Simulation of Dual Site-Bond Network Modeling for Porous Media through the Graphics Class in Java

Benjamin Moreno-Montiel, Carlos-Hiram Moreno-Montiel, Jacqueline Luna-Sánchez, and René MacKinney-Romero

***Abstract*— In the last four decades the advance in the study of porous surfaces has been considerable mainly due evolution of new technologies that allow theoretical and experimental study by Computational Simulations. In this work we will show one of these simulations, we use the Dual Site - Bond Network Modeling for porous media using the Graphics class of Java. In this model, the largest holes in porous media are modeled with Spheres or Sites and the smaller ones or that connect a Site with another one is modeled with Cylinders or Bonds. The objective of the Dual Site - Bond Network Modeling is create three - dimensional models with these two entities to later have a simulation of Mercury Intrusion and Retraction. The construction of these three-dimensional networks is a complex task, since thousands or millions of entities that must be sort under a series of rules defined in the Dual Site - Bond Network Modeling. In this work we will show each of the algorithms that were considered to achieve the simulation of construction of three-dimensional porous media based on the Dual Site - Bond Network Modeling. We will review the evaluation of each implemented version and finally we will show the graphic application developed through the Graphics class.**

***Index Terms*—Bound, capillary phenomena, chemical, computational simulations, elementary errors, geometric errors, graphics class, porous media, sites.**

I. Introduction

All materials of nature have holes or cavities or pores that are connected, allowing to describe some physicochemical characteristics of materials. Even in the most compact material, such as metal, pores appear in the atoms that make them up. Another example is wood [1], due to its nature this material has a greater number of pores on its surface and depends on these pores is the quality and durability of the wood.

This last material is important, since only when the spaces of the holes of a material have sufficiently large dimensions, and are sufficiently interconnected, as to be invaded by the molecules of other substances, then a variety of phenomena appeared, which are due precisely to existence of an extensive hollow-matter interface. In physicochemical when we have these characteristics in materials, it is said that we have a porous medium.

There is a wide variety of porous media, natural materials such as wood, rocks, minerals, bones, lung tissues, etc. and they are also presented in materials that are synthesized for use in some industrial applications such as catalysis, gas storage, filtering and separation of fluids, etc [2].

The uses in the study of porous media are very extensive. One study in which the porosity is applied is for the prevention of plagues that invade the wood like woodworm and termites [3]. The Woodworm leaves its eggs in the pores of the wood where its larvae build galleries in which they live, feeding on this material for approximately three years. After this time the insects leave the wood making round holes of 1.5 to 3mm in diameter. To avoid this plague a protective liquid is applied, such as varnish or ink. In this way the active principles of the substance remain fixed in the wood, thus preventing the female from laying the eggs. Figure 1 shows an image of wood with woodworm.

Fig. 1. Wood surface with woodworm.

One important aspect of a porous medium is the existence of an important hollow-matter interface. When the molecules of a fluid that invades the pore, space exist a variety of physicochemical phenomena, among which are the capillary processes. The capillary processes are those processes in which two or more fluids compete for the possession of the porous space, when the adhesion forces and capillaries are dominant. An example of capillary processes is presented in mercury penetration porosimetry techniques, specifically the intrusion and retraction of mercury [4]-[5].

It is important to know the textural properties of a material or porous medium, since one could predict the actual behavior of the capillary phenomena that occurred within them [2]-[6]. It is not yet possible to fully know these properties because many porous media possess chaotic and complex structures that are difficult to describe quantitatively [7]. Therefore, it was necessary to create a theory that adequately predicts the behavior of capillary processes and that can be proven or supported by experimental evidence.

One of these theories is the Dual Site - Bond Network

Manuscript received July 15, 2019; revised October 10, 2019.

Benjamin Moreno-Montiel, Jacqueline Luna-Sánchez and René MacKinney-Romero are with the Universidad Autónoma Metropolitana – Unidad Iztapalapa, Departamento de Ingeniería Eléctrica, México (e-mail: bmm@xanum.uam.mx, jacquiluna32@gmail.com, rene@xanum.uam.mx).

Carlos-Hiram Moreno-Montiel is with Universidad Tecnológica de México – Plantel Sur, Ingeniería en Sistemas Computacionales, México (e-mail: hiramoreno@gmail.com).

Modeling for the simulation of porous media [8]. By means of this model it is possible to visualize a solid as a porous network constituted by two types of hollow elements interconnected alternately and inevitably throughout the structure, this structure is shown in Fig. 2.

