SHIFT: An implementation for lattice Boltzmann simulation in low-porosity porous media

Jingsheng Ma,[*,†] Kejian Wu,[*] Zeyun Jiang,[*] and Gary D. Couples[*]
Institute of Petroleum Engineering, Heriot-Watt University, Edinburgh EH14 4AS, United Kingdom
(Received 11 February 2010; published 6 May 2010)

The lattice Boltzmann (LB) method has proven to be a promising method for simulating fluid dynamics in porous media. When fluid flow in pores is the only concern, a standard LB implementation, which stores one or two sets of particle distribution functions (PDFs) for both pore and solid cells, wastes a large amount of memory, especially for low-porosity media. This paper proposes a LB implementation scheme that stores a single set of PDFs for pore cells only and therefore makes it possible to simulate flow through larger and more-realistic porous models. A unique feature of this scheme is that it decomposes all PDFs into a set of 1D arrays in such a way that each array corresponds to a set of pore cells that connect one another along a pair of opposite LB velocity directions. This allows LB propagation and a standard bounce-back rule to be realized together as one or two circular shifting operations on every array. For this reason, this scheme is called SHIFT. Although PDFs are not stored in an efficient way for LB collision operation, it is shown that the incurred overhead could be reduced by properly arranging PDF arrays according to the pore structures. A D3Q15 LB implementation of SHIFT using the lattice Bhatnagar-Gross-Krook model is applied to simulate the Stokes flow through models of four natural and synthetic rock samples with porosities ranging from about 10% to 38%. Results show that SHIFT requires 36–82 % less memory than a comparable D3Q15 LB does, which stores a single set of PDF for both pore and solid cells. SHIFT achieves minimum performances of over 11 and 3.8 mega-lattice-updates-per-second (MLUPS) for the combined propagation and bounce-back operation and the collision operation, respectively, and therefore a minimum of 2.8 MLUPS in total on a computer with one AMD Opteron 2218. The performance of the collision operation is significantly improved for all cases when a simple *K*-mean clustering technique is employed to rearrange PDF arrays. It is argued and shown that the number of PDF arrays per pore cell and the length frequency of PDF arrays are useful measurements on the geometry and topology of the pore structures and these characteristics are able to explain SHIFT performance variations.

DOI: 10.1103/PhysRevE.81.056702 PACS number(s): 02.70.−c

I. INTRODUCTION

Recent advances in computer-aided imaging have made it possible to obtain high-resolution three-dimensional (3D) voxel models of the microstructure of natural porous materials. Those models can be simplified further to yield appropriate surrogates (e.g., binary pore or solid models, pore network models, grain models, etc.) on which various computational methods may be applied to simulate a variety of physical behaviors, concerning fluid flow and transport, thermal conduction, mechanics and/or electric conductance, and to predict relevant macroscale properties of modeled porous materials. One particular method is the lattice Boltzmann (LB) method (see Refs. [1–6]) for simulating fluid flow through geometrically complex porous models where solids and pores are resolved explicitly, leading to the prediction of single-phase and multiphase fluid flow and fluid transport properties (see Refs. [7–13]). The LB method represents fluid flow by particle distribution functions (PDFs) on regular or irregular grids. The evolution of PDFs is characterized by propagation and collision operations, as well as methods for handling the flow boundary conditions at the interfaces of pore and solid cells. In this paper, the grids considered are regular and uniform.

Standard LB implementations require PDFs for both pore and solid cells to be stored. However, for simulating fluid dynamics problems where the pore geometries are static, only pore cells, and the interfaces of pore and solid cells are of concern. Therefore, the standard implementations incur a memory waste because cells which are not relevant still incur memory costs. For natural low porosity porous media such as rocks (porosity $<35\%$ for typical reservoir sandstones), any model is required to be large enough so that the average flow is able to account for the intrinsic heterogeneity of the pores, whose sizes and shapes are nonrepeating [14,15], while the grid space of that model must be small enough to resolve the pores properly in order to obtain accurate and reliable results [12,13,16,17]. Thus, the unnecessary memory incurred by such a model can cause the total memory size to far exceed the memory capacity of computer systems even for single-phase fluid flow simulation. As a result, a standard LB implementation is limited to relatively small models. The need to reduce the memory requirement to enable fluid flow simulation on large high-resolution models calls for computational schemes that store PDFs for pore cells only.

Recently several LB schemes have been proposed for sequential and/or parallel computer systems [18–22]. All of these schemes share a common feature. That is, each devises a data structure to encapsulate cell adjacency information among pore cells and with boundary solid cells. This

[*]Also at ECOSSE (Edinburgh Collaborative of Subsurface Science and Engineering), a part of the Edinburgh Research Partnership in Engineering and Mathematics.
[†]Corresponding author; jingsheng.ma@pet.hw.ac.uk

information is required, but no longer determinable, from the relative locations of cells as in a full model. Since the data structures provide only semi-indirect or indirect access to PDFs, they lead to "irregular" memory access patterns that make it more difficult to implement the LB operations efficiently.

In this paper, a computational scheme for LB simulation is proposed in low-porosity porous media as a further attempt at finding efficient schemes. This scheme requires a single set of PDFs to be stored only for pore cells. Unlike other schemes, this scheme explicitly explores the internal structures of connected pore cells in respect of the chosen lattice structure in the LB method. It decomposes PDFs into a set of one-dimensional (1D) arrays, with each PDF array corresponding to one set of connected pore cells along a pair of opposite velocity directions. This leads to a propagation-optimized layout of PDFs as opposed to collision-optimized layouts (see discussions on these terms in a later section). This scheme reduces the computational costs of the propagation operation and the handling of the standard bounce-back (i.e., simply reversing the momentum of the particle that meets a solid cell) to simple one or two circular shifting operations on every array, and it thus maximizes the performance of the combined propagation and bounce-back operation. This scheme is called SHIFT to reflect the nature of shifting operations. For the collision operation this scheme utilizes a simple data indexing scheme to map each pore cell to all PDFs associated with that cell, one for each velocity direction. Although this introduces some memory and computation overhead due to the nature of indirect addressing, the overhead can be reduced effectively by rearranging the PDF arrays in memory according to the pore structures of a model. By exploring the pore structure directly, SHIFT results in two by-products that relate to the geometry, connectivity, and tortuosity of the pores: (1) the number of the PDF arrays per pore cell and (2) the adjusted length frequency of PDF arrays. These measures are shown to be able to differentiate models that have similar porosities, and they explain SHIFT performance variations.

The remainder of this paper is organized as follows. Section II introduces the basics of LB. Section III describes the SHIFT scheme in detail. Section IV presents numerical experiments to explore computational aspects of SHIFT. Merits of and future improvements to SHIFT are discussed in Sec. V followed by concluding remarks.

II. LATTICE BOLTZMANN METHOD AND IMPLEMENTATIONS

A. LB method

A lattice Bhatnagar-Gross-Krook (LBGK) model [23,24] is briefly introduced here for the purpose of describing the SHIFT scheme in the following section. For a fuller discussion of the LB method, especially for porous media, the reader is referred to the following recent publications (see Ref. [12] and references therein).

The BGK LB model is defined by the following equation:

$$f_j(x+e_j\Delta t,t+\Delta t)-f_j(x,t)=\frac{1}{\tau}[f_j^{(eq)}(x,t)-f_j(x,t)],$$

$$j=0,1,\ldots,b-1, \tag{1}$$

where $f_j(x,t)$ is a PDF representing the probability of finding a fluid particle with a velocity $\boldsymbol{e}_j$ at location $\boldsymbol{x}$ and time t. τ is the relaxation time. b is the number of velocities. $f_j^{(\mathrm{eq})}(x,t)$ is an equilibrium distribution function, which can be chosen so that fluid obeys the isothermal Navier-Stokes equation in the incompressible limit [25].

