2004) method is used for preliminary contact detection to establish contacting couples, and then, the precise contact judgment is made according to the element distance of the contacting

couples. According to the detected contacting couples, the contact force is calculated by the penalty function method based on the potential function, that is, the penetration amount between the elements is eliminated by introducing the "repulsion force" between the elements, thereby simulating the impenetrability between the contacting elements.

During the FDEM solution process, contact detection is the main consumer of computing resources. Improving the calculation efficiency of contact detection is the key point to speed up the FDEM solution. Munjiza et al. (2011) proposed the MR (Munjiza-Rougier) contact detection algorithm to solve the problem that NBS's RAM utilization depends on packaging density. On this basis, Lei et al. (2014) developed the MRCK contact detection algorithm and proposed the Virtual Parallel Machine for FDEM. For the effect of element size on the computational efficiency of the NBS method, Zhao et al. (2018) proposed an element grouping method. To develop an FDEM framework suitable for GPU parallelism, Liu et al. (2022) abandoned the linked list-based data structure (which was used in NBS, MRCK, and other methods) and proposed a more efficient 3D contact detection algorithm based on the method of boundary box and region partitioning, which demonstrated significant advantages in simulating non-uniform tetrahedral element models. Here, based on Liu's method, INFIDEP utilizes a simplified Domain Decomposition and Bounding Box (DDBB) based contact detection algorithm to optimize computational efficiency for models with various sizes of elements. The buffering parameters in the algorithm were simplified to enhance its stability. And the global search (Zheng et al. 2017) is utilized to identify all potential contacting couples, which involves the following steps:

Step 1: Space decomposition

For the simplified DDBB algorithm, first, the spatial domain needs to be decomposed into cells of the same size (Fig. 1), and each cell is represented by a set of integer coordinates ($c_i, i = 1, 2, 3$). In the direction of each coordinate axis, the number of cells is $n_i, i = 1, 2, 3$

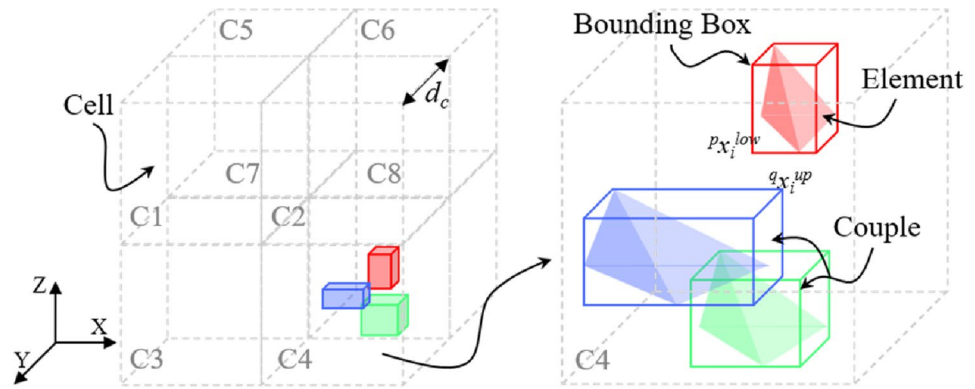
$$n_i = \text{Int} \left(\frac{x_{i,\max} - x_{i,\min}}{d_c} \right) + 1, \tag{3}$$

where $x_{i,\max}$ is the upper bounds of the contact domain, and $x_{i,\min}$ is the lower bounds ($i = 1, 2, 3$). *Int* denotes the integerization truncating after the decimal point in the floating point. d_c is the cell size, which can be given by

$$d_c = \lambda \frac{1}{N} \sum_{i=1}^N d_n; \tag{4}$$

here, λ is the factor related to the cell size, which is typically chosen as 1.0–2.0 depending on the allocated memory size of GPU. N is the total number of four-node tetrahedral

Fig. 1 Three discrete tetrahedral elements with bounding boxes located in the domain subdivided by 8 cells



elements and d_n is the circumscribed sphere diameter of n th tetrahedral element.

Step 2: Bounding boxes

The bounding boxes method is used here to avoid performing search every time step. Construct the bounding box of each tetrahedral element as an axial cuboid. Then, the upper and lower bounds of each tetrahedral element in a potential discrete-element couple are

$$\begin{aligned} n x_i^{up} &= \max (n p_{ij}) \\ n x_i^{low} &= \min (n p_{ij}); \end{aligned} \tag{5}$$

here, $n x_i^{up}$ is the upper boundary of the n th tetrahedral element ($i = 1, 2, 3$). $n x_i^{low}$ is the lower boundary of the n th tetrahedral element ($i = 1, 2, 3$). $n p_{ij}$ is the j th node coordinate of the n th 4nodes-tetrahedron element ($i = 1, 2, 3; j = 1, 2, 3, 4$).

The element bounding box boundaries are then mapped into a regular cell, yielding integer coordinates for each bounding box

$$\begin{aligned} n c_i^{up} &= \text{Int} \left(\frac{n x_i^{up} - x_{i,\min} + 2\delta_b}{d_c} \right) \\ n c_i^{low} &= \text{Int} \left(\frac{n x_i^{low} - x_{i,\min} + 2\delta_b}{d_c} \right); \end{aligned} \tag{6}$$

here, c is the boundary integer coordinate of the cell to which the n th tetrahedral element in the potential couples is mapped, $i = 1, 2, 3$. δ_b is size of the buffer around each finite element to modify the upper and lower bound, usually 1/5 of the size of the smallest finite element.

Step 3: Determination of the potential contacting couples

In search for tetrahedral elements in each cell, if two tetrahedral element bounding boxes have a common cell, they are considered to be in contact, and then, these two elements are recorded as a couple. That is, for any two tetrahedral elements

p, q in the same cell, a couple is added if the following formula is satisfied

$$p x_i^{up} - q x_i^{low} \geq tol \text{ or } q x_i^{up} - p x_i^{low} \geq tol, \tag{7}$$

where $i = 1, 2, 3$. tol is a buffer zone which is used to reduce the error generated by floating-point number operations. Its value has a certain relationship with the element size, and here, $tol = 1.0e-5$. It should be noted that a single element may be mapped to multiple cells. Therefore, it is necessary to find the cell for judgment in the following way before detecting the intersection of bounding boxes:

$$c_i = \min (p c_i^{up}, q c_i^{up}). \tag{8}$$

In the simplified DDBB method mentioned above, the calculation of each cell is independent when calculating the tetrahedral elements contacting couples in the cell, which is suitable for GPU-parallel computing. Moreover, the difference in element size in this method has little influence on the search efficiency, and it has certain advantages for calculation of complex engineering problems.

2.3.2 Contact Force Calculation

The calculation of the contact force between discrete elements in INFIDEP adopts the potential function method proposed by Munjiza (2004), and based on the potential function method, the two elements in a couple can be recorded as contactor E_c and target E_t , respectively, the overlapping area of the two elements is recorded as S , and the contact boundary is recorded as Γ . The contact force generated on the infinitesimal contact area dA can be expressed as

$$\mathbf{f}_c = \sum_{i=1}^n \sum_{j=1}^m \int_{\Gamma_{E_i \cap E_j}} \mathbf{n}_{\Gamma_{E_i \cap E_j}} (\varphi_{c_i} - \varphi_{t_j}) p d\Gamma, \tag{9}$$

where φ_c and φ_t are the potential functions of the contact elements at the contact points of the contactor and the target, respectively. \mathbf{n} is the outer normal direction of Γ , and p is the penalty coefficient. Further based on the penalty function method, the contact force can be expressed as

$$\begin{aligned} \mathbf{f}_n &= -\mathbf{n} \int_0^{L_p} p\varphi(l)dl \\ \mathbf{f}_t &= \mu \frac{\|\mathbf{f}_n\| \mathbf{v}_r}{\|\mathbf{v}_r\|}; \end{aligned} \tag{10}$$

here, f_n is the normal force, and f_t is the tangential force. L_p is the penetration depth, φ is the potential function along the boundary of the target element, \mathbf{v}_r is the relative velocity of the contact interface, and μ is the friction coefficient in the shearing process. For the FDEM contact algorithm, please refer to the works of Munjiza (2004) and Lei et al. (2020b).

3 The Elastoplastic-Damage-Fracture Joint Mechanical Model in INFIDEP

The finite-element method based on large strain and large displacement is adopted by FDEM, which uses a mechanical method combined with a so-called material embedding non-Cartesian coordinate system (Munjiza et al. 2015). It achieves multiplicative decomposition through a mechanical approach based on deformation functions expressed in material embedding curvilinear coordinates. The classic set of deformation functions has been replaced by two sets of deformation functions, deformation gradients obtained using tensor calculus, and stretch tensors represented by a user-supplied material embedding vector basis. In the FDEM deformation framework, Rougier et al. (2020) extended the deformation of plane-strain elements to include plastic deformation, fracture, and crushing based on the stretch tensor. Here, under the deformation framework of FDEM, we constructed a plastic deformation simulation framework, introduced the Johnson–Cook (J–C) plasticity model and Drucker–Prager (D–P) yield criterion, and extended the tetrahedral element deformation to plastic, fracture, and fragmentation.

3.1 Elastoplastic Constitutive Model in INFIDEP

The theoretical framework for the elastoplastic-damage-fracture joint mechanical model in INFIDEP mainly includes the J_2 flow theory for elastic–plastic constitutive models, the radial return algorithm, the hardening flow rule, and the damage-fracture model. The elastic–plastic model is based on finite deformation finite-element

analysis, while the damage-fracture model is based on cohesive crack elements embedded on the boundaries of discrete finite elements. More specific details on the theoretical framework and algorithm are presented below.

3.1.1 Time Integration Algorithm of J_2 Plasticity

Time integration algorithm based on finite difference method is used in INFIDEP (Han et al. 2015). The elastic stress–strain equation is written in the following incremental form (Belytschko et al. 2014):

$$\boldsymbol{\sigma}^{\sigma J} = \mathbf{C}_{el}^{\sigma J} : \mathbf{D}^e, \tag{11}$$

where $\boldsymbol{\sigma}$ is an objective stress rate, and \mathbf{C} is the linear isotropic fourth order elastic tensor given by (Belytschko et al. 2014)

$$\mathbf{C}_{el}^{\sigma J} = K\mathbf{I} \otimes \mathbf{I} + 2G\mathbf{I}^{dev}, \mathbf{I}^{dev} = \mathbf{I} - \frac{1}{3}\mathbf{I} \otimes \mathbf{I}; \tag{12}$$

K is the Bulk modulus, G is the shear modulus, \otimes is the Dyadic product, and \mathbf{I} is the second-order identity tensor. And \mathbf{D}^e is the elastic part of the rate of deformation tensor \mathbf{D} which is (Belytschko et al. 2014)

$$\mathbf{D} = \mathbf{D}^e + \mathbf{D}^p, \tag{13}$$

where \mathbf{D}^p is the plastic part of \mathbf{D} . J_2 plasticity is based on the von Mises yield surface (Lubliner 2008). The scalar yield function f is given by

$$f(\boldsymbol{\sigma}, \mathbf{q}) = \bar{\sigma} - \sigma_Y(\bar{\boldsymbol{\epsilon}}^p, \dot{\boldsymbol{\epsilon}}^p) = 0, \tag{14}$$

where $\boldsymbol{\sigma}$ is Cauchy stress and \mathbf{q} is internal variable. Here, $q_1 \equiv \bar{\boldsymbol{\epsilon}}^p$ and $q_2 \equiv \dot{\boldsymbol{\epsilon}}^p$. $\bar{\boldsymbol{\epsilon}}^p$ is the equivalent plastic strain and $\dot{\boldsymbol{\epsilon}}^p$ is the equivalent plastic strain rate. The von Mises equivalent stress, or effective stress, defined from the deviatoric stress \mathbf{S} . σ_Y is the current yield stress of uniaxial tension, depending on $\bar{\boldsymbol{\epsilon}}^p$ and $\dot{\boldsymbol{\epsilon}}^p$. $\bar{\boldsymbol{\epsilon}}^p$ is given by (Ming and Pantalé 2018)

$$\bar{\boldsymbol{\epsilon}}^p = \int_0^t \dot{\bar{\boldsymbol{\epsilon}}^p} dt = \int_0^t \sqrt{\frac{2}{3} \mathbf{D}^p : \mathbf{D}^p} dt \tag{15}$$

$$\mathbf{D}^p = \gamma \mathbf{n}, \tag{16}$$

where γ is a scalar representing the flow intensity and \mathbf{n} is a second-order tensor (the unit normal to the flow stress determined exclusively in terms of the trial elastic stress Simo and Hughes 2006; Ming and Pantalé 2018). Then, we can obtain

$$\dot{\bar{\boldsymbol{\epsilon}}^p} = \sqrt{\frac{2}{3}}\gamma \text{ and } \bar{\boldsymbol{\epsilon}}^p = \sqrt{\frac{2}{3}} \int_0^t \gamma dt = \sqrt{\frac{2}{3}}\Gamma; \tag{17}$$

Γ is the plastic corrector. The plasticity model described here above has to be integrated in time with respect to an incremental objective algorithm.