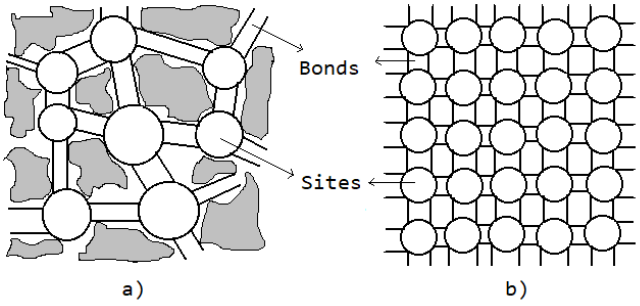

Fig. 2. Representation of the porous medium through sites and bounds. a) Real porous material, b) Porous network in 2D.

The sites (Fig. 2-a) or cavities or dens that for simplicity are modeled as spheres and whose size *RS* corresponds to the size of their radius, are elements in which one or more bonds converge [9]. The bonds (Fig. 2-b), that is, the passages, capillaries or windows that can also be modeled for simplicity as cylinders and whose size *RB* is the radius inscribed in the minimum cross section of the cylinder, which they are passages by which two sites communicate.

In this way we can visualize the porous solid formed by two collections of different entities (in shape and size), which can be represented by two size distribution functions, one for the sites *FS(R)* and another for the bonds *FB (R)*, which should be standardized and can be expressed on the basis of the total number of porous elements of each class, that is, the sites or bonds, this distributions we can see in Equation (1)

$$S(R) = \int_0^R F_S(R)dR\,; \qquad B(R) = \int_0^R F_B(R)dR\,; \qquad (1)$$

where, $S\,(R)$ and $B\,(R)$ represent the probabilities to find a site or bond, respectively, of size R or smaller.

The Dual Site - Bond Network Modeling can be simulated in a computer, however, there is a high complexity implicit in its construction. Since, when handling miles and even millions of entities, the ordering of these based on a series of laws (they will be revised in the Previous Work) complicates the phase of construction of the porous medium. This is the motivation of this paper, since we will show a different algorithm of how to construct a porous medium and how to visualize it by means of a Simulator developed in Java with the use of the Graphics Class.

This simulator that we present in this paper besides using different algorithms for its construction explores another programming language different from the one used in past versions. In this case, we decided to use the Java language due to its object-oriented programming paradigm.

This paradigm allows programming to be closer to reality, modeling is done through classes and instances of these that are called objects. In our case, the porous medium is modeled as a class and each network that is generated is an object of this class. In this paradigm we also have attributes and methods, for this simulation the attributes are the sites and bonds that make up the porous medium and the methods or behavior are the algorithms to correct the errors that appear in the construction phase that we will see in the section of Previous Work.

This paper is organized as follows. In Section II, we describe the complexity to construct a porous network and the algorithms that are used to implement these networks. In Section III, we described the model based on object-oriented programming that was used to perform this simulator, as well as the algorithms to fix the errors present in the construction of a porous medium. In Section IV, we present the experiments that were performed using the Simulation of Dual Site - Bond Network Modeling for Porous Media through the Graphics class in Java. Finally, we will present some conclusions and future work.

II. Previous Work

A. Dual Site-Bond Network Modeling (DSBM)

The DSBM allows simple construction of porous media of different nature [8]-[10] and to study some capillary phenomena that occur in them, such as: a) nitrogen sorption with the help of analytical methods [11] and Monte Carlo [12]; b) Mercury retraction and intrusion [5], which are special cases of imbibition and drainage respectively; c) bubble growth [13]. The formation of arboreal aggregates of silica particles [14], polymer morphology [15] and aggregation products from the sol-gel transition [16], as well as catalytic surfaces have also been explained and simulated [17].

The DSBM establishes that the hollow space of a porous material is composed of two types of cavities: Sites and Bonds. The Sites (represented by spheres) are connected to each other by means of Bonds (represented by cylinders). Sites and links are characterized by their size, that is, by their radius (radius of the sphere and radius of the cylinder). Each site has a certain number of links, this is called connectivity ($\bar{C}$). In Fig. 3 a 3D representation of a homogeneous porous network is shown using this model with $\bar{C} = 6$.