Velocity $\boldsymbol{e}_j$ is determined by a lattice structure. For 3D regular cell models, three of the most commonly used lattice structures are D3Q15, D3Q19, and D3Q27 according to the DdQb notation of Qin *et al.* [24]. Here, d is the space dimensions, while b is the number of velocities including the zero velocity, $\boldsymbol{e}_0$. He and Luo [4,25] give two general forms of $f_j^{(\mathrm{eq})}(x,t)$ leading to the recoveries of compressible and incompressible Navier-Stokes equations, respectively, for these lattice structures and others.

From Eq. (1), the fluid particle evolution involves two distinct steps:

$$\text{collision:}\quad \tilde{f}_j(x,t)=f_j(x,t)-\frac{1}{\tau}[f_j(x,t)-f_j^{(eq)}(x,t)],$$

$$j=0,1,\ldots,b-1, \tag{2}$$

$$\text{propagation:}\quad f_j(x+e_j\Delta t,t+\Delta t)=\tilde{f}_j(x,t),$$

$$j=0,1,\ldots,b-1. \tag{3}$$

A third step, which may be combined with the previous ones, is also required to deal with flow conditions at the interfaces between pore and solid cells.

B. Implementation

In LB implementations, PDFs are typically expressed as a multidimensional array or a set of 1D arrays, depending on the memory management techniques and programming languages used. In order to maximize the LB performance, the layouts of array elements need to be chosen carefully because the collision and propagation steps operate in different fashions. The collision step operates cell by cell. At each time step t, at cell x_i, it loads the contents of all PDFs of that cell, one for each velocity direction, that is, $f_j(x_i,t), j=0,1,\ldots,b-1$; then it calculates postcollision PDFs, $\tilde{f}_j(x_i,t)$. On the other hand, the propagation step operates direction by direction. At each time step t, for each velocity $\boldsymbol{e}_j$, it streams the new PDF of every cell to its next neighboring cell along that velocity direction. In the case that cells and velocity directions are indexed by i and j, respectively, PDFs could be stored in a matrix. Assuming the matrix is in the column first order, the collision and propagation operations would therefore prefer forms of $\{\mathrm{pdf}(i,j)\}$ and $\{\mathrm{pdf}(j,i)\}$, respectively. This is because modern computer

systems permit more efficient access (read/write) to data close to one another than to those far away from one another in memory space. Since only one layout may be implemented in a code, either the propagation or the collision operation will inevitably take longer to access PDFs, limiting the LB performance. Wellein *et al.* [26] compared a variety of layout schemes for a D3Q19 LB model on several computer systems, and they found that the propagation-optimized layouts offer overall better performance than their counterparts do.

In a typical LB implementation, two sets of PDFs are stored, predominantly for simplifying propagation operations as explained below. At the propagation step, the next neighboring cell is determined as follows. Given a n_1 by n_2 by n_3 grid, a cell x at (i_1, i_2, i_3), where i_1, i_2, i_3 are the cardinal indices of the cell in the grid, and a velocity direction $v=(v_1,v_2,v_3)$ of $\{e_j: j=0,1,\ldots,b-1\}$, the next cell y, starting from x along v, will be at $(\langle i_1,v_1\rangle,\langle i_2,v_2\rangle,\langle i_3,v_3\rangle)$, where $\langle i_j,v_j\rangle=1$ if $i_j+v_j>n_j$, n_j if $i_j+v_j<1$, or i_j+v_j if otherwise, and v_j takes -1, 0 or 1 for $j=1,2,3$. Clearly, a propagation along a velocity direction is not consistent with the cell indexing, which typically follows cardinal directions of a model as shown here or other directions (e.g., when space-filling curves are used [27]). Because a PDF at a cell can be streamed only if the PDF at the next cell has already been streamed or backed up, a second set of PDFs is used to simplify the propagation operation.

Some LB schemes have been proposed to store PDFs for pore and interface cells only, in schemes with either two sets of PDFs (see [19–21]) or one set of them (see [18,22]). Because of the choice to exclude solid cells, the cardinal indexing is no longer available for these schemes. Hence, adjacency information among pore cells and with the boundary solid cells has to be provided by semidirect or indirect indexing schemes for both propagation and collision operations. Pan *et al.* [20] and Wang *et al.* [21] proposed two sets of LB implementation schemes for parallel processing on a cluster of computers, and they compare their performances for different indexing and domain decomposition schemes, respectively. Mattila *et al.* [22] extended a compressed-grid scheme of Pohl *et al.* [28] by introducing a bundled data layout of a vector compressed grid and fusing the propagation and collision operations at the cell level and called this extended scheme "swap." Mattila *et al.* [29] compare a number of LB schemes that employ different approaches in the following three areas of LB implementation: (1) PDF storage and layouts (i.e., solid+pore/pore cells and propagation-optimized/collision-optimized/bundled layout); (2) pore cell indexing schemes (i.e., direct, semi-direct, indirect); and (3) the ways of handling propagation and collision as a two-pass or one-pass LB iteration. The authors of [29] show that the LB schemes based on the semidirect swap algorithm in [22] achieve the best performance and lowest memory consumption among the compared schemes for simple porous models. The key difference of that scheme from others is that it exploits the pore cell relationships in all three areas above.

C. Flow boundary conditions

Handling flow conditions at the interfaces between pore and solid cells is an important aspect of LB simulation. A no-slip boundary condition has been widely used in simulating fluid flow at the pore scale. A number of schemes have been developed to approximate the no-slip boundary condition, including so-called bounce-back rules (see [30] and references therein), higher-order interpolation schemes [31–34], and multireflection schemes [35,36]. The accuracy of these schemes may depend on the lattice resolutions, the locations of the solid walls between pore and solid cells and their shapes, the selection of the relaxation parameter(s), the magnitude of the fluid flow driving force and the types of the LB models—single relaxation-time (SRT) or multirelaxation-time (MRT) LB models [37,38]. Of many bounce-back schemes, the most widely used one is the so-called standard bounce-back (SBB) rule that simply reverses the momentum of the particle at the solid walls. SBB is very simple to implement, and when this is done appropriately, incurs little extra computation. The main drawback of this SBB rule is that it can produce a nonzero slip velocity, dependent on the lattice resolution and the selection of the relaxation factor τ and thus fluid viscosity [30].

For porous models with more complicated pore geometry, the accuracy of LBGK (a SRT model with SBB) is known to have more complex relationships with those factors above than for a simple model containing a single channel. This leads to difficulties in the selection of appropriate parameters in LB simulations for realistic media. For example, it has been suggested that a smaller τ than the "ideal" one for a channel model may be used for natural porous media with tight pores [16,17,39]. Using a few well-resolved simple synthetic models of packed spherical beads, Pan *et al.* [12] showed that this dependence can be reduced using MRT LB models [37] in conjunction with the higher-order interpolation and multireflection schemes mentioned above. However, little is known about them for realistic models of natural rocks. Chun and Ladd [34] proposed an interpolation scheme in conjunction with a MTR LB [38] and compared it with other schemes [30,31,35]. That scheme results in a velocity field that is second-order accurate in space and independent of fluid viscosity. As opposed to the other interpolation and the multireflection schemes [12,35], it has two advantages; (1) it only requires the information on every pair of boundary pore and solid cells along each velocity direction; and (2) it works on pores with a minimum spacing of one cell. Despite all the disadvantages of SBB, it is employed in the initial implementation of SHIFT. The use of other boundary schemes in SHIFT will be considered separately.