In INFIDEP, because the time increment is very small due to the explicit time integration scheme, it is to use a straightforward evaluation of the equivalent plastic strain increment based on the strain tensor without requiring any local iterative algorithm (Ming and Pantalé 2018).

3.1.2 Radial Return Mapping Algorithm

The return mapping algorithm (Wilkins 1963; Dunne and Petrinic 2005) is used for the time integration of J_2 plasticity with isotropic hardening which is shown in the Box 1.

3.1.2.1 Box 1. The return mapping algorithm Step 1.
 Set initial values:
 $k = 0$:

$$\epsilon^{p(0)} = \epsilon_n^p, \bar{\epsilon}^{(0)} = \bar{\epsilon}_n, \boldsymbol{\sigma}^{(0)} = \mathbf{C} : (\epsilon_{n+1} - \epsilon^{p(0)}) \quad (18)$$

$$f^{(0)} = \bar{\sigma}_{try} - \sigma_Y(\bar{\epsilon}^{p(0)}, \dot{\bar{\epsilon}}^{p(0)}) = \left[\frac{3}{2} \mathbf{S}^{(0)} : \mathbf{S}^{(0)} \right]^{1/2} - \sigma_Y(\bar{\epsilon}^{p(0)}, \dot{\bar{\epsilon}}^{p(0)}) \quad (19)$$

where \mathbf{S} is the deviatoric stress.

Step 2. Check the yield condition at the k -th ($k \geq 1$) iteration:

$$f^{(k)} = \bar{\sigma}_{try} - \sigma_Y(\bar{\epsilon}^{p(k)}, \dot{\bar{\epsilon}}^{p(k)}) = \left[\frac{3}{2} \mathbf{S}^{(k)} : \mathbf{S}^{(k)} \right]^{1/2} - \sigma_Y(\bar{\epsilon}^{p(k)}, \dot{\bar{\epsilon}}^{p(k)}) \quad (20)$$

If $f^{(k)} \leq 0$, $\mathbf{S}^{(k)} = \mathbf{S}^{(k-1)}$ and end loop.
 Else, go to step 3.
 Step 3. Update internal variables:

$$\mathbf{S}^{(k)} = \mathbf{S}^{(k-1)} - 2G\Gamma \mathbf{n} \quad (21)$$

$$\bar{\epsilon}^{p(k)} = \bar{\epsilon}^{p(k-1)} + \sqrt{\frac{2}{3}}\Gamma, \dot{\bar{\epsilon}}^{p(k)} = \frac{1}{\Delta t} \sqrt{\frac{2}{3}}\Gamma \quad (22)$$

where G is the shear modulus.

Combination of equations (20~22) leads to the following form of the consistency parameter $f^{(k)}$:

$$f(\Gamma)^{(k)} = \bar{\sigma}_{try} - \sqrt{6}G\Gamma - \sigma_Y(\Gamma) \quad (23)$$

where the only unknown value is the scalar Γ , already defined in equation (17). Calculate the value of Γ according to the bisection method, and $\Gamma \in [0, \bar{\sigma}_{try} / (\sqrt{6}G)]$. Then according to the final values of $\mathbf{S}^{(k)}$, updated the final stress:

$$\boldsymbol{\sigma}^{(k)} = \mathbf{S}^{(k)} + \frac{1}{3} \text{trace}(\boldsymbol{\sigma}^{(k)}) \mathbf{I} \quad (24)$$

Then $k = k+1$, go to step 2.

3.1.3 The Johnson–Cook Hardening Flow Law

In INFIDEP, the Johnson–Cook hardening flow law has been implemented (Johnson and Cook 1985). It is probably the most widely used flow law for the simulation of high strain rate deformation processes taking into account plastic strain, plastic strain rate, and temperature effects. Only the effects of plastic strain and plastic strain rate are considered here. The general formulation is given by the following equation (Ming and Pantalé 2018):

$$\sigma_Y = \left(A + B\bar{\epsilon}^{p^n} \right) \left[1 + C \ln \left(\frac{\dot{\bar{\epsilon}}}{\dot{\bar{\epsilon}}^0} \right) \right]; \quad (25)$$

here, $\dot{\bar{\epsilon}}^0$ is the reference strain rate. A, B, C, n are the constitutive flow law parameters. The dependence on the plastic strain rate is only taken into account if $\dot{\bar{\epsilon}}^p \geq \dot{\bar{\epsilon}}^0$.

3.2 Elastoplastic-Damage-Fracture Joint Mechanical Model

Within the FDEM framework, fracture is allowed to occur at the boundaries of the finite elements, as shown in Fig. 2. The fracture process of the material is represented by the failure of cohesive elements (Guo 2014). For four-node tetrahedral elements obtained through finite-element meshing, when cohesive elements are embedded, the tetrahedral elements that originally share nodes separate. The six nodes formed at the boundaries of adjacent tetrahedrons are the nodes of the cohesive element ($A_1, B_1, C_1, A_2, B_2, C_2$) (Fig. 2a).

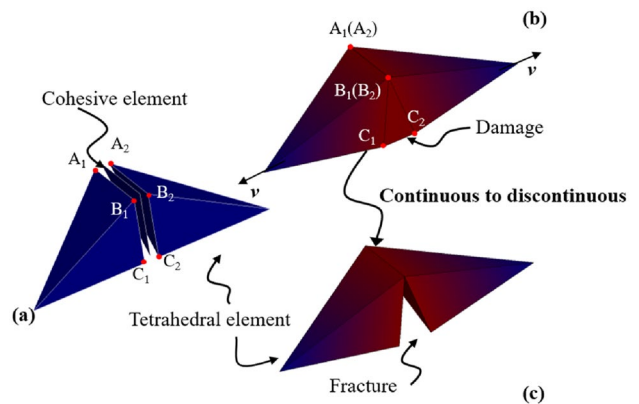


Fig. 2 Material from continuous to discontinuous in INFIDEP: **a** cohesive elements embedded in tetrahedral elements, **b** cohesion element damage, and **c** cohesive elements fail, while tetrahedral elements are discrete

The continuity between the tetrahedral elements is therefore constrained by the cohesive element $A_1B_1C_1-A_2B_2C_2$. The thickness of the initial cohesive element is zero, that is, the adjacent faces of the two tetrahedral elements coincide with the cohesive element.

The fracture types of cohesive elements in INFIDEP are Mode-I (stress orthogonal to the local plane of the crack surface), Mode-II (stress parallel to the crack surface but orthogonal to the crack front), and Mode-III (stress parallel to the crack surface and to the crack front). Taking Model-I as an example (Fig. 3), fracture initiation and propagation occur when the opening of a cohesive element, o , reaches a critical value, o_p , corresponding to the intrinsic tensile strength of the element, f_t . As a cohesive element is opened beyond o_p , the normal stress begins to decrease. At this stage, adjacent tetrahedral elements are still constrained by the cohesive elements, which we call "damage" (Fig. 2b). When the opening of a cohesive element, o , reaches o_c , the cohesive elements completely fail and are deleted, and the separation between the finite elements produces a crack. At this stage, adjacent tetrahedral elements are no longer constrained by cohesive elements and are completely separated, which we call "fracture" (Fig. 2c). The behavior of these cohesive elements follows the combined single and smeared crack model was introduced by Munjiza et al. (1999), and a detailed description of fracture modeling within FDEM is outside the scope of this work; however, the interested reader can refer to (Lisjak et al. 2018; Fukuda et al. 2020) for more information.

We introduce the plasticity model into the finite-element constitutive model of FDEM, which means that the plastic deformation here is the behavior of the finite element, and the material damage is the property of the cohesive element.

The material's plastic deformation can now be accommodated by two competing mechanisms:

- (1) Plasticity inside the finite element in the form of plastic flow.
- (2) Damage of the cohesive element in the form of discrete fractures.

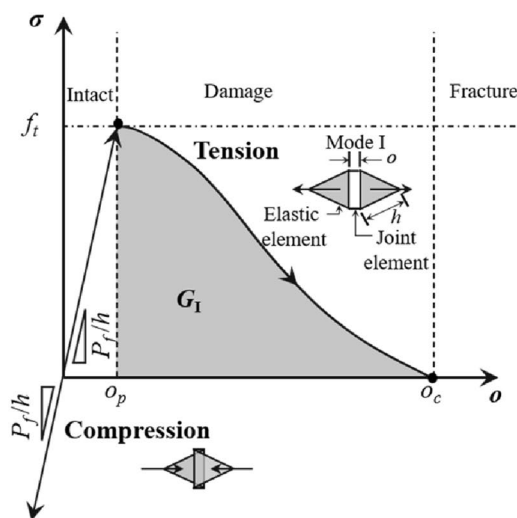
Therefore, there is a need to coordinate these two mechanisms for (quasi-)brittle materials, so that they act together to represent the correct physical behavior of the material. When calibrating material parameters, plasticity reflects the macroscopic yield strength of the material, and damage reflects the microscopic fracture strength of the material. They need to be unified according to the specific properties of the material, and the criterion used to "switch-on" the discrete fractures at the boundaries of the finite elements is to introduce a smooth transition from distributed plasticity to localized damage, as is shown in Fig. 4. It should be noted that distributed plasticity is caused by finite elements, while localized damage is caused by cohesive elements.

3.3 Joint Mechanical Model Verification

To demonstrate the capabilities introduced by this new formulation, two case studies are reported in here: (a) 3D cube test without fracturing and (b) 3D cube test with fracturing. The results reported in this section were obtained using INFIDEP.