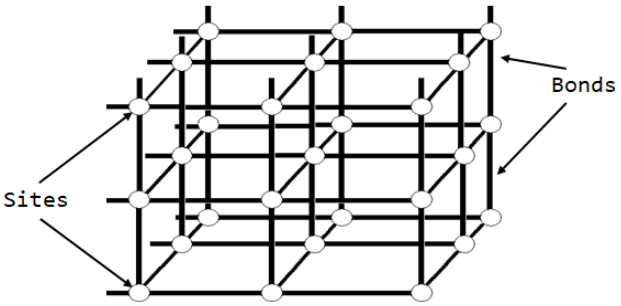

Fig. 3. Homogeneous porous network in 3D.

In Fig. 3, we can see that there is a regular or homogeneous network, that is, each of the sites are connected exactly by the same number of bonds. In this case, the connectivity is 6, that is, each site is surrounded by 6 bonds. In Section III we will show that the developed simulator only considers 3 bonds and the others are shared by the neighboring Site.

Mayagoitia and Kornhauser [2] establish that, if it is desired to build a porous network in a self-consistent manner,

in which the topology and morphology of the sites and bonds that constitute it are taken into account, it is necessary that the following Principle of Construction (PC): "*The size of a site must be greater or at least equal to the size of any of the bonds to which it is attached*".

For the simulation of these networks of porous media the generation of random numbers with Gaussian distribution is used to simulate the radio of the sites and the bonds, and this is based on trying to simulate the random nature. Because these random numbers are used, two types of errors can occur under the PC, called elementary errors and geometric errors, which will be described in the following sections.

B. Elementary Errors

An elementary error is when the radius of the Site is less than one of the radios of the Bounds that surround it. Fig. 4 shows a 3D and 2D representation of these errors.

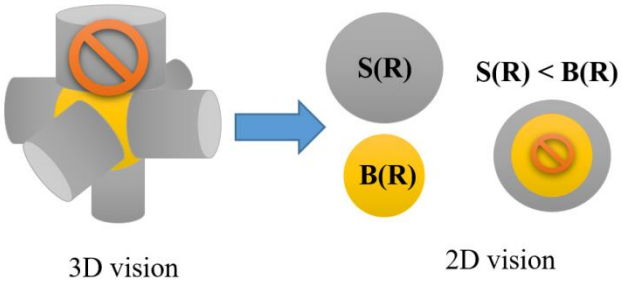

Fig. 4. Representation of elementary errors.

Based on Fig. 4 and the PC, we can enunciate the first rule to construct a porous medium using the DSBM in equation 2:

$$First\ rule = B(R) \leq S(R) \quad \forall R \tag{2}$$

This law applies to the entire network and imposes a restriction on the distribution functions of sizes of sites and bonds. Therefore, the distribution of the sites must always be located on the right or at the most totally overlapped with that of the bonds. In addition, this law ensures that there is a enough bonds of adequate sizes, smaller than the sizes of the sites that connect each bond, that can be connected to all the sites belonging to a given size distribution.

Even if you have an appropriate collection of sites and bonds, that is, that the first law is fulfilled in both size and number, the PC may not be fulfilled if there is a certain degree of overlap Ω (common area) between the two distributions , since there would be some bonds larger than some sites, which when assigned incorrectly (connected to smaller sites), will automatically breach the PC.

In order to avoid this inconsistency, different construction algorithms have been proposed for this type of porous media network. In Sub-Section D we will review one which is discarded for this simulator due to its high degree of complexity when the overlap is very compromised.

C. Geometric Errors

A geometric error in a porous network occurs when the bonds that surround a site are encircled with each other, in Fig. 5 we can see a schematic of a geometric error in a 3D and 2D view.

The geometrical constraints of Figure 5 may or may not be considered in the modeling of the porous medium, because the range of porous materials is very diverse. It goes from those whose geometry is very complex, such as sedimentary rocks, to those of well-defined geometry such as MCM-41 (cylinders not interconnected, hexagonally packed) [18] and SBA-15 (cylinders in hexagonal packing, with walls interconnected by micropores) [19]. These restrictions prevent any possible geometric interference between the sizes of the $\bar{C}$ bonds connected to a site.

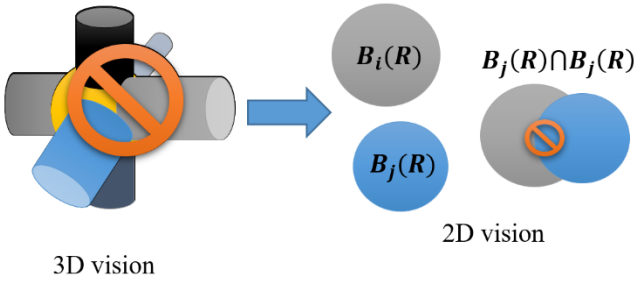

Fig. 5. Representation of elementary errors.