III. SHIFT

Unlike other LB schemes mentioned previously, SHIFT exploits the characteristics of the pore structures explicitly and arranges PDFs along every pair of opposite velocity directions of a LB lattice structure. More precisely, it divides PDFs into a set of one-dimensional arrays, and each array corresponds to one set of connected pore cells along a pair of two opposite velocity directions. The key advantages of this are the following: (1) there is no need to express the adjacency of pore cells for the propagation and (2) the propagation can be done on each of the PDF arrays in parallel with-

out introducing a second set of PDFs. In the discussion to follow, it becomes clear that, when the PDFs in every array are arranged in a certain order, the propagation and SBB can be performed as one or two circular shifting operations on the elements of every array.

However, when the PDFs are decomposed in this way, the natural cell indexing to the PDFs for every cell, required by the collision operation, no longer exists. SHIFT addresses this problem by providing an indexing scheme that maps every cell to the locations of all PDFs of that cell in corresponding PDF arrays. This requires additional memory to store adjacency information. There is a trade-off (see below for more explanations) between what information is to be stored and how fast adjacent relations can be determined (see Mattila *et al.* [29], for a review). Although PDFs are not stored in an efficient way for the LB collision operation, it is demonstrated later that a good performance can still be achieved if the PDF arrays can be arranged accounting for the geometry and topology of the pore structures.

A. Decomposition of PDFs

To decompose PDFs in this scheme, one first needs to identify every set of connected pore cells along each pair of opposite velocity directions. For any volume of a porous model that can be subdivided into a set of smaller nonoverlapping cubic volumes of the same size, this can be done by repeating a simple scan-line procedure as follows: from a starting cell, and along a positive velocity direction, traverse cell by cell in the same way as LB propagates a PDF from one cell to its next neighboring cell. It is easy to see that each scan will traverse back to the starting cell, and therefore, all cells being visited form a circular chain. For a circular chain, every set of connected pore cells, i.e., a segment of that chain containing only pore cells, can be identified readily. Note that each set may be bounded by either one or two solid cells or none. If it is bounded, one can define the two end cells of each set as the head and tail cells so that traversing from the tail to head cells follows the positive velocity direction. In the other case, a chain contains only pore cells and one can choose any cell to be the tail and the next cell, along the positive direction, to be the head.

The cell scan-line procedure above can be repeated for every pore cell on any three orthogonal boundary faces of a volume. First, for every cell on one of the faces, it scans from that cell along every positive velocity direction that is not perpendicular to any of the other two faces. Second, for every cell on each of the other two faces in turn, it scans from that cell along the positive direction that is perpendicular to that face. This procedure finds all possible circular cell chains of a volume for any given lattice structure and therefore every set of connected pore cells of that volume.

Once all sets of the connected pore cells are identified, PDFs can then be decomposed. For each set of the connected pore cells, one can arrange all PDFs associated with every cell in that set, and in both positive and negative directions, as a symmetric array as follows: (1) arranging all PDFs along the positive direction one by one in the cell order from head to the tail and (2) appending PDFs along the negative direction in the reverse cell order. Repeating this procedure over every set completes the decomposition.

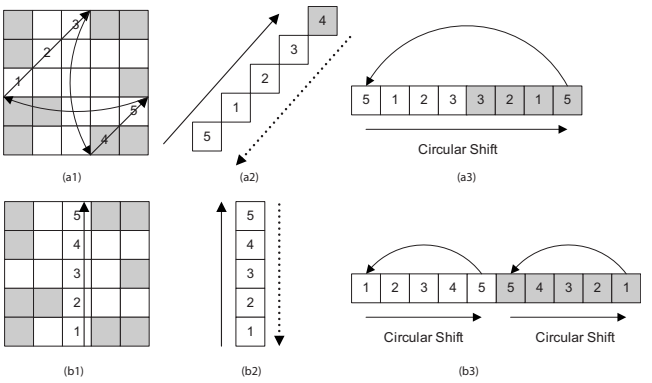

FIG. 1. Illustration of PDF decomposition in SHIFT. In (a1), (a2), and (b1), shaded cells represent solids. In (a3) and (b3), shaded cells represent the array elements corresponding to the opposite velocity direction with respect to unshaded counterparts, respectively.

Figure 1 illustrates the decomposition of PDFs in SHIFT for a simple two-dimensional (2D) pore model without losing any generality. In Fig. 1(a1), pore cells 5, 1, 2 and 3 are connected one another according to the PDF decomposition procedure (scan-line starts from cell 1). Cells 3 and 5 are the tail and the head, respectively, where the positive direction is indicated by the thicker arrow lines. The thin curves indicate that the scan-line reaches one end of the boundary and returns to another side according to the scanning rule. These pore cells bounded by one solid cell can be rearranged logically as in Fig. 1(a2). Figure 1(a3) shows a PDF array for that set. It is easy to see that only one single circular shifting on that PDF array along the positive direction is needed to complete the propagation and SBB operations. Note that this is true for a set bounded by two solid cells. Figures 1(b1) and 1(b2) show a whole chain that contains only pore cells and forms a circular set where cells 1 and 5 are chosen to be the head and the tail, respectively. For a circular set of the connected pore cells, two circular shift operations are needed on the first and second halves of the PDF array, respectively [see Fig. 1(b3)], to complete the propagation. Note that for each cell, its two PDFs, one for each of two opposite directions, are located symmetrically in a PDF array with respect to the center.

B. Mapping cells to PDFs and SHIFT data structure

The procedure for decomposing PDFs yields the following information: (1) the number of the PDF arrays; (2) the number of PDFs in each array; (3) the type of every array in terms of the number of shifting operations required; (4) the locations of the two PDFs in a corresponding PDF array for every cell along each pair of opposite velocity directions; and (5) the original cardinal indices of every pore cell. Note that the PDF arrays can be arranged freely in any logical order. In the following discussion, the PDF arrangement resulted from the decomposition process described above is referred to as the default arrangement.

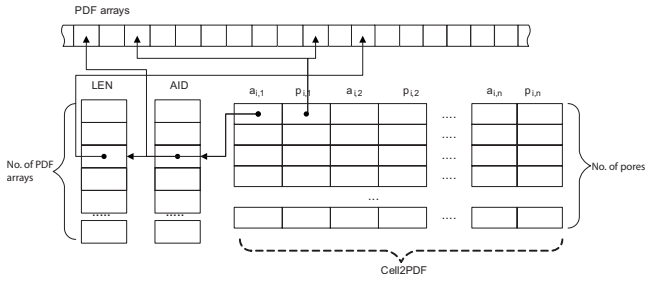

FIG. 2. A schematic diagram of key data components in SHIFT and their relationships.

To facilitate the collision operation, a mapping table can be constructed using the information above to associate each pore cell with all its PDFs in corresponding PDF arrays. This table contains N_p rows and $(N_v-1)/2$ columns, where N_p is the number of pore cells and N_v is the number of velocity directions of a lattice structure, which is equal to b. Each element contains a pair of integers, $(a_{i,j},\ p_{i,j})$, where $a_{i,j}$ is an index to the beginning of a PDF array, for cell i along the jth positive direction. $p_{i,j}$ defines the positions, relative to either end of that PDF array, at which the two PDFs for that cell are located. This table is named as Cell2PDF. Note that because of the way PDFs are decomposed, no two a_{i,j_1} and a_{i,j_2} can be the same if $j_1 \neq j_2$. But two $a_{i_1,j}$ and $a_{i_2,j}$ can be the same even if $i_1 \neq i_2$.

Given $(a_{i,j},\ p_{i,j})$, the length of the referenced PDF array is needed to determine the exact positions of two PDFs in that array. For this purpose, LEN, an integer array, is introduced, to store the lengths for every PDF array. If one uses another array, denoted as AID, to store the beginning addresses for every PDF array in the same order as in LEN, $a_{i,j}$ is reduced to an index to a pair of LEN and AID. Figure 2 shows a schematic diagram of the SHIFT key data components, including the PDF arrays, LEN, AID, and Cell2PDF, and the relations of these components as indicated by arrowed links. Note that the PDF arrays do not have to include the PDFs corresponding to the zero direction, $\boldsymbol{e}_0$, which can be stored as a separate 1D array cell by cell corresponding to the Cell2PDF rows. The type of each PDF array, in terms of one or two shifting operations to perform, can be expressed by the signs of LEN items accordingly.