3.3.1 3D Cube Test: Case Without Fracturing

In this section, the performance of the Johnson–Cook law programmed in INFIDEP is compared with the



- o Cohesive element opening
- f_t Cohesive element tensile strength
- P_f Fracture penalty
- h Nominal element edge length
- G_I Mode I fracture energy release rate

Fig. 3 Cohesive model for Mode I of INFIDEP FDEM code

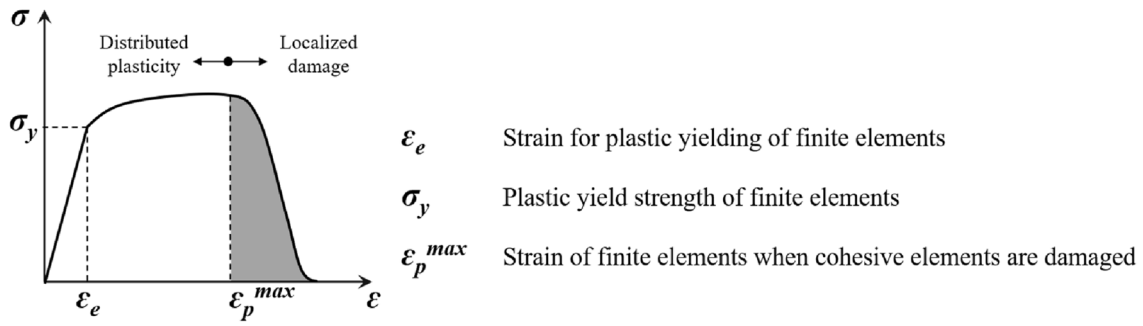


Fig. 4 General depiction of distributed plasticity (inside the finite elements) and localized damage (inside the cohesive elements)

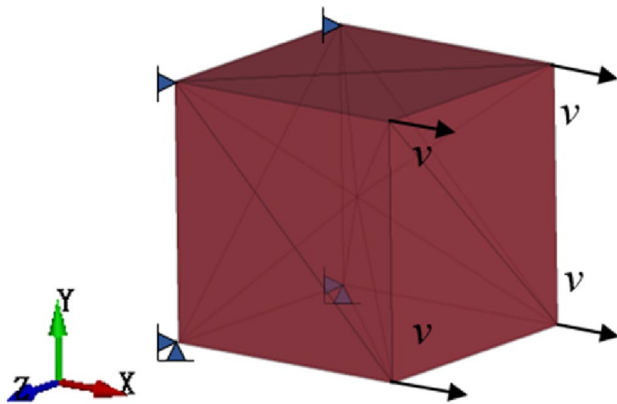


Fig. 5 The 3D cubit model

performance of the Abaqus/Explicit (Smith 2009). A 3D cube (1 × 1 × 1 m) was used as a benchmark for the plasticity model developed in this work. The setup of the problem is shown in Fig. 5. For the boundary conditions, the vertices on the left side of the cube are fixed constraints, and the vertices on the right side move in the positive direction of the X-axis at a constant velocity of 0.05 m/s. The calculation time is 4 s, and the total node displacement is 0.2 m. The simulations were conducted using four-node tetrahedral elements (C3D4). The material, a 42CrMo4 steel, has been selected for all those tests, and material properties are reported in Table 1 (Ming and Pantalé 2018). And Fig. 6 shows the evolution of the von Mises stress with displacement-x. It can be observed that the evolution for both models (Abaqus and INFIDEP) are virtually on top of each other.

3.3.2 3D Cube Test: Case with Fracturing

The same 3D cube test described in the previous case was used, but this time the cube was allowed to fracture. To accurately describe the change process of materials from continuous to discontinuous, FDEM considers both macroscopic mechanical parameters and microscopic mechanical parameters. In general, mechanical parameters measured by standard laboratories cannot be directly used as input parameters for FDEM models, and need to be determined through a calibration process (Tatone and Grasselli 2015). The tensile strength of 42CrMo4 steel is usually 800–1200 MPa. After parameter calibration, the ultimate tension strength f_t , and the fracture energy G_s , of the material, were set to 1.6GPa and 1N/m.

Figure 7 shows the fracture results of the elastoplastic-damage-fracture cube. The green part in the figure is the generated

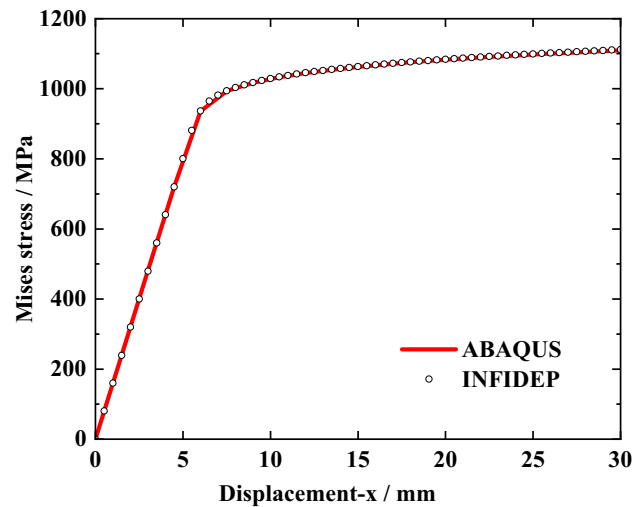


Fig. 6 The evolution of the von Mises stress with displacement-x

Table 1 Material properties of the 42CrMo4 steel

E/GPa	ν	A/MPa	B/MPa	C	n	$\frac{\sigma_0}{\epsilon}$	$\rho/\text{kg}\cdot\text{m}^{-3}$
206.9	0.29	806	614	0.0089	0.168	1	7830

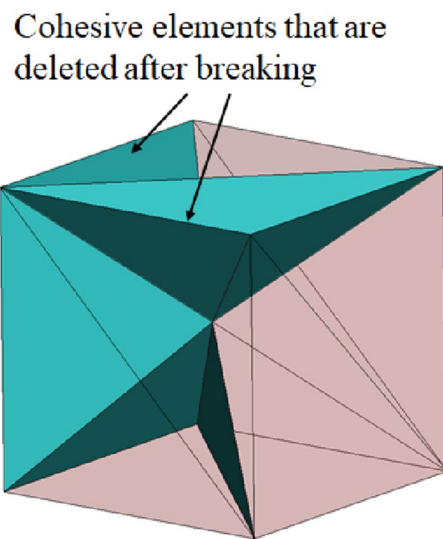


Fig. 7. 3D cube cracking results based on elastoplastic-damage-fracture mechanical model

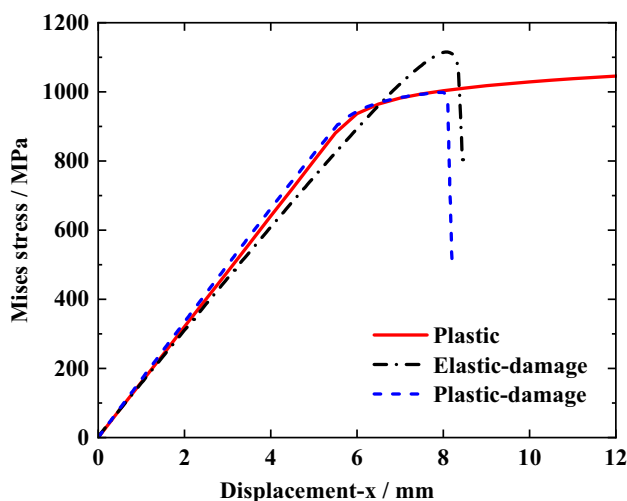


Fig. 8 The evolution of the von Mises stress with displacement-x with different models

crack, which is represented by the deleted cohesive elements at the element boundary. For the impact of the proposed model on Misses stress, we compared the results of the pure plastic model, elastic-damage-fracture model and elastoplastic-damage-fracture model, as shown in Fig. 8. It can be seen that the elastoplastic-damage-fracture model well simulates the material fracture and the irreversible yield before the fracture. The pure plastic model cannot simulate the phenomenon of local material failure, and the elastic-damage model cannot

reflect the plastic flow process. It should be noted that the cohesive element has a certain influence on the stiffness of the model. Therefore, in the simulation considering the cracking process, it is necessary to calibrate the elastic modulus, and use the damping force to reduce the instability of the fracture calculation.

3.4 Scalability of the Joint Mechanical Framework

Under the elastoplastic-damage-fracture mechanical framework of FDEM, different elastoplastic models can be easily developed for finite elements. Here, a pressure-dependent constitutive model with combined multilinear kinematic and isotropic hardening is developed in INFIDEP.

3.4.1 The Drucker–Prager (D–P) Model in INFIDEP

The chosen pressure-dependent yield function is the Drucker–Prager yield function (Drucker and Prager 1952). The Drucker–Prager yield function is written in (Smith 2009) notation as

$$f = t - p \tan \theta - d = 0, \tag{26}$$

where t is a pseudo-effective stress, θ is the slope of the linear yield surface in the p - t stress plane and is commonly referred to as the friction angle of the material, p is the hydrostatic pressure, d is the cohesion of the material.

The flow potential, g , for the linear D–P model is defined as

$$g = t - p \tan \psi, \tag{27}$$

where ψ is the dilation angle in the p - t plane. Associated flow results from setting $\psi = \theta$. Therefore, the original D–P model leads to

$$t = \sqrt{3J_2}. \tag{28}$$

Thus, the yield function could be given by

$$f = \sqrt{3J_2} + \frac{1}{3}I_1 \tan \theta - d, \tag{29}$$

where I_1 is the first stress invariant. To conveniently compare ABAQUS Drucker–Prager material property variables with those used by Wilson (2003) in their material testing, let

$$a = \frac{1}{3} \tan \theta. \tag{30}$$

Finally, Eq. (29) can be written as

$$f = \sigma_{eff} - 3ap - \sigma_{ys}(\bar{\epsilon}^p) = 0. \tag{31}$$

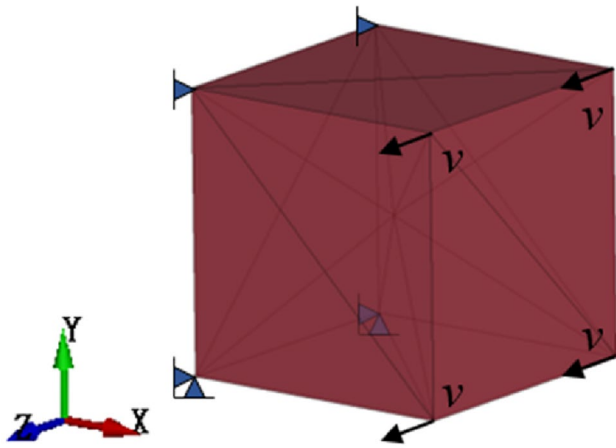


Fig. 9. 3D cube model for D–P constitutive model verification

The combined kinematic and isotropic hardening models are used in INFIDEP. Assuming a linear combination of the two hardening types, a scalar parameter, β , can be defined which determines the amount of each type of hardening with $0 \leq \beta \leq 1$. A value of $\beta = 1$ indicates only isotropic hardening, and a value of $\beta = 0$ indicates only kinematic hardening. There has been a lot of research on algorithm development of D–P model. The model used here is similar to the extended D–P model. For more details about the D–P model theory, the interested reader can refer to (Wilson 2003; Smith 2009; Karantzoulis 2017) for more information.