Analytically the way to find the geometric errors is using the formula of the Pythagorean Theorem, formulating the second rule to construction a network of a porous medium which is presented in equation 3:

$$Second\ rule = S(R^2) \geq B_i(R^2) + B_j(R^2) \quad \forall R \tag{3}$$

The fundamental parts that allow the formulation of the second construction rule are shown in Fig. 6, in which a site with two bonds is represented.

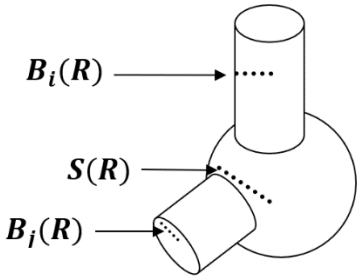

Fig. 6. Fundamental parts of geometric errors.

This strategy of using the Pythagorean Theorem is used for various relaxation methods that are proposed for the correction of geometric errors that a network of porous media could have. In Sub-Section E only these relaxation methods are described.

D. Algorithms to Solve Elementary Errors

After having generated the network of the porous medium through the strategy of generating random numbers with Gaussian distribution, we had a few elementary errors, that is why an algorithm was developed to correct them. These were solved by making shift to the porous network. A shift is when change Sites by Sites or Bonds by Bonds in the network. This change was accepted if the number of errors in the porous network decreased, otherwise the previous configuration was returned. Fig. 7 shows the exchange of a Site-Site.

In Fig. 7 we can see that if the elementary errors decrease

the change is carried out, otherwise the sites are returned to their original position.

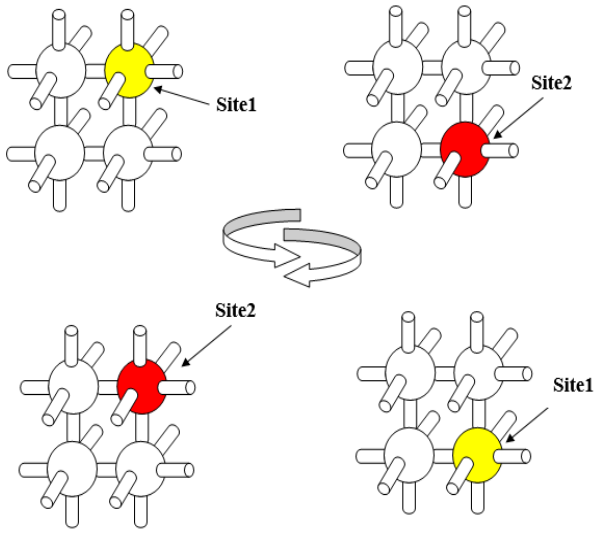

Fig. 7. The shift strategy under bt Sites to fix the elementary errors.

Once the elementary errors were corrected, more shifts were made in the network. This in order to relax the network, that is, to arrive at a configuration in which there were large sites with large bonds and small sites with small bonds.

E. Algorithms to Solve Geometric Errors

For the correction of geometric errors there are four methods of Network Relaxation [9]. The operation of each of them is explained briefly below

- **Relaxation Site:** The radius of the Sites increases a certain percentage (30%) and decreases (1%) until returning to the original radius. In each decrement the geometric errors were corrected by means of a shift in the network. These shifts were random, that is, the sites or the bonds to be exchanged were chosen randomly.
- **Site-Bond Relaxation:** Follow the same procedure as the previous one, except that in this method the Bonds radius is also modified. After correcting the geometric errors with the increase of the sites, the bond radius is increased, and the previous procedure is repeated until the bond radius returns to its original size.
- **Bond-Bond Relaxation:** This procedure follows the same steps as the previous one, except that instead of increasing the radius of the sites, the bond radius decreased a certain percentage (30%). This takes place within a for cycle. Geometric errors are modified while the bond radius is increased until reaching the original radius.
- **Relaxation on the Formula of the calculation of geometric errors:** In this procedure the bonds or the sites are not modified, but the formula of the Pythagorean Theorem is modified. The sum of the squares of the radius of the bonds is multiplied by a variable A. In doing this, the geometric errors are calculated considering the value of A. The modification of the second rule for the construction of porous media networks can be seen in Equation 4:

$$S(R^2) \geq A \times \left(B_i(R^2) + B_j(R^2)\right) \ \forall R \ y \ A \in (0,1) \quad (4)$$

III. Simulation of Dual Site-Bond Network Modeling for Porous Media through the Graphics Class in Java (SIMGRAPH of the DSBM)

A. Creation of the Porous Network

As mentioned in the introduction, SIMGRAPH of the DSBM was created under the object-oriented programming approach. That is why we created a project in the Java language called SimGraph with two implicit classes Node_DSBM and Porous_Network. The first class is the one that represents the structure of a Site with its Bonds.