The SHIFT data structure, as shown in Fig. 2, can be implemented in a straightforward fashion. A SHIFT LB can be realized via two separate passes. The propagation operation with SBB can be implemented by a pass over AID and LEN, one pair of items at a time; it determines the two ends of the corresponding PDF array and the circular type of that array and performs one or two circular shifting operations accordingly. On the other hand, the collision operation can be realized in a cell-by-cell pass over the Cell2PDF table row by row. At each row, it does the following: (1) loading all PDFs associated with that cell; (2) calculating the new collision values; and (3) updating the PDFs with the new values. The locations of the arrays and the positions of those PDFs in each of the arrays are determined from the corresponding row in the Cell2PDF table, i.e., $\{(a_{i,j},\ p_{i,j})\}$, as discussed above. Note that $(a_{i,j},\ p_{i,j})$ may be replaced with the exact locations of the two PDFs in the corresponding PDF array to avoid repeating integer arithmetic calculations required for resolving the addresses. Note that a single-pass approach may be developed instead by simply carrying out the propagation on demand prior to step (1) in the collision operation. In this work, the two-pass approach is selected for the reason of simple and efficient implementation. A more efficient implementation of the single-pass iteration might be sought by further exploring relationships between pore cells as demonstrated in [22].

The performance of the collision operation will depend on the arrangement of the PDF arrays in memory, which determines how fast the collision operation can read/write access to the PDFs for every cell. On a typical microprocessor computer system, the optimal access requires the PDF arrays to be arranged as close to each other as possible in the memory space for those pore cells which are geometrically close to each other. This is because the closer are the cells in the same pore (i.e., a group of connected pore cells), the fewer PDF arrays the PDFs of those cells are likely to belong to. Hence, when those PDF arrays are stored as close as possible, the PDFs of each of those cells are likely to be close to one another too. Note that the proximity of PDFs for each cell in memory depends on the length of their corresponding PDF arrays and the number of velocity directions. Obviously the shorter the lengths of the PDF arrays are, the more likely that the PDFs can be arranged close to one another in memory.

Based on the intuitive postulation above, PDF arrays and the Cell2PDF table may be optimized for more efficient collision operations as follows: (1) classify all cells into a number of close-proximity cell groups, in which cells belong to the same pore; (2) reorder the rows of the Cell2PDF table group by group; (3) reindex Cell2PDF by looping through every $(a_{i,j},\ p_{i,j})$ in each row in turn to assign an index (starting from 1) to that pair and every other pair that shares the same $a_{i,j}$ in the same column; and (4) rearrange PDF arrays according to the indices. This procedure results in a new arrangement of the PDF arrays. If all cells, which belong to the given pores, are allowed to be in the same group, the first step of this procedure could be implemented using K-mean classification over cell coordinates, $\{(i,\ j,\ k)\}$. For any given number of groups, the K-mean classification allocates cells into the groups so that the sum of within-group variations is minimized [40]. It will be shown in the next section that this simple procedure can improve the performance of the collision operation for porous models.

Based on the discussion above, an implementation of the SHIFT scheme can be divided into three independent operational units to complete the following three tasks in a pipeline. They are the following: (1) preprocessing a porous model to construct a SHIFT data structure (i.e., decomposing PDFs and optimizing the PDF arrays); (2) performing SHIFT LB operations on that structure; and (3) postprocessing LB results.

C. SHIFT memory requirement

According to Fig. 2, an implementation of SHIFT for performing LB only (i.e., for the second task above) has a memory requirement, M_{shift}, equal to

$$M_{\text{shift}} = \{N_p[N_v S_f + 0.5(N_v - 1)(S_l + S_i)]\} + [N_a(S_l + S_i)], \tag{4}$$

where N_a is the number of the PDF arrays. S_f is the memory size of a floating point value for each PDF. S_l is the memory size of a reference for $a_{i,j}$ and each element in AID, equal to 32/64 bits. S_i is the memory size of an integer for $p_{i,j}$ and each element in LEN. Note that the term in the braces corresponds to the PDF arrays and Cell2PDF and is the same for all models with the same porosity and the same LB lattice structure (e.g., DdQb). The term in the square brackets fater the plus sign, on the other hand, corresponds to LEN and AID, and may vary among models of the same porosity. An apparent, but less useful, upper bound of N_a is $N_p \times N_v$, which holds when every pore cell is isolated from any other. N_a is found to be less than $1.3N_p$ for D3Q15 for all models analyzed in the next section. When $(a_{i,j}, p_{i,j})$ are references to the locations of the two PDFs in a PDF array, $p_{i,j}$ takes the same size of memory as $a_{i,j}$ does. This may require slightly more memory on a 64-bit processor than $p_{i,j}$ being treated as an integer.

Because the pore geometry, tortuosity, and connectivity of a model influence the number of sets of the connected pore cells in every circular chain, that influence must be reflected in N_a. However, N_a is not a useful measurement itself to distinguish one model from another because it is dependent on the model resolution (e.g., same pore structures but different resolutions) and the selections of lattice structures (i.e., the number of velocity directions). To overcome these problems, N_a may be normalized by dividing N_p, the number of pore cells, and $(N_v - 1)/2$. This leads to a new quantity, the NAPC. In the next section, NAPC and the profile of lengths of the PDF arrays will be shown to be able to differentiate the pore structures within a set of different models, and it explains the performance variations.

IV. NUMERICAL EXPERIMENTS

Numerical experiments have been conducted to explore the behaviors of SHIFT. More precisely, the following aspects of SHIFT are examined: (1) the memory requirement and reduction with respect to the geometric and topological characteristics of the models; (2) the performance with respect to PDF array arrangements; (3) the peak performance with respect to the model size; and (4) the incurred overhead in collision in comparison with a "fastest" collision implementation. To these ends, four binary voxel models of the same size, 512^3, but having different characteristics, are chosen for this analysis, with subvolumes from them being used in the experiments. Of the models, three are obtained from samples of natural sandstones, Fontainebleau sandstone, Castle-gate sandstone, and an unconsolidated fluvial sand pack by means of x-ray computed tomography. The other one is a synthetic model of packed spherical beads. They are labeled as FB, CG, LRC, and BP, respectively. The latter three models are available from [50]. These four models have different characteristics in terms of their porosity (from 10% to 36%) and the geometry and topology of the pore structures. Further information on the latter three models is available from [50] and can be found in the work of Sheppard *et al.* [41].

A D3Q15 implementation of SHIFT, lb_shift, is employed in this work. A preprocessing code is implemented to derive a SHIFT data structure for each model to set up the SHIFT LB simulation for lb_shift. Postprocessing is not considered here. lb_shift has been validated by comparing the results against analytical solutions for some models containing simple pore geometry like circular channels and against the results obtained from a modified version of a public LB code, SUNLIGHTLB (see [51,42]). SUNLIGHTLB implements a D3Q15 LB with a single set of PDFs for solid and pore cells to simulate the incompressible Navier-Stokes and Stokes flows. It is also capable of handling moving solid particles for modeling particle suspensions. Internally, it arranges PDFs as a single array in a cell-by-cell and thus collision-optimized fashion using a layout similar to the grid compression of Pohl *et al.* [28]. This code has been modified for this test by removing the implementations for extra features, which are not required in this work, resulting in a modified version, called lb_base. Both lb_shift and lb_base use body forces to drive fluid and apply the SBB on solid walls. To evaluate the fourth aspect as accurately as possible, the collision operation in both codes has been implemented with exactly the same number of arithmetic operations and the same expressions in C/C++ except for reading and writing PDFs. Note that lb_base skips the collision operation on solid cells, and thus is considered to be the fastest prior implementation relative to the collision operation.