3.4.2 3D Cube Test

The 3D cube model has the same dimensions as in Sect. 3.3, and the boundary conditions are as shown in Fig. 9, the

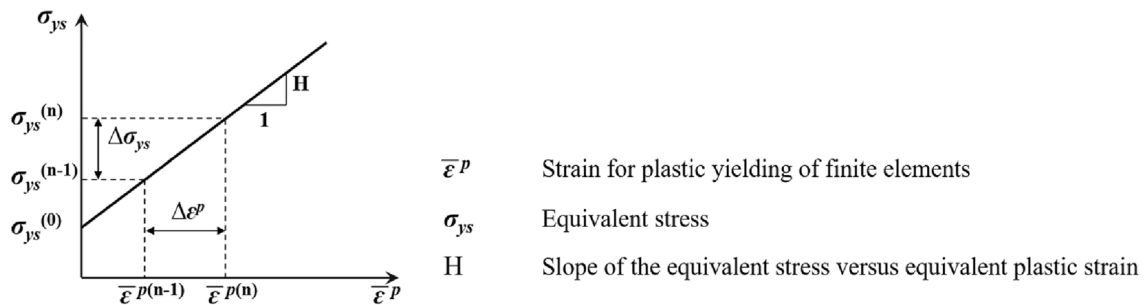


Fig. 10 Illustration of the relationship between yield stress and equivalent plastic strain for the bilinear hardening case

Table 2 Material properties of the D–P model

E/GPa	ν	a	β	$\rho/\text{kg}\cdot\text{m}^{-3}$	$\sigma_{ys}^{(0)}/\text{MPa}$	$ \bar{\epsilon}^{p(0)} $	$\sigma_{ys}^{(1)}/\text{MPa}$	$ \bar{\epsilon}^{p(1)} $
30	0.3	806	614	2700	50	0	100	1

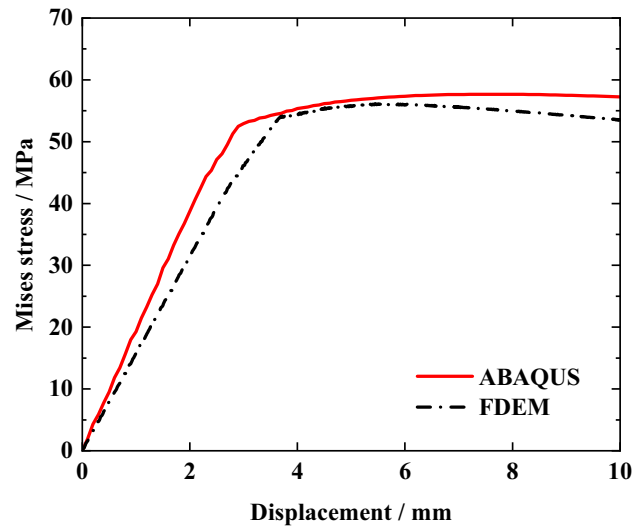


Fig. 11 The evolution of the von Mises stress with displacement-z

vertices on the left side of the cube are fixed constraints, and the vertices on the right side move in the positive direction of the Z-axis at a constant velocity of 0.05 m/s. The calculation time is 0.5 s, and the total node displacement is 10 mm. The simulations were conducted using four-node tetrahedral elements (C3D4).

For the material, test data are entered as tables of yield stress values versus equivalent plastic strain at different equivalent plastic strain rates, one table per strain rate. Compression data are more commonly available for geological materials, whereas tension data are usually available for polymeric materials. The relationship between yield stress and equivalent plastic strain for the bilinear hardening case is shown in Fig. 10, where H is the slope of

Table 3 The microscopic parameters used in INFIDEP D–P model case

Parameter	Unit	Value
Tensile strength	MPa	1
Shear strength	MPa	10
Fracture energy (Mode I and Mode II)	J/m ²	5
Contact friction coefficient	–	0.1
Contact penalty	GPa	300
Cohesive element penalty	GPa	300

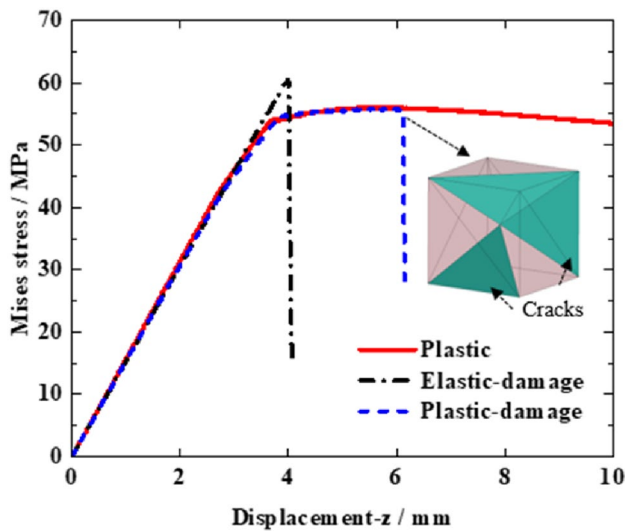


Fig. 12 The evolution of the von Mises stress with displacement-x with different models

the equivalent stress versus the equivalent plastic strain. In INFIDEP's D–P plasticity model, the material parameters that need to be input include elastic modulus, Poisson's ratio, density, α , β , and the data table of the relationship between yield stress and equivalent plastic strain. The material parameters used here are shown in Table 2. It should be noted that the parameters here are only used

to compare with ABAQUS results to verify the correctness of the model, and are not calibrated for specific rock material tests. The evolution of the von Mises stress with displacement-z is shown in Fig. 11. It can be observed that the evolution for both models (Abaqus and INFIDEP) is virtually on top of each other.

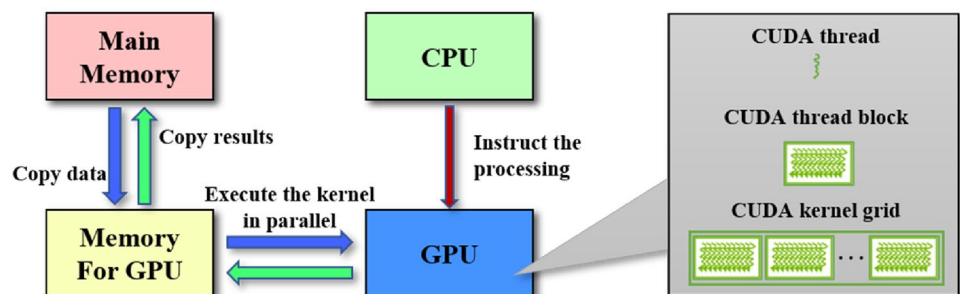
Further, we consider a joint mechanical model. The selection of microscopic parameters is shown in Table 3. For the effect of the proposed model on Misses stress, we compared the results of the pure plastic model, elastic-damage-fracture model and elastoplastic-damage-fracture model, as shown in Fig. 12. It can be seen that the joint mechanical model successfully simulates the plastic stage before material damage and fracture, and can adapt to a wider range of material deformation types than the pure plastic model and the traditional FDEM model. This also demonstrates that our proposed joint mechanical framework can easily extend the plastic constitutive models of finite elements.

4 GPU Based Parallelization of 2D/3D INFIDEP by CUDA C + +

As mentioned in the introduction, for engineering-scale numerical calculations, the real-time requirements for calculation efficiency are too high to be a problem. The bottleneck of single-core CPU processing massive data makes it difficult to meet the needs of large-scale computing. For the FDEM program, there are mainly the following problems to realize the engineering scale:

- (1) FDEM is based on the explicit analysis method. To ensure the calculation convergence, a smaller time step is required, and the total time for simulating material failure is longer.
- (2) After the material changes from continuous to discontinuous, new boundaries are generated, elements are separated from each other, and the contact detection between discrete elements is time-consuming.

Fig. 13 CUDA processing flow and kernel execution on GPU



- (3) After INFIDEP introduces the plastic frame, the requirement of nonlinear iterative calculation accuracy on the time step is further increased.

To speed up the simulation process of the 2D/3D INFIDEP, a parallel computation scheme based on the NVIDIA® GPGPU accelerator is incorporated. In our case, the computation on the GPGPU device is controlled through NVIDIA's CUDA C++ (Zeller 2011), which is essentially an ordinary C/C++ programming language with several extensions that make it possible to leverage the power of the GPGPU in the computations. The CUDA programming model uses abstractions of "threads", "blocks", and "grids" (Fig. 13). Each CUDA block is executed by one streaming multiprocessor (SM) and cannot be migrated to other SMs in GPU (except during preemption, debugging, or CUDA dynamic parallelism). One SM can run several concurrent CUDA blocks depending on the resources needed by CUDA blocks. Each kernel is executed on one device and CUDA supports running multiple kernels on a device at one time. Only a single "grid" system is used in this study.

4.1 Establishment of GPU-INFIDEP Parallel Framework

At present, there are two main methods for research on the parallel architecture of GPU-FDEM: one uses CPU–GPU joint computing for parallelism, the main program is still calculated in the CPU (Schiava D'Albano 2014; Lukas et al. 2014), and GPU-parallel acceleration is used for time-consuming functions. Although this architecture can take advantage of the advantages of CPU logic processing and GPU-parallel computing, it requires frequent data transfer, which increases the complexity of the algorithm and has a lot of communication time. The second type mainly relies on the GPU for calculation (Lisjak et al. 2018; Fukuda et al. 2020), and the CPU is responsible for reading and storing user input data and working with re-mesh. The main calculations are done in the GPU, and when the results need to be output, they are passed from the device to the host. This architecture saves the time of data transmission and has high efficiency, so it is widely used.

However, the traditional FDEM program is written in C language. To realize the transfer of variables in subroutines, a large number of secondary pointers and tertiary pointers are used. When using an array on a device, memory allocation and assignment can be performed through *cudaMalloc* and *cudaMemcpy* for one-dimensional arrays. However, for two-dimensional and three-dimensional arrays, when transferring from host to device, it usually needs to be converted into a one-dimensional array for data transfer or use *cudaMallocPitch* to apply for memory. Continuous array

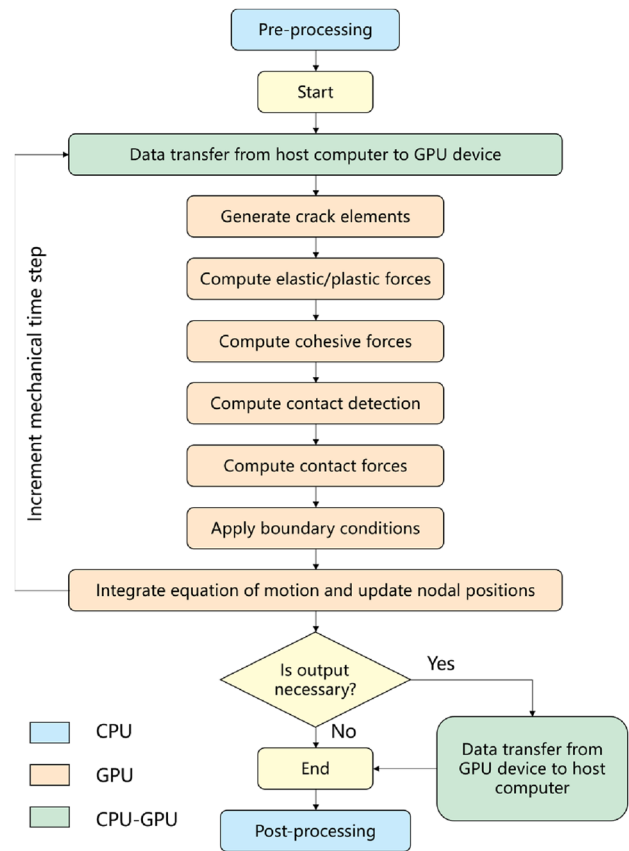


Fig. 14 Flowchart of INFIDEP

conversion not only increases the complexity of program development, but also causes major changes in the original program algorithm and data structure, which increases the difficulty of program maintenance.

In INFIDEP, an improved GPU-parallel framework has been innovated (Fig. 14). After using the CPU to read the input file, the data are directly transferred from the host to the device in text form, and the input data are stored on the device. All functions are computed by GPU. The data are transferred from the device to the host when the results file needs to be written. That is to say, it is no longer necessary to convert between multidimensional arrays and one-dimensional arrays. Compared with the traditional GPU architecture, the waste of memory space is avoided, the complexity of the program is reduced, and the scalability of the program is greatly improved. This method needs to convert the char type data to the initial data type on the device. It should be noted that due to the accuracy error of the computer, there is a certain numerical error when converting char to double, but the comparison shows that this has a very limited impact on the model calculation results. The specific verification will be shown in Sect. 4.3.1.