A. Procedure

In order to evaluate the four aspects above robustly and reliably, a group of nine submodels are extracted from each of the four original models at each of three subvolumes of DIM^3, where DIM takes 100, 150, and 200. For each submodel, its cell at (DIM/2, DIM/2, DIM/2) coincides with one of the following nine cells in the original model: (256, 256, 256), (100, 100, 100), (412, 100, 100), (100, 412, 100), (412, 412, 100), (100, 100, 412), (412, 100, 412), (100, 412, 412), or (412, 412, 412). A model is valid if it contains at least one component of connected pore cells that intersects the two opposite domain faces perpendicular to the X axis. For each valid model, isolated pore components are removed and the Stokes flow along the X direction is simulated. Since the SHIFT behaviors are of primary interest, no additional treatment is applied to make asymmetric models symmetric along the X direction. If the calculation of model permeability were of concern, one would choose to add a mirror of each model along the X axis and then to simulate or to modify predensity or postdensity quantities to mimic fluid behaviors as in a symmetric model [13].

For each group, all valid models are preprocessed to derive their respective SHIFT data as well as the estimates on the following quantities. They are the following: the SHIFT memory requirement (MR), in megabytes (Mb); the memory saved (MS) relative to lb_base, in percentage; the number of pore (NP) cells in a model; the model porosity (Φ); the number of PDF arrays (NA); and the number of PDF arrays re-

TABLE I. The number of valid models for every case, FB, CG, LRC, and BP, at each model size; the rearrangement settings for PDF arrays (i.e., the number of arrangements and the number of clusters in each arrangement) and the number of LB iterations used in LB simulation.

Size	No. of valid models				Rearrangement settings for PDF arrays (number of arrangements and number of clusters in each arrangement)	Iterations
	FB	CG	LRC	BP		
100^3	8	9	9	9	40, 60, 80, 100	1000
150^3	9	9	9	9	20, 40, 60, 80, 100 150, 200, 300, 400, 500, 600	500
200^3	9	9	9	9	20, 40, 60, 80, 100 150, 200, 300, 400, 500, 600	200

quiring two circular shifting operations (NT). An estimate is obtained by averaging the values of a quantity over all valid models.

In order to evaluate aspects (2)–(4), for each valid model lb_base is run once. lb_shift is run once with the default arrangement of PDF arrays followed by several additional runs. At each additional run, a new arrangement is obtained from the same default arrangement; a *k*-mean code, KMLOCAL [43] is used to classify pore cells into a number of clusters, according to their closeness as described in the previous section; the number of the clusters is chosen to take several values in the ascending order, one by one. For each model size, the number of the clusters and the number of LB iterations are chosen so that they are large enough to obtain stable performance measurements (see Table I).

Table II shows the estimates of the measures collected from the preprocessing stage for every case and model size. The coefficients of variation (CV) for every quantity are listed in brackets. According to CVs, the valid models in each group become more similar with the increase in the model size. The MS is calculated as $(M_{\text{base}}-M_{\text{shift}})/M_{\text{base}}$. M_{base} is the memory requirement by lb_base, equal to $N[N_v S_f+1.25S_i]$, where N is the total number of pore and solid cells. The first and second terms correspond to the memory required for storing PDFs and for storing the masks and pore or solid flags in lb_base. M_{shift}, as defined by Eq. (4), is taken to be the average of memory requirements for all valid models for each case and model size in this calculation.

Performance was measured in terms of mega/million lattice updates per second (MLUPS) for pore cells. For each model and each PDF arrangement, MLUPS was calculated from the CPU times (in seconds) taken to complete the combined propagation and bounce-back operation (PropOp) and the collision operation (CollOp) of lb_shift, respectively. The CollOp performance of lb_base was determined for each model, too.

All experiments were carried out on a microprocessor computer system with a 64-bit dual-core AMD Opteron 2218 processor at 2.6 GHz running Linux 2.6.18. Because lb_shift and lb_base were implemented as single threads, respectively, MLUPS measurements may be considered to correspond to only one processor. The CPU times were taken by calling the clock() function. All codes were compiled using GCC 4.1.2 with options O3 tuned for Opteron.

B. Results and analysis

1. Memory reduction

Figure 3 shows MS vs the averaged porosity for the model size of 150^3. As expected, there is a strong negative linear relationship between them. The total number of PDF arrays has only a secondary effect on MS. Figure 3 also shows the relationship between NAPC and the porosity at the same model size. As postulated previously, NAPC ought to

TABLE II. Averaged quantities for all valid models for each case and model size. Each value in a bracket is the coefficient of variation in the corresponding quantity.

Size	Case	MR(Mb)	MS	NP	Φ	NA	NT
100^3	FB	18.84 (29.99%)	84.10% (5.67%)	88644 (30.37%)	8.88% (27.85%)	109635 (25.24%)	0.00 (0.00%)
	CG	43.04 (11.83%)	63.68% (6.75%)	203242 (12.32%)	20.44% (11.78%)	237790 (5.63%)	0.00 (0.00%)
	LRC	70.84 (26.56%)	40.21% (39.49%)	343171 (26.61%)	34.44% (26.15%)	241895 (25.47%)	11.00 (114.75%)
	BP	79.49 (11.91%)	32.92% (24.27%)	392890 (12.12%)	39.33% (12.10%)	135667 (5.89%)	747.56 (68.87%)
150^3	FB	71.24 (6.50%)	82.19% (1.41%)	335494 (6.59%)	9.78% (8.04%)	409548 (6.02%)	0.00 (0.00%)
	CG	145.60 (7.84%)	63.59% (4.49%)	689108 (8.08%)	20.33% (8.36%)	778398 (5.37%)	0.00 (0.00%)
	LRC	242.98 (13.24%)	39.24% (20.49%)	1178046 (13.24%)	34.89% (13.13%)	812623 (13.35%)	0.44 (215.06%)
	BP	255.64 (8.50%)	36.08% (15.06%)	1265303 (8.67%)	37.33% (8.38%)	405862 (3.18%)	360.67 (70.79%)
200^3	FB	176.13 (4.01%)	81.42% (0.91%)	830114 (4.09%)	10.44% (4.76%)	1001516 (3.77%)	0 (0.00%)
	CG	345.63 (6.60%)	63.54% (3.79%)	1636663 (6.78%)	20.56% (6.92%)	1832300 (4.61%)	0 (0.00%)
	LRC	594.91 (7.05%)	37.24% (11.88%)	2888717 (7.06%)	36.11% (7.20%)	1912638 (7.49%)	0 (0.00%)
	BP	619.12 (5.49%)	34.69% (10.34%)	3068210 (5.61%)	38.22% (5.89%)	917206 (2.01%)	266 (81.93%)

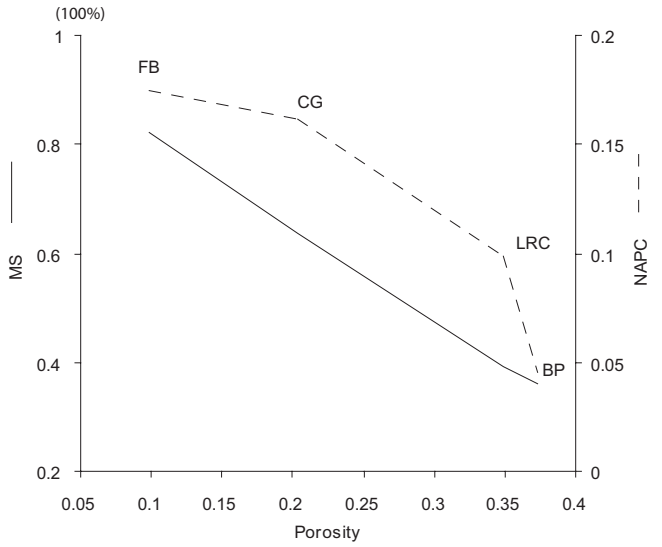

FIG. 3. Memory reduction and NAPC as a function of the model porosity, respectively.

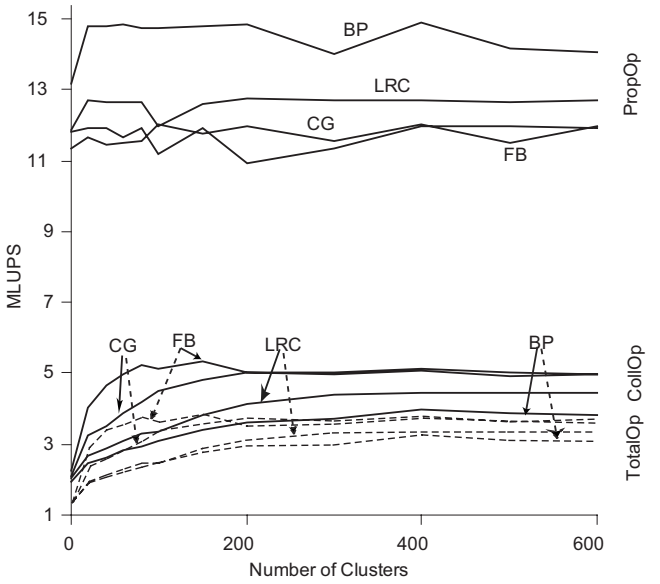

FIG. 4. MLUPS of CollOp, PropOp, and TotalOp (dotted lines) as a function of the number of clusters, NCLS, for FB, CG, LRC, and BP at the model size of 150^3, respectively.

measure the geometry, tortuosity, and connectivity of pore structures in addition to the porosity and is, therefore, likely to vary nonlinearly as a function of the porosity from model to model. As shown in Fig. 3, NAPC indeed decreases with the increase in the porosity but nonlinearly at a much steeper gradient from LRC to BP than from FB to CG. One should note from Table II that the ratio between the porosity difference of the former pair and that of the latter pair is about 1/5. Hence the figure indicates that LRC and BP seem to have different pore structures that cannot be explained away by the porosity alone. The corresponding plots at two other model sizes are not shown since they have similar patterns.

2. Performance vs PDF arrangement

The SHIFT performance is expected to depend on the characteristics of pore systems and the arrangements of PDF arrays in memory. To investigate this, for each case and size, averaged performances of lb_shift PropOp and CollOp over all valid models were calculated for every rearrangement setting (see Table I). Figure 4 shows these averaged values as a function of the rearrangement settings in terms of the number of clusters at the model size of 150^3. The total performance (TotalOp), i.e., the harmonic average of performances of PropOp and CollOp, is also shown. The results at other model sizes are not shown because they have similar patterns.

In terms of MLUPS, Fig. 4 shows that CollOp increases with an increase in the number of clusters at a steeper rate from 0 to 200/300 for CG, LRC, and BP. It peaks around 400 and keeps almost constant afterwards. For FB it peaks before 100, oscillates slightly between 100 and 200, and keeps almost constant afterwards. CollOp runs at least twice as faster at its peak for every case than with the default arrangement, assumed to have the zero number of clusters in the figure for display. This shows that rearranging PDF arrays can improve the CollOp performance.

CollOp is also shown to increase more sharply at the smaller number of clusters (<200) for low porosity cases, FB and CG, than high porosity cases, LRC and BP. The reason for this could be that a low-porosity model has more short than long PDF arrays so that they can be arranged more closely to achieve a better performance using a smaller number of clusters. Figure 5 shows the adjusted length frequency of PDF arrays at the model size of 150^3. The frequency is adjusted by dividing the corresponding averaged porosity over all valid models for each case in order to reduce the effect induced solely by porosity differences. As shown in the figure, FB and CG follow a similar pattern of the length distributions except that FB has more short and less long arrays than CG does. As expected, LRC and BP depart from FB and CG in their distributions. However, there is a significant difference in the distributions between LRC and BP although they have similar porosities. Synthetic BP has much fewer short arrays than LRC does, but more long arrays, with many being circular chains.

On the other hand, PropOp achieves much greater MLUPS than CollOP does. This is anticipated because PDF arrays are arranged in favor of lb_shift PropOp. Unlike CollOp, PropOp achieves greater MLUPS for high than low po-

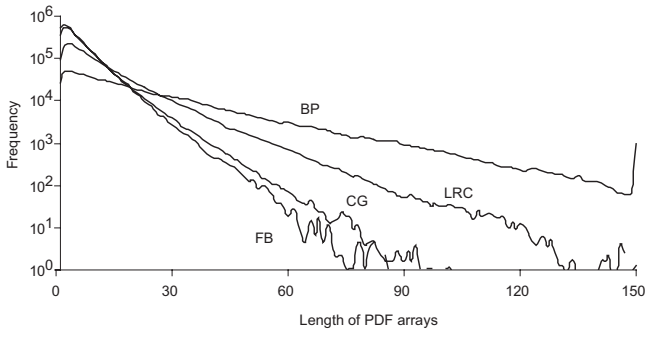

FIG. 5. Adjusted length frequencies of PDF arrays for every case at the model size of 150^3.

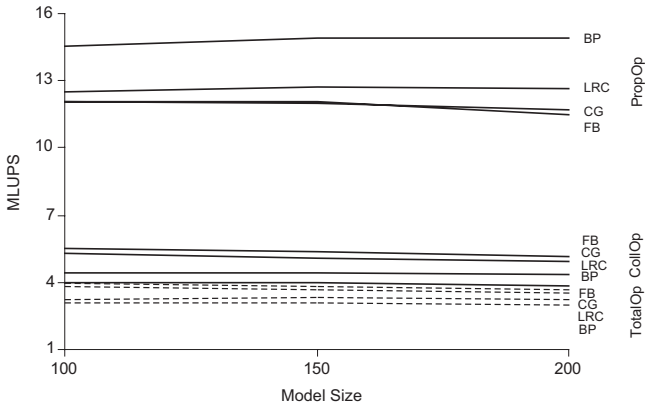

FIG. 6. Peak performances as a function of the model size.

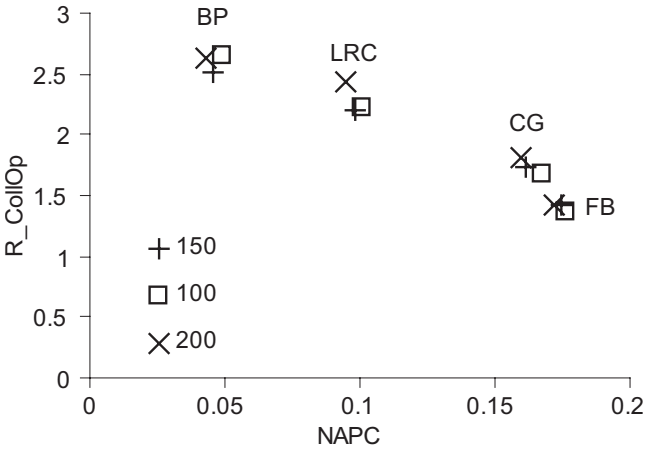

FIG. 7. Performance ratio of MLUPS of lb_base CollOp over that of lb_shift CollOp as a function of NAPC for all cases and model sizes.

rosity models. This is because a high-porosity model tends to have more long PDF arrays and therefore to incur less shifting operations than a low-porosity counterpart does. It is not difficult to note that BP achieves a much higher MLUPS than LRC does, relative to the other pair. This may be because BP contains more long and two-circular-shift PDF arrays than LRC does, as shown in Fig. 5, despite BP having only about 3% more pore cells than LRC does (see Table II). Note that if there are too many two-circular-shifting PDF arrays, PropOp may be slowed down. However, when the resolution of a model is high enough, the number of two-circular-shifting PDF arrays is likely to become relatively small.