4.2 Parallelization Implementation

The main theory of FDEM in INFIDEP can be mainly broken down into the following parts:

Generation of zero-thickness cohesive interface elements, calculation node force (for the elastic elements based on limited strain and the elastoplastic element based on the limited strain incremental increase), calculation cohesive force of the joint elements, detection of discrete elements that may come into contact, calculation of the discrete-element contact force, and solving of the governing equations.

For the calculation of node force of elastoplastic elements and crack elements, the algorithm is easy to achieve parallel in GPU based on the characteristics of FDEM discrete element (Liu et al. 2020). Three kernels are constructed in INFIDEP. Each kernel specifies N_B blocks, and the number of threads is specified to N_T , $N_B = \max(N_{element}, N_{node})/N_T + 1$. Kernel 1 is used for data initialization and memory allocation. Kernel 2 is used to calculate the node force of the elastic element (including 3D tetrahedron elements and 2D triangle elements). The Kernel 3 is used for the calculation of node force for the cracking unit (including 3D hexahedral elements and 2D quadrilateral elements), as well as the addition and operation of the joint node force. For the calculation of the elastic-plastic element node force, the basic idea is the same as the above process. It is worth noting that the damage state variable (global variable) needs to be defined during the plastic calculation process.

For the contact detection calculation subroutine, the three stages in Sect. 2.3.1 can be constructed using three kernels respectively. Kernel 1 is used to initialize memory and divide the space domain into cells, and calculate the number and size of cells. Kernel 2 is used to calculate the bounding box information of elements (tetrahedral element in 3D and triangular element in 2D), and place elements in cell and record the feature coordinates of the bounding box. Kernel 3 is used to determine whether elements are adjacent and to mark couples. After testing, it is found that kernel 3 takes the longest time, while kernel 1 and kernel 2 are less computationally intensive. The calculation time of contact detection mainly depends on the calculation speed of kernel 3. Therefore, kernel 1 and kernel 2 are assigned 1 thread for calculation, and kernel 3 is calculated using $x \times y$ blocks, and each block is assigned z threads, that is, one thread processes one cell. It is worth noting that the reading and writing speed of local variables is much higher than that of global memory. Therefore, when calculating, first load the coordinates of the tetrahedron elements in a single thread to local variables, and then judge whether to add contact pairs to the elements in the thread.

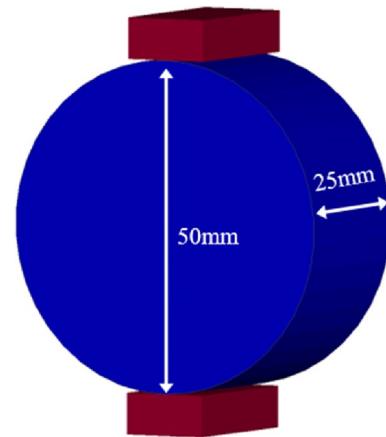


Fig. 15 3D BTS test model

Table 4 Material properties and computational parameters used in INFIDEP for error verification (Munjiza et al. 2011)

Parameter	Unit	Value
Density	kg/m ³	2700
Young's modulus	GPa	50
Poisson's ratio	–	0.25
Tensile strength	MPa	3
Shear strength	MPa	3.15e10
Fracture energy (Mode I and Mode II)	J/m ²	5
Contact friction coefficient	–	0.1
Contact penalty	GPa	40
Cohesive element penalty	GPa	40

For the contact force calculation subroutine, its parallel scheme is similar to that of elastic element nodal force calculation. Kernel 1 is used to store parameters from the graphics card into local memory. Kernel 2 is used to calculate contact potentials and contact forces for actual couples based on contact detection. Each thread is used for contact force calculations for a couple. If the two elements in the couple do not intersect, the thread ends early, saving resources for the next thread calculation.

For the solution of the governing equation, due to the independence of data and calculation process, it is easier to realize parallel calculation by adopting a method similar to that of contact force.

For cohesive elements' generation subroutines, it is only calculated in the first step, not the main time-consuming factor, so it is directly calculated in parallel calculation with a block (including 1 thread). It should be noted that the mesh elements algorithm in INFIDEP contain a lot of logical operations, so the calculation speed on the GPU is lower than the CPU. However, the mesh element algorithm has proven to be optimized to be applied to the

GPU-parallel algorithm (Alhadeff et al. 2015). When the number of elements is large, using a GPU-parallel framework to run the element generation algorithm will have a tremendous advantage.

4.3 Error Verification and Efficiency Testing

The GPU-parallel scheme of INFIDEP adopts an improved parallel framework and a simplified DDBB contact retrieval algorithm, which significantly improves the computational efficiency of FDEM. To validate the correctness of the new algorithm and quantify the acceleration effect of INFIDEP, we simulated the same model with both INFIDEP and traditional FDEM programs, compared the numerical errors

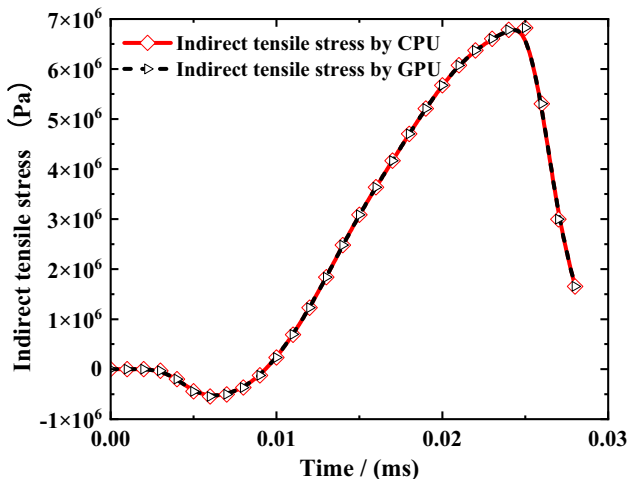


Fig. 16 Indirect tensile stress with time from the simulation of the BTS tests under CPU-FDEM and GPU-INFIDEP

generated by the new algorithm and analyzed the acceleration ratio of the GPU-parallel program. All GPU-parallel computing tests were performed on NVIDIA RTX A6000 GPUs, and all CPU serial computing tests were performed on Intel Xeon Gold 6248R CPUs, with a CUDA core number of 10,752, a core frequency of 1455 MHz, and a video random-access memory capacity of 48 GB.

4.3.1 Error Verification

We utilized a 3D Brazilian tensile strength (BTS) test simulation analysis algorithm to study the error analysis, with a model size shown in Fig. 15. The same example was computed using both the traditional CPU-FDEM program and the GPU-INFIDEP program. Due to the low computational efficiency of the CPU, the number of elements in the simulation model should not be too large. Therefore, the BTS model element characteristic size was set to 3 mm, with 9,381 elastic tetrahedral elements and 19,759 cohesive elements. The material parameters (Table 4) and boundary conditions were identical to those of the "Y" FEM/DEM-provided example (Munjiza et al. 2011); here, we only consider one type of crack, so the shear strength is set to a large value to avoid the occurrence of other types of cracks. The loading plate compressed the specimen with an initial velocity of 2 m/s.

For the CPU and GPU versions of the FDEM program, we simulated the stress–time curves (Fig. 16), the contour plot of stress field (Figs. 17 and 18), and the fracture mode of the specimen, respectively. Our comparison revealed that the obtained stress–time curves produced by both programs were nearly overlapping. Further, the computed stress cloud maps were near identical, with an error of the maximum

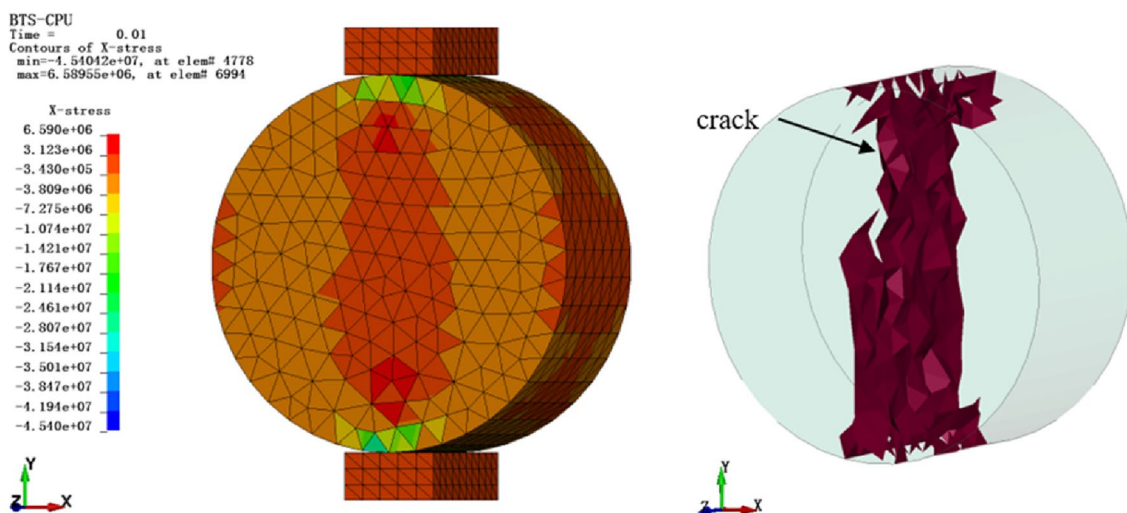


Fig. 17 Contour plot of stress field at 0.01 s of loading and distribution of cracks after failure in the BTS test model, calculated using CPU-FDEM

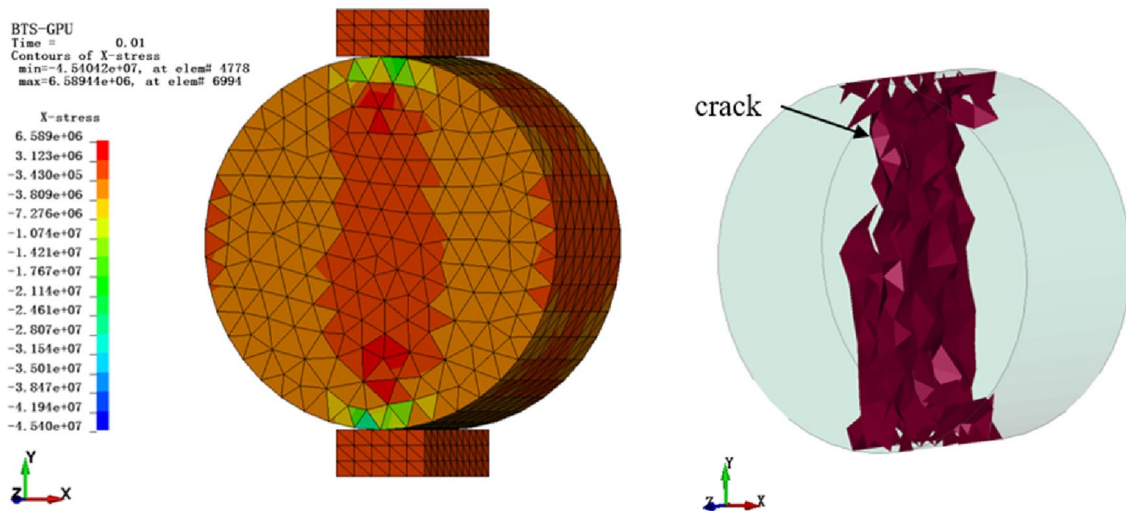


Fig. 18 Contour plot of stress field at 0.01 s of loading and distribution of cracks after failure in the BTS test model, calculated using GPU-INFIDEF