Unlike CollOp, PropOp oscillates around a constant value for each case. The exact causes of these oscillations are not clear to the authors although they might be attributed to the performance fluctuation of the computer systems due to data misfetching in memory caches. This issue is not pursued further in this work.

3. Peak performance vs model size

For each case (FB, CG, LRC, and BP) and all PDF arrangements at every model size, the respective peak performances of PropOp and CollOp for lb_shift were obtained. Figure 6 shows the peak performances vs the model size for each case. The TotalOp is shown by dotted lines. In terms of MLUPS, both PropOp and CollOp remain almost constant across the model sizes for every case, and increase and decrease, respectively, from FB, CG, LRC to BP. PropOp is much greater for BP than for any other case because it contains more long PDF arrays. As far as the total performance TotalOp is concerned, SHIFT performs better for models with low than high porosities.

4. Impact of pore structures on the collision performance

Since SHIFT is suited naturally for the propagation operation, it is expected to perform worse than lb_base in the collision operation. As discussed above, the performance of lb_shift CollOP depends critically on how closely the PDF arrays can be arranged in memory for cells belonging to the same connected pores. This is influenced by the geometry and topology of the pore structures of a model. Since NAPC ought to reflect pore characteristics, it is used here to explain the computational overhead incurred in lb_shift CollOp due to irregular data access via Cell2PDF.

Figure 7 plots the ratio of MLUPS of lb_base CollOP over that of lb_shift CollOP, denoted as R_CollOp, averaged over models for every case and every model size, as a function of NAPC. Note that the lb_shift CollOP performances are the corresponding peak performances. As shown, there is a negative but slightly nonlinear relationship between R_CollOp and NAPC. R_CollOp decreases from about 2.5 to below 1.5 from BP to FB, suggesting that lb_shift takes 0.5–1.5 times longer than lb_base, solely for accessing the data. There is a trend that a low-porosity case incurs less computational overhead than a high-porosity one.

V. DISCUSSION AND REMARKS

SHIFT has been shown to be capable of achieving several times better performance in PropOp than in CollOp for all cases considered in this work. This is because of the way in which SHIFT arranges PDFs. Hence an important element in constructing the SHIFT data structure is to find an optimal arrangement of PDF arrays so that the PDFs of those pore cells belonging to the same pores are close one another in memory. It has been demonstrated that even a simple technique, such as the *K*-mean clustering, can be used to improve the CollOp performance. However, the performance reaches its plateau when the number of clusters goes beyond a limit. This might be because all PDF arrays may have been arranged closely or in such a way that prevents CollOp from gaining a better performance. Neither of these possibilities has been fully understood yet. For the former, a scenario is that there are too many large pores in a model. One scenario for the latter is that the pore cells might be classified into the same cluster though they may not actually belong to the same set of connected and nearby pore cells. This may result in some PDF arrays, which should have been positioned closer to each other in memory, become further apart. So the performance gained from the properly positioned clusters is cancelled out by the performance overhead incurred from the improperly positioned ones. The authors believe that there is

still room to improve the performance of CollOp by developing specific classification techniques using the geodic rather than Euclidean distances of pore cells within pore space.

Although SHIFT is described using the BGK LB model for a single component fluid, it is possible to implement SHIFT with other LB models for simulating single-component/multicomponent/single-phase/multiphase fluid flow [11,44], thermal fluid flow [45], and chemical transport [10]. In addition, the multiple-time relaxation LB [37,38] may be implemented with SHIFT to reduce the viscosity dependency of simulated velocity.

Because no solid boundary cell is involved in SHIFT, the SBB implemented here behaves slightly different from the typical one where the PDF of a pore cell is reflected back to its opposite direction only at the next propagation via an adjacent solid cell. This typical behavior could be enforced by simply adding one or two dummy elements into each PDF array, at position(s) before the head and/or after the tail accordingly. This will incur only few changes to the whole scheme. Other boundary conditions, based on high-order interpolations of curved walls [31,32] and multiple reflections [35,36], might also be implemented with SHIFT to obtain more accurate no-slip approximations. However, these schemes are likely to incur even more performance overheads because they need to access the PDFs of neighboring cells. This makes it difficult in SHIFT to handle the boundary condition efficiently. However, SHIFT may be extended to use the interpolation method of [34]. It would be of interest to investigate what advantages this boundary condition would bring to flow modeling in natural porous materials of low porosity. In this situation, more accurate representations of solid and pore interfaces may be required [46].

As found from the experiments, the length distribution of the PDF arrays and NAPC could distinguish pore structures even for models with very similar porosities and explain the performance variations in SHIFT. Therefore, it would be worthwhile to investigate them further because the lack of proper combined geometric and topologic measurements has been one of the impediments to establishing the relationships between the fluid flow and pore structures. The authors and others have been developing appropriate measurements [47] for porous media and have applied them to correlate the fluid flow and transport properties using a network fluid flow modeling approach [48].

The preprocesses for SHIFT may be utilized to identify the narrow paths at pores that may have adverse impacts on the LB simulations. To do this, one can replace every item in the SHIFT Cell2PDF table with the length of the corresponding PDF array. This gives a length distribution for every cell at every velocity direction. Given a threshold in terms of the number of pore cells, one may define a narrow path at a pore if there is only one PDF array that passes through a cell in that pore and is longer than the threshold. In this case, one can identify all narrow paths by simply counting the rows in the table that have only one item greater than the threshold. The application of this and SHIFT may be used to investigate flow impact of the narrow paths in low-porosity realistic porous models.

Although SHIFT is shown to achieve good computational performances, the intensity of LB computation calls for parallelization. In an environment consisting of a cluster of computers, parallelization of SHIFT needs to subdivide the SHIFT data structure (Fig. 2) into smaller units. When two units share different parts of one or more PDF arrays, the related collision operations have to be synchronized. The main challenge here is to do this in such a way that decomposed units have even load (the amount of computations) and share as few PDF arrays as possible. How this can be done and what impacts it may have on the overall performance are worth investigating.

In this work, the numerical experiments were carried out using particular porous models and a single computational environment, and both of these factors are different from those used in other work. Hence, it is not possible to directly compare the merits of SHIFT with other schemes (see [29]) based on published results, except the compressed grid memory layout used in lb_base. Some of those schemes, such as the swap schemes of Mattila *et al.* [22], are known to have better properties in terms of computational efficiency and memory requirements than others. However, those experiments were carried out on porous models with simpler pore structures [29]. It is known that, for the schemes that store only fluid cells, they might perform differently on porous models with different pore structures but same or similar porosities. It is, therefore, the interest of the authors to thoroughly and systematically compare SHIFT with those schemes on the identical computer system in a separate work. There, the authors would use a much wider range of realistic models of low-porosity porous media obtained by 3D x-ray tomography and an advanced stochastic modeling technique that has been developed by the authors and the others [49].