Table 5 Benchmarking tests for 3D INFIDEF

Mesh size/mm	Number of elastic elements	Number of cohesive elements	Steps	CPU elapsed times/s	GPU elapsed times/s	Speedup ratio
7	2,842	6,168	500	125.141	0.863	145
5	9,381	19,759	500	475.349	1.917	248
2	24,103	49,944	500	986.791	2.066	478
1.5	128,251	262,080	500	3642.614	8.035	453
0.84	184,395	376,583	500	4044.677	9.254	437
0.5	619,122	1,259,144	500	36,920.070	86.197	428

stress value of approximately 0.0016%. The resulting cracking direction, crack path, and macroscopic crack shape of the specimen were also consistent, indicating that the new

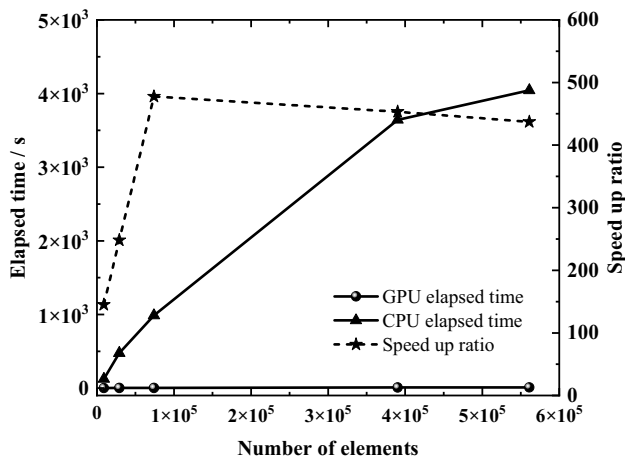


Fig. 19 Elapsed time of parallel and serial calculations in simulated 3D BTS model (500 steps)

parallel framework and contact algorithm within INFIDEF had virtually no impact on simulation outcomes, and the results of the parallel program were reliable. It should be noted that the number of elements in the model is small. Due to the fast-loading speed of the loading plate, the compressive stress of the loading plate affects the central element of the specimen, so the tensile stress in the initial loading stage shows a negative value.

4.3.2 Efficiency Testing

The test of the GPU-parallel acceleration effect of 2D and 3D INFIDEF was completed through BTS experimental simulation. For the calculation of the 3D BTS model, the number of potential contacting couples usually does not change significantly within the first 500 steps, i.e., the material has not yet undergone fracture, and there are relatively few discrete elements that need to undergo contact detection. Therefore, the communication between blocks in the GPU has little effect on the parallel efficiency, and the computational time

Table 6 Benchmarking tests for 2D INFIDEP

Mesh size/mm	Number of elastic elements	Number of cohesive elements	Steps	CPU elapsed times/s	GPU elapsed times/s	Speedup ratio
1	14,110	21,335	10,000	247.019	18.351	13.4
0.7	21,058	31,804	10,000	298.241	21.825	13.6
0.5	41,324	62,289	10,000	568.855	36.765	15.4

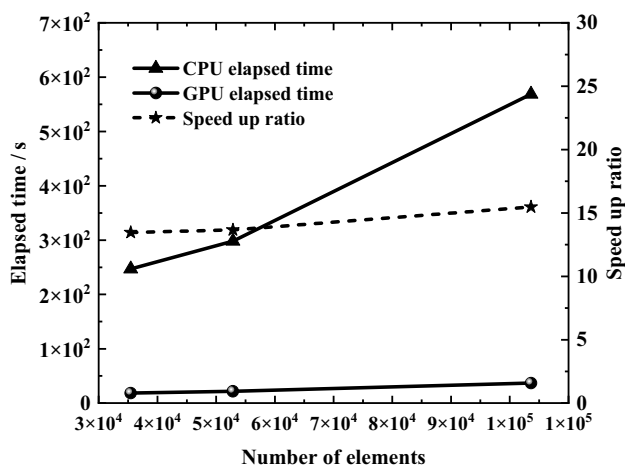


Fig. 20 Elapsed time of parallel and serial calculations in simulated 2D model (10,000 steps)

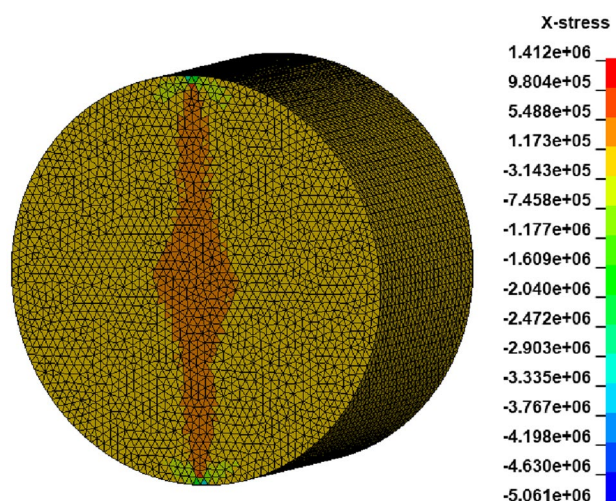


Fig. 21 Distributions of the horizontal stress before the peak stress

Table 7 Material properties and computational parameters used in INFIDEP for BTS models (Fukuda et al. 2020)

Parameter	Unit	Value
Density	kg/m ³	1800
Young’s modulus	GPa	12.2
Poisson’s ratio	–	0.25
Tensile strength	MPa	1.2
Shear strength	MPa	12
Fracture energy (Mode I and Mode II)	J/m ²	2
Contact friction coefficient	–	0.1
Contact penalty	GPa	122
Cohesive element penalty	GPa	12,200
Viscous damping	kg/(m·s)	25,000
Mass scaling factor	–	5

in the first 500 steps is recorded for comparison. We established BTS models with different element sizes (Table 5) and calculated them using CPU-FDEM and GPU-INFIDEP, respectively. The computing time required by the CPU and GPU to compute these BTS models is shown in Fig. 19. All data points in the figure were obtained by numerical tests.

According to the results shown in Fig. 19, with an increasing number of elements in the model, the efficiency

of parallel computing gradually improves. When the number of elastic elements reaches approximately 25,000, the calculation of 500 steps using the CPU serial program takes about 1,000 s, while using the INFIDEP GPU-parallel program only takes 2 s, resulting in a speedup ratio of about 500 times. Overall, when the number of model elements is less than 10,000, the speedup ratio is approximately 200–300. When the number of elastic elements exceeds 20,000, the speedup ratio stabilizes at around 400–500 times. This shows that INFIDEP’s parallel method significantly improves computational efficiency and provides a new technical means for FDEM to be applied to the simulation of large-scale engineering problems.

For 2D INFIDEP, we also conducted GPU acceleration tests on the BTS model for different mesh sizes (Table 6). The obtained speedup results are shown in Fig. 20. It can be seen that when the number of elements exceeds 10,000, the speed of parallel computing using GPU is about 15 times that of using single-core CPU.

To further verify the parallel efficiency of the program, a 3D BTS model with approximately half a million elements was simulated using INFIDEP. The average edge length of the tetrahedral elements was set to 0.94 mm, and the model was discretized with 184,395 tetrahedral elements and 376,583 cohesive elements. The loading rate was 0.05 m/s,

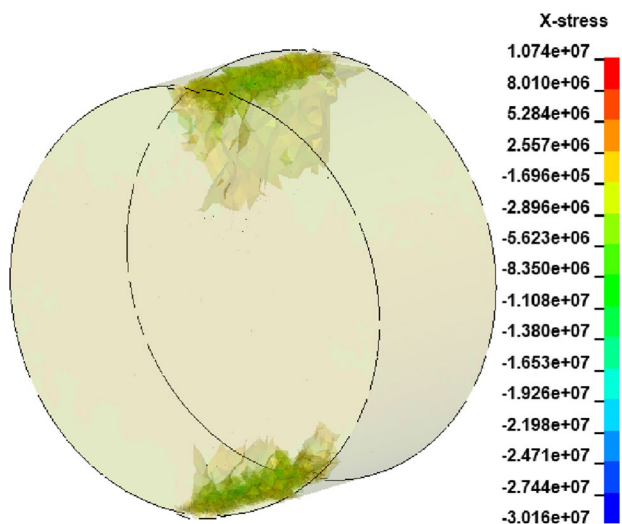


Fig. 22 Distributions of the horizontal stress and cracks at the peak stress

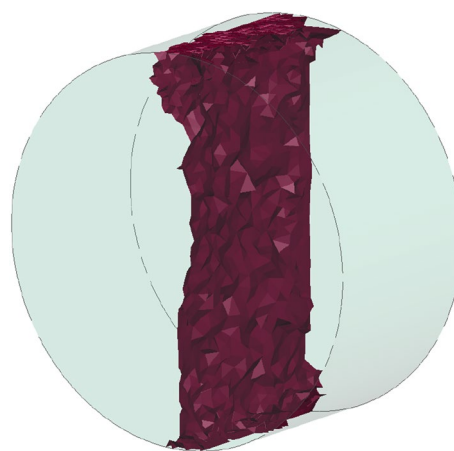


Fig. 24 Simulation of internal fracture mode after failure of the BTS test specimen

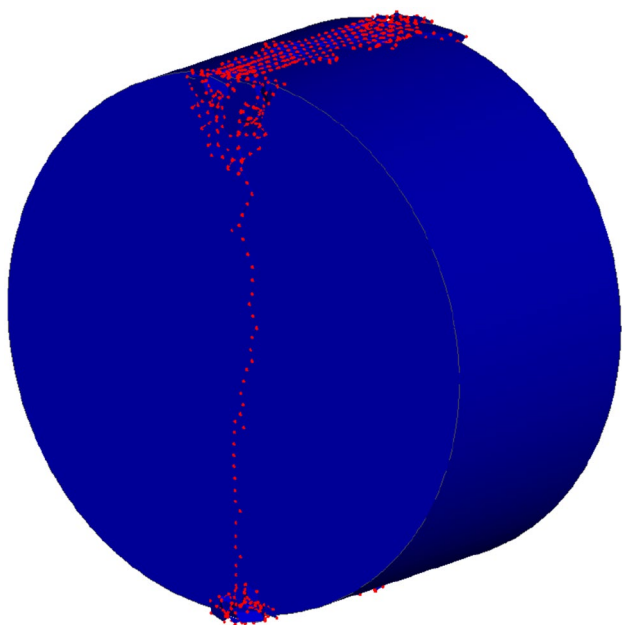


Fig. 23 Simulation of surface cracks after failure of the BTS test specimen

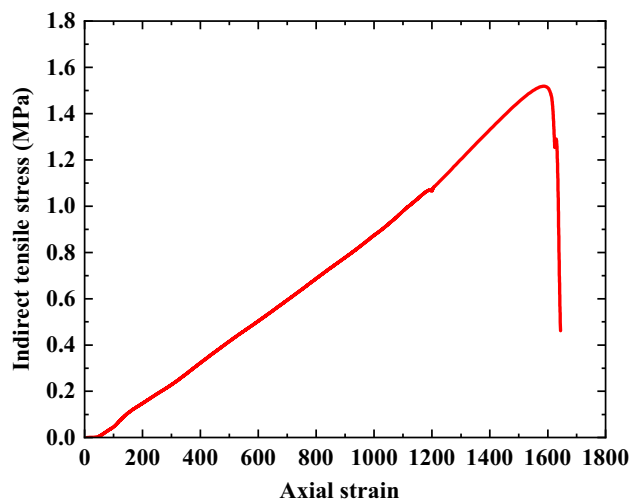


Fig. 25 Brazilian indirect tensile stress versus axial strain curve

and the time step was 9 ns. The material and computational parameters are listed in Table 7 (Fukuda et al. 2020). The critical damping scheme was used, and it should be noted that the penalty parameter values and the mass scaling factor used here are consistent with Fukuda's study.

During the loading process of the specimen, as the loading displacement gradually increases, a uniform horizontal (tensile) stress field is gradually formed along the centerline of the BTS specimen surface (Fig. 21). Due to the brittleness of the

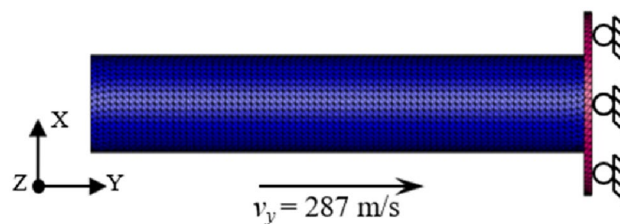


Fig. 26. 3D Taylor bar example setup

rock, cracks first appear in the contact area between the loading plate and the specimen. When the indirect tensile strength of the rock reaches its peak, the cracks coalesce and expand, forming macroscopic cracks along the diameter line of the rock surface (Fig. 22). As the loading plate continues to move, the

resulting macroscopic cracks grow and merge, splitting the rock specimen in half (Fig. 23). The cracks in the specimen do not propagate uniformly along a straight line, mainly due to the use of an unstructured mesh that more closely mimics the actual situation of crack formation. The final form of the macroscopic cracks (Fig. 24) in the specimen is consistent with Hobbs' experimental results (Li and Wong 2013; Hobbs 1965). After the stress exceeds the peak value, it rapidly drops and tends to zero (Fig. 25), which is consistent with the FDEM simulation results of Y-HFDEM 3D IDE (Fukuda et al. 2020). The INFIDEP computation for this model took about 14 h, whereas it would take over 1 year to complete the same 3D BTS model using a sequential FDEM code with the full contact activation approach. Based on the numerical experiments conducted above, we conclude that the proposed contact algorithm performs well in quasi-static tests.

4.4 Full 3D Modeling of the Fracturing Process of Taylor Bar Impact Test

To investigate the effectiveness of INFIDEP in simulating dynamic problems, a 3D Taylor bar impact test was performed. In this test, a cylindrical specimen was launched to impact a rigid target with a prescribed initial velocity (Fig. 26). The bar had a height of 32.4 mm and a radius of 3.2 mm (Ming and Pantalé 2018), and was initially traveling at 287 m/s as it impacted a fixed, rigid wall. The total simulation time for this case was set to 3 ms, and the bar's material properties were consistent with those outlined in Sect. 3.3. The bar was discretized using 83,704 tetrahedral elements and 172,667 cohesive elements, with an average element size of 0.2 mm.

Through INFIDEP, simulations were carried out using an elastic–plastic model, elastic–damage–fracture model, and

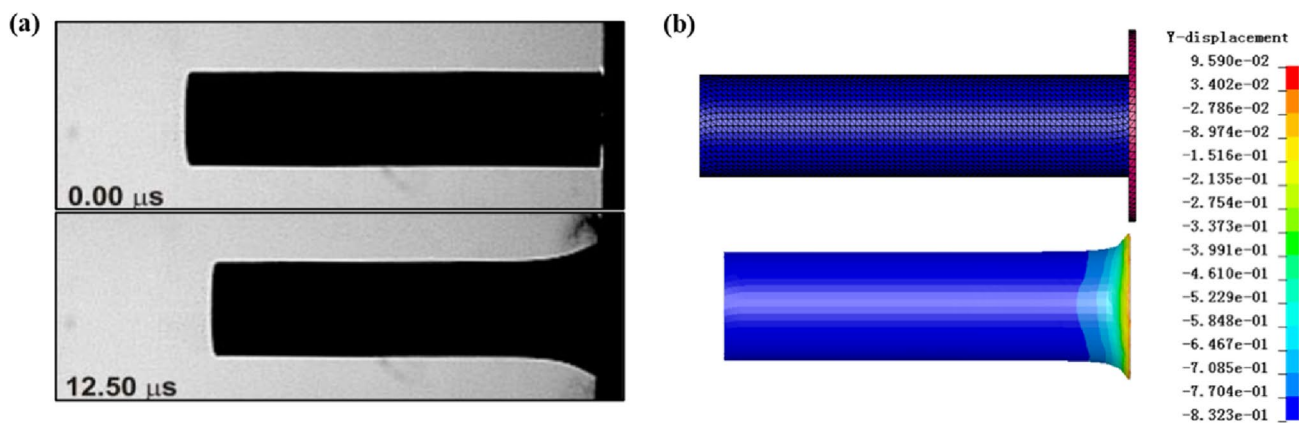


Fig. 27 a High-quality images of Taylor impact tests by Močko et al. (2015). b Taylor impact test based on INFIDEP elastoplastic model simulation

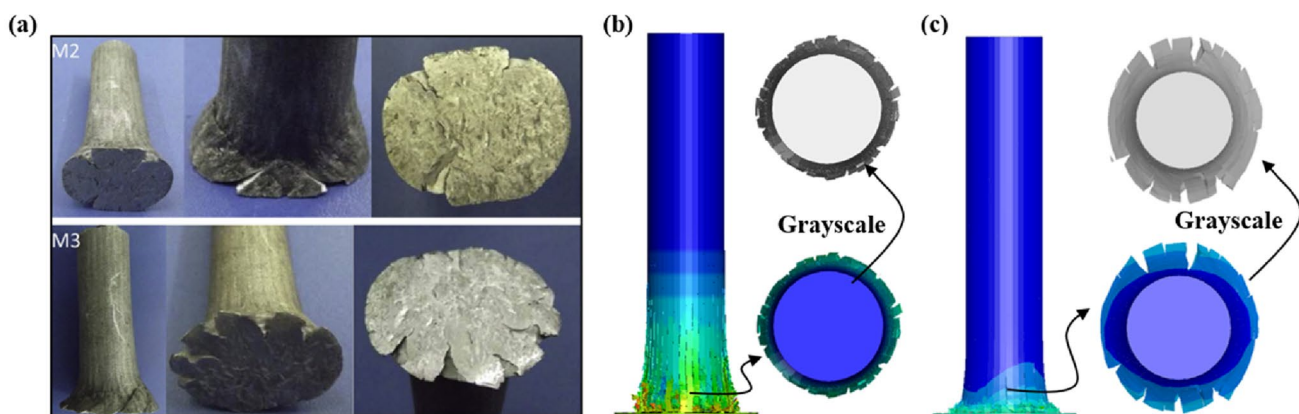


Fig. 28 a Taylor impact tests by Wei et al. (2014). b Taylor impact test based on FDEM elastic–damage–fracture model. c Taylor impact test based on INFIDEP elastoplastic–damage–fracture model

elastoplastic-damage-fracture joint mechanical model, with the calculation results shown in Figs. 27 and 28.

The simulation results using the pure plasticity model reasonably reflected the plastic deformation of the Taylor bar after the impact deformation and were consistent with the results of Moćko et al. (2015)'s study (Fig. 27a). An increase in the specimen diameter near the impacted surface (often described as mushrooming) was observed. The pure J–C plasticity model in INFIDEP can simulate this mushroom-shaped deformation well (Fig. 27b), but it cannot simulate the fracture damage caused by the impact at the end of the specimen (Fig. 28a).

When the simulation was carried out using only the elastic-damage-fracture model, the specimen experienced brittle fracture (Fig. 28b). The entire specimen deformed uniformly along the longitudinal axis, with numerous macroscopic cracks appearing on each cross-section and distributed evenly. The surface cracks were also distributed uniformly, and the diameter of the specimen near the impact face did not increase significantly. This is not in line with the distribution of cracks observed in many Taylor impact tests (Fig. 28a) (Wei et al. 2014), mainly due to the fact that this model cannot describe the material's ductility before breaking, causing the entire cross-section of the specimen to fracture simultaneously.

For the elastoplastic-damage-fracture joint model (Fig. 28c), the introduction of the plastic constitutive model caused non-uniform deformation of the specimen's cross-section, resulting in an irregular distribution of surface cracks. This model not only reasonably simulated the mushroom-shaped plastic deformation of the specimen's head but also simulated the shape of the cracks generated by the impact on the specimen's head surface, consistent with the results of Wei et al.'s study (Fig. 28a) (Wei et al. 2014), which will not be possible if 2D plane-strain modeling is conducted. Simulating 3000 timesteps using GPU-INFIDEP took 308 s, while simulating the model with the full contact activation approach using CPU would take over 30 h. The simulation efficiency is improved by about 260 times. Therefore, the proposed contact detection algorithms are suitable for both quasi-static and dynamic problems.

5 Conclusion

A GPGPU parallel elastoplastic-damage-fracture joint FDEM deformation framework has been developed and implemented into the INFIDEP code. The proposed plastic model assumes large elastic and limited plastic deformation accompanied by highly localized strains represented by discrete cracks for simulating the continuum–discontinuum deformation of 2D/3D materials.

The main novelty of the proposed model is mainly that it combines a three-dimensional discrete form of the damage-fracture model with a strain-increment-based plasticity model within the framework of the combined finite–discrete-element method. The elastoplastic-damage-fracture joint mechanical model was developed and applied to the GPU parallelized FDEM package INFIDEP. This expands the practical algorithm family of the continuous-discontinuous method. The main conclusions obtained are as follows:

- The proposed joint mechanical model consists of two stages: continuous (elastic–plastic) and discontinuous (damage-fracture). The J–C and D–P elastic–plastic constitutive models were successfully constructed for finite elements. Different combinations of the joint mechanical model can describe both brittle and ductile fracture materials, greatly expanding the applications of FDEM. By developing a data type conversion algorithm and an improved GPGPU parallelization framework, the need for repeated memory allocation is avoided, simplifying the parallel structure of the INFIDEP program, improving its stability, computational efficiency, and applicability to engineering-scale problems in FDEM simulations.
- The proposed algorithm was validated for accuracy and computational performance by performing quasi-static BTS experiments using INFIDEP. The simulation results showed that INFIDEP simulated the BTS experimental phenomena well, with stress calculation results having an error of only 0.0016% when compared to the CPU serial simulation results. When simulating BTS models with different element sizes on the NVIDIA A6000 graphics card platform, the speedup ratio stabilized at around 400–500 times when the number of elastic elements exceeded 20,000. The results of the complete BTS model, which contains millions of elements, were consistent with those obtained from the 3D GPGPU HFDEM IDE. This indicates that the GPU parallelization method employed in INFIDEP has good acceleration performance and accuracy, effectively solving the problem of rock fracture in quasi-static loading in the field of rock engineering.
- The accuracy of the proposed plastic constitutive model (based on the FDEM framework) was demonstrated by obtaining stress–strain curves for a 3D cube model, which were consistent with ABAQUS simulation results. The simulation results of the Taylor impact test based on the joint mechanical model for elastoplastic-damage-fracture were able to simulate the deformation trend of the head of the specimen in indoor experiments. It also successfully simulated non-uniform mushroom-shaped crack patterns. Conversely, the traditional FDEM results were not satisfactory, while the 2D plane-strain model was unable to simulate this phenomenon. The use of GPU-parallel acceleration technology increased the cal-

ulation speed of the plastic model by about 400 times compared to CPU serial programs. Overall, INFIDEP provides a powerful computational tool for solving fracture problems of brittle and elastic–plastic materials under quasi-static and dynamic loads in the field of rock engineering.