VI. CONCLUSIONS

This paper proposes a computational scheme, SHIFT, for lattice Boltzmann simulation in natural low-porosity media. The following conclusions can be drawn:

(1) SHIFT makes explicit use of pore structures and decomposes PDFs into a set of 1D arrays along each pair of velocity directions of a given LB lattice structure; it needs to store a single set of PDFs for pore cells only;

(2) The arrangement of each PDF array in SHIFT allows LB propagation and standard bounce-back operations to be combined in one or two circular shifting operations on that array; it achieves good performance in propagation;

(3) SHIFT enables the access to all PDFs for every pore cell through a SHIFT indexing scheme, sufficient for implementing LB collision operations for many LB models;

(4) For all tested porous models, with porosity ranging from 10% to 38%, a D3Q15 SHIFT implementation shows a reduction in the memory requirement by 36% and 82%, compared with a comparable D3Q15 implementation that stores a single set of PDFs for both solid and pore cells;

(5) For all tested cases, the D3Q15 SHIFT implementation achieves a minimum performance over 11 and 3.8 MLUPS in the propagation and bounce-back operations, and

the collision operations, respectively, and therefore a minimum of 2.8 MLUPS in total on a single computer with an AMD Opteron 2218;

(6) The numerical experiments show that rearranging PDF arrays in the Cell2PDF table is necessary to achieve better performances in the collision operations. By using a simple *K*-mean technique over cell indices to rearrange PDF arrays, it is shown that the collision performance is improved by a factor of no less than two compared with those obtained using default arrangements;

(7) The performance variations among the cases are found to associate with both the geometry and topology of their pore structures. The propagation does better for high than low porosity cases, whereas the collision does better for low than high porosity cases;

(8) The computational overhead in the SHIFT collision operation, compared with the fastest collision operation, is found to correlate with the number of PDF arrays per pore cell, NAPC;

(9) NAPC and the adjusted length frequency of PDF arrays are found to be able to distinguish the pore structures even for models with similar porosities. They may be considered as useful indicators of the pore structures.

In the future, the authors wish pursue along those lines identified in the previous section.

ACKNOWLEDGMENTS

The authors are grateful to Marinus I. J. van Dijke and Kenneth S. Sorbie at the Institute of Petroleum Engineering, Heriot-Watt University, UK, and two anonymous reviewers for their useful comments that helped to greatly improve the quality of our manuscript. The authors wish to thank Mark A. Knackstedt at the Department of Applied Mathematics, Research School of Physical Sciences and Engineering, Australian National University, Australia, for allowing us to use three porous models in this work. This work was supported in part by UK Engineering and Physical Sciences Research Council (EPSRC) Grant No. EP/D002435/1.

[1] G. R. McNamara and G. Zanetti, Phys. Rev. Lett. **61**, 2332 (1988).
[2] F. J. Higuera and J. Jimènez, Europhys. Lett. **9**, 663 (1989).
[3] A. J. C. Ladd, J. Fluid Mech. **271**, 285 (1994).
[4] X. He and L.-S. Luo, Phys. Rev. E **56**, 6811 (1997).
[5] S. Chen and G. D. Doolen, Annu. Rev. Fluid Mech. **30**, 329 (1998).
[6] R. R. Nourgaliev *et al.*, Int. J. Multiphase Flow **29**, 117 (2003).
[7] S. Succi *et al.*, Europhys. Lett. **10**, 433 (1989).
[8] N. S. Martys and H. Chen, Phys. Rev. E **53**, 743 (1996).
[9] X. Zhang *et al.*, Adv. Water Resour. **25**, 1 (2002).
[10] X. Zhang *et al.*, Water Resour. Res. **41**, W07029 (2005).
[11] C. Pan, M. Hilpert, and C. T. Miller, Water Resour. Res. **40**, W01501 (2004).
[12] C. Pan, L.-S. Luo, and C. T. Miller, Comput. Fluids **35**, 898 (2006).
[13] J. T. Fredrich, A. A. DiGiovanni, and D. R. Noble, J. Geophys. Res. **111**, B03201 (2006).
[14] D. Zhang *et al.*, Geophys. Res. Lett. **27**, 1195 (2000).
[15] A. Kameda *et al.*, Geophysics **71**, N11(2006).
[16] A. Koponen, D. Kandhai, E. Hellen, M. Alava, A. Hoekstra, M. Kataja, K. Niskanen, P. Sloot, and J. Timonen, Phys. Rev. Lett. **80**, 716 (1998).
[17] C. Manwart, U. Aaltosalmi, A. Koponen, R. Hilfer, and J. Timonen, Phys. Rev. E **66**, 016702 (2002).
[18] M. Schulz *et al.*, in *The Third International FORTWIHR Conference on HPSEC*, edited by M. Breuer, F. Durst, and C. Zenger (Springer-Verlag, Berlin, 2001), p. 115.
[19] N. S. Martys and J. G. Hagedorn, Mater. Struct. **35**, 650 (2002).
[20] C. Pan, J. F. Prins, and C. T. Miller, Comput. Phys. Commun. **158**, 89 (2004).
[21] J. Wang, X. Zhang, A. G. Bengough, and J. W. Crawford, Phys. Rev. E **72**, 016706 (2005).
[22] K. Mattila *et al.*, Comput. Phys. Commun. **176**, 200 (2007).
[23] P. L. Bhatnagar, E. P. Gross, and M. Krook, Phys. Rev. **94**, 511 (1954).
[24] Y. H. Qian *et al.*, Europhys. Lett. **17**, 479 (1992).
[25] X. He and L.-S. Luo, J. Stat. Phys. **88**, 927 (1997).
[26] G. Wellein *et al.*, Comput. Fluids **35**, 910 (2006).
[27] H. Sagan, *Space-Filling Curves* (Springer-Verlag, New York, 1994).
[28] T. Pohl *et al.*, *Parallel Processing Letters* (World Scientific, Singapore, 2003), p. 549.
[29] K. Mattila *et al.*, Comput. Math. Appl. **55**, 1514 (2008).
[30] X. He *et al.*, J. Stat. Phys. **87**, 115 (1997).
[31] M. Bouzidi, M. Firdaouss, and P. Lallemand, Phys. Fluids **13**, 3452 (2001).
[32] R. Mei, D. Yu, W. Shyy, and L. S. Luo, Phys. Rev. E **65**, 041203 (2002).
[33] M. Junk and Z. Yang, Phys. Rev. E **72**, 066701 (2005).
[34] B. Chun and A. J. C. Ladd, Phys. Rev. E **75**, 066705 (2007).
[35] I. Ginzburg and D. d'Humieres, Phys. Rev. E **68**, 066614 (2003).
[36] I. Ginzburg, Adv. Water Resour. **28**, 1196 (2005).
[37] D. d'Humières, Philos. Trans. R. Soc. London, Ser. A **360**, 437 (2002).
[38] R. Adhikari *et al.*, Europhys. Lett. **71**, 473 (2005).
[39] D. Kandhai *et al.*, J. Comput. Phys. **150**, 482 (1999).
[40] J. B. MacQueen, in *Proceeding of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* (University of California Press, Berkeley, 1967).
[41] A. P. Sheppard *et al.*, in *20th International Symposium of the Society of Core Analysts* (Society of Core Analysis, Trondheim, Norway, 2006), p. SCA2006/26.
[42] P. Warren, Int. J. Mod. Phys. C **8**, 889 (1997).
[43] T. Kanungo *et al.*, Comput. Geom. **28**, 89 (2004).

[44] X. Shan and H. Chen, Phys. Rev. E **47**, 1815 (1993).
[45] M. Yoshino and T. Inamuro, Int. J. Numer. Methods Fluids **43**, 183 (2003).
[46] B. Ahrenholz, J. Tölke, and M. Krafczyk, Int. J. Comput. Fluid Dyn. **20**, 369 (2006).
[47] Z. Jiang *et al.*, Water Resour. Res. **43**, W12S03 (2007).
[48] Z. Jiang *et al.*, Adv. Eng. Mater. (to be published).
[49] K. Wu *et al.*, Transp. Porous Media **65**, 443 (2006).
[50] http://xct.anu.edu.au/network_comparison/#data_sets
[51] http://sourceforge.net/projects/sunlightlb/