Acknowledgements This work was financially supported by the Joint Funds of the National Natural Science Foundation of China No. U23A20663, the National Natural Science Foundation of China under Grant No. 52171266.

Funding The work of Bo Han was supported by Joint Funds of the National Natural Science Foundation, under Grant No. U23A20663, and National Natural Science Foundation of China, under Grant Nos. 52171266.

Data availability The authors confirm that the data supporting the findings of this study are available within the article and data will be made available on request.

Declarations

Conflict of Interest 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.

References

- Alhadeff A, Celes W, Paulino GH (2015) Mapping cohesive fracture and fragmentation simulations to graphics processor units. *Int J Numer Meth Eng* 103:859–893
- Belytschko T, Liu WK, Moran B, Elkhodary K (2014) *Nonlinear finite elements for continua and structures*. John Wiley & sons
- Deng PH, Liu QS, Ma H, He F, Liu Q (2020) Time-dependent crack development processes around underground excavations. *Tunn Undergr Space Technol*. <https://doi.org/10.1016/j.tust.2020.103518>
- Drucker DC, Prager WJQ (1952) Soil mechanics and plastic analysis or limit design. *Quart Appl Math* 10:157–165
- Dunne F, Petrinic N (2005) *Introduction to computational plasticity*. OUP Oxford
- Froment M, Rougier E, Larmat C, Lei Z, Euser B, Kedar S, Richardson JE, Kawamura T, Lognonné P (2020) Lagrangian-based simulations of hypervelocity impact experiments on Mars Regolith Proxy. *Geophys Res Lett*. <https://doi.org/10.1029/2020gl087393>
- Fukuda D, Mohammadnejad M, Liu HY, Zhang QB, Zhao J, Dehkoda S, Chan A, Kodama J, Fujii Y (2020) Development of a 3D hybrid finite-discrete element simulator based on GPGPU-parallelized computation for modelling rock fracturing under quasi-static and dynamic loading conditions. *Rock Mech Rock Eng* 53:1079–1112. <https://doi.org/10.1007/s00603-019-01960-z>
- Geomechanica (2019) *Irazu 2D Geomechanical Simulation Software. Theory Manual*
- Guo L (2014) Development of a three-dimensional fracture model for the combined finite-discrete element method. Imperial College London
- Han K, Peric D, Crook A, Owen DJEC (2000a) A combined finite/discrete element simulation of shot peening processes—Part I: studies on 2D interaction laws. *Eng Comput* 17:593–620
- Han K, Peric D, Owen D, Yu JJEC (2000b) A combined finite/discrete element simulation of shot peening processes—Part II: 3D interaction laws. *Eng Comput* 17:680–702
- Han B, Zdravkovic L, Kontoe S (2015) Stability investigation of the Generalised- α time integration method for dynamic coupled consolidation analysis. *Comput Geotech* 64:83–95. <https://doi.org/10.1016/j.compgeo.2014.11.006>
- Hobbs DW (1965) An assessment of a technique for determining the tensile strength of rock. *Br J Appl Phys* 16:259. <https://doi.org/10.1088/0508-3443/16/2/319>
- Johnson GR, Cook WH (1985) Fracture characteristics of three metals subjected to various strains, strain rates, temperatures and pressures. *Eng Fract Mech* 21:31–48
- Karantzoulis N (2017) Development and implementation of inelastic material models for use in femdem numerical methods with applications. In *Applied Modelling and Computational Group Department of Earth Science and Engineering*. Imperial College London, London
- Knight EE, Rougier E, Lei Z, Euser B, Chau V, Boyce SH, Gao K, Okubo K, Froment M (2020) HOSS: an implementation of the combined finite-discrete element method. *Computational Particle Mechanics* 7:765–787. <https://doi.org/10.1007/s40571-020-00349-y>
- Lei Z, Rougier E, Knight EE, Munjiza A (2014) A framework for grand scale parallelization of the combined finite discrete element method in 2d. *Comput Part Mech* 1:307–319. <https://doi.org/10.1007/s40571-014-0026-3>
- Lei Z, Rougier E, Knight EE, Frash L, Carey JW, Viswanathan H (2016) A non-locking composite tetrahedron element for the combined finite discrete element method. *Eng Comput*. <https://doi.org/10.1108/EC-09-2015-0268>
- Lei Z, Bradley CR, Munjiza A, Rougier E, Euser BJCPM (2020a) A novel framework for elastoplastic behaviour of anisotropic solids. *Comp Part Mech* 7:823–838
- Lei Z, Rougier E, Euser B, Munjiza A (2020b) A smooth contact algorithm for the combined finite discrete element method. *Comput Part Mech* 7:807–821
- Lei Z, Zang M, Munjiza A (2010) Implementation of combined single and smeared crack model in 3D combined finite-discrete element analysis. *Discrete element methods—simulation of Discontinua: theory and applications*, London, UK, 102–107
- Li D, Wong LNY (2013) The Brazilian disc test for rock mechanics applications: review and new insights. *Rock Mech Rock Eng* 46:269–287
- Lisjak A, Mahabadi OK, He L, Tatone BSA, Kaifosh P, Haque SA, Grasselli G (2018) Acceleration of a 2D/3D finite-discrete element code for geomechanical simulations using General Purpose GPU computing. *Comput Geotech* 100:84–96. <https://doi.org/10.1016/j.compgeo.2018.04.011>
- Liu QS, Deng PH (2019) A numerical investigation of element size and loading/unloading rate for intact rock in laboratory-scale and field-scale based on the combined finite-discrete element method. *Eng Fract Mech* 211:442–462. <https://doi.org/10.1016/j.engfracmech.2019.02.007>
- Liu Q, Wang W, Ma H (2020) Parallelized combined finite-discrete element (FDEM) procedure using multi-GPU with CUDA. *Int J Numer Anal Meth Geomech* 44:208–238
- Liu H, Liu QS, Ma H, Fish J (2021) A novel GPGPU-parallelized contact detection algorithm for combined finite-discrete element method. *Int J Rock Mech Min Sci*. <https://doi.org/10.1016/j.ijrmm.2021.104782>
- Liu H, Ma H, Liu Q, Tang X, Fish J (2022) An efficient and robust GPGPU-parallelized contact algorithm for the combined finite-discrete element method. *Comput Methods Appl Mech Eng*. <https://doi.org/10.1016/j.cma.2022.114981>
- Lubliner J (2008) *Plasticity theory*. Courier Corporation

- Lukas T, D'Albano GS, Munjiza A (2014) Space decomposition based parallelization solutions for the combined finite–discrete element method in 2D. *J Rock Mech Geotech Eng* 6:607–615
- Mahabadi OK, Lisjak A, Munjiza A, Grasselli G (2012) Y-Geo: new combined finite–discrete element numerical code for geomechanical applications. *Int J Geomech* 12:676–688. [https://doi.org/10.1061/\(asce\)gm.1943-5622.0000216](https://doi.org/10.1061/(asce)gm.1943-5622.0000216)
- Ming L, Pantalé O (2018) An efficient and robust VUMAT implementation of elastoplastic constitutive laws in Abaqus/Explicit finite element code. *Mech Ind.* <https://doi.org/10.1051/meca/2018021>
- Močko W, Janiszewski J, Radziejewska J, Grązka M (2015) Analysis of deformation history and damage initiation for 6082–T6 aluminium alloy loaded at classic and symmetric Taylor impact test conditions. *Int J Impact Eng* 75:203–213
- Munjiza AA (2004) The combined finite–discrete element method. John Wiley & Sons
- Munjiza A, Andrews K, White J (1999) Combined single and smeared crack model in combined finite–discrete element analysis. *Int J Numer Meth Eng* 44:41–57
- Munjiza AA, Knight EE, Rougier E (2011) Computational mechanics of discontinua. John Wiley & Sons
- Munjiza A, Rougier E, Knight EE (2015) Large strain finite element method: a practical course. John Wiley & Sons
- Munjiza A, Rougier E, Lei Z, Knight EE (2020) FSIS: a novel fluid–solid interaction solver for fracturing and fragmenting solids. *Comput Part Mech* 7:789–805. <https://doi.org/10.1007/s40571-020-00314-9>
- Munjiza A (1992) Discrete elements in transient dynamics of fractured media. Swansea University
- Rougier E, Munjiza A, Lei Z, Chau VT, Knight EE, Hunter A, Srinivasan G (2020) The combined plastic and discrete fracture deformation framework for finite–discrete element methods. *Int J Numer Meth Eng* 121:1020–1035. <https://doi.org/10.1002/nme.6255>
- Schiava D'Albano GG (2014) Computational and algorithmic solutions for large scale combined finite–discrete elements simulations. University of London, Queen Mary
- Simo JC, Hughes TJ (2006) Computational inelasticity. Springer Science & Business Media
- Smith M (2009) ABAQUS/standard user's manual. Version 6:9
- Tatone BS, Grasselli G (2015) A calibration procedure for two-dimensional laboratory-scale hybrid finite–discrete element simulations. *Int J Rock Mech Min Sci* 75:56–72
- Wei G, Zhang W, Huang W, Ye N, Gao Y, Ni Y (2014) Effect of strength and ductility on deformation and fracture of three kinds of aluminum alloys during Taylor tests. *Int J Impact Eng* 73:75–90
- Wilkins ML (1963) Calculation of elastic–plastic flow. California Univ Livermore Radiation Lab
- Wilson CD. (2003) Development of a pressure-dependent constitutive model with combined multilinear kinematic and isotropic hardening. In 2004 International ABAQUS Users'Conference
- Wu D, Li H, Fukuda D, Liu HJC, Geotechnics, (2023) Development of a finite–discrete element method with finite–strain elasto–plasticity and cohesive zone models for simulating the dynamic fracture of rocks. *Comput Geotech* 156:105271
- Xiang J, Latham J-P, Vire A, Anastasaki E, Pain CCJCEP (2012) Coupled fluidity/y3d technology and simulation tools for numerical breakwater modelling. *Coast Eng Proceed* 1:57–66
- Yan C, Wei D, Wang & Engineering, (2022a) Three-dimensional finite discrete element-based contact heat transfer model considering thermal cracking in continuous–discontinuous media. *Comput Methods Appl Mech Eng* 388:114228
- Yan C, Xie X, Ren Y, Ke W, Wang M (2022b) A FDEM-based 2D coupled thermal–hydro–mechanical model for multiphysical simulation of rock fracturing. *Int J Rock Mech Min Sci* 149:104964
- Yan C, Zhao Z, Yang Y, Zheng HJC (2023) A three-dimensional thermal–hydro–mechanical coupling model for simulation of fracturing driven by multiphysics. *Comput Geotech* 155:105162
- Yang P, Xiang J, Fang F, Pain CC (2019) A fidelity fluid–structure interaction model for vertical axis tidal turbines in turbulence flows. *Appl Energy* 236:465–477. <https://doi.org/10.1016/j.apenergy.2018.11.070>
- Zeller C (2011) Cuda c/c++ basics. NVIDIA Coporation
- Zhao LH, Liu XN, Mao J, Xu D, Munjiza A, Avital E (2018) A novel contact algorithm based on a distance potential function for the 3D discrete–element method. *Rock Mech Rock Eng* 51:3737–3769. <https://doi.org/10.1007/s00603-018-1556-4>
- Zheng Z, Zang M, Chen S, Zhao C (2017) An improved 3D DEM–FEM contact detection algorithm for the interaction simulations between particles and structures. *Powder Technol* 305:308–322

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.