We mentioned in Sub-Section II_A that the networks that will be simulated will be connectivity six, since each Site will be connected by 6 Bonds. However, in the SIMGRAPH this is not necessary since it only remains to model each Site with only 3 Bonds. This is done in the class Node_DSBM, in which it is modeled that a Site will only have one Left, Front and Up Bond, the part of the code and the modeling of each Site can be seen in Fig. 8.

```
public class Network implements Serializable{
    private double Site_R;
    private double Left_Bond_R;
    private double Front_Bond_R;
    private double Up_Bond_R;

    public RED(double Site_R, double Left_Bond_R, double
       Front_Bond_R, double Up_Bond_R){
          this.Site_R = Site_R;
          this.Left_Bond_R = Left_Bond_R;
          this.Front_Bond_R = Front_Bond_R;
          this.Up_Bond_R = Up_Bond_R;
    }
    ...
}
```

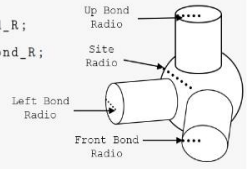

Fig. 8. Code and graphic representation for the creation of a node of the porous network.

In Fig. 8 we can see that each node of the network is made up of only one Site and three Bonds, the question is, how can we obtain connection 6? For this we are going to present a scenario in which three sites are put together, each of them under the model of Fig. 8 will have three bonds. Connectivity 6 is achieved because each site shares its bonds with another site, the left bond will be the right bond of another site, the bond above will be in bond under another site and finally the bond in front will be the behind bond of another site, we can see this in Fig. 9.

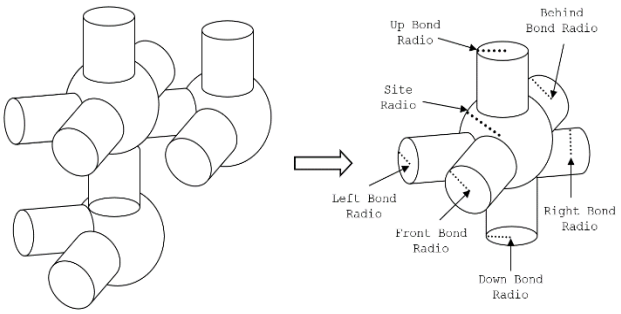

Fig. 9. Node with connectivity 6 under the bond-sharing scheme between each porous network site.

Finally, to achieve a porous three-dimensional network, the Serializable behavior of the Porous_Network class was used. When you have a Serializable class in Java, we can create dynamic arrays of the nodes that compose them, in this case each of them is the one shown in Fig. 8. This is the reason why random numbers are generated for the construction. sites and bonds with Gaussian distribution, setting the value of the mean of each one in order to have a degree of overlap Ω, variable by the user.

Once a network is generated following this strategy, the state of the porous network is completely disordered, having elementary and geometric errors. This is where the algorithms that were proposed in SIMGRAPH come into operation to fix these errors, which will be reviewed in the following Sub Section.

B. Fix the Elementary Errors

As mentioned in the introduction, SIMGRAPH of the DSBM was created under the object-oriented programming approach. That is why we created a project in the Java language called SimGraph with two implicit classes Node_DSBM and Porous_Network. The first class is the one that represents the structure of a Site with its Bonds.

As already mentioned, the elementary errors are when the radius of the Site is less than one of the radios of the Bonds (see Fig. 4). Once having the Network created, we proceed to calculate the number of elementary errors that are found in the Network. For this, the radius of the Site is compared with the radius of the Bonds. If the radius of some Bond is greater than the radius of the Site, the elementary errors increase. In Figure 10 we can see the code of the elementary_ERR method that allows to count the elementary errors.

```
elementary_ERR(RED_G){
  for(int i=0; i < rows; i++)
    for(int j=0; j < columns; j++)
      for(int k=0; k < depth; k++){
        if(RED_G.getSite_R()<RED_G.getUp_Bond_R()) nErrors++;
        if(RED_G.getSite_R()<RED_G.getLeft_Bond_R()) nErrors++;
        if(RED_G.getSite_R()<RED_G.getFront_Bond_R()) nErrors++;
      }
```

Fig. 10. Code to count the elementary errors.

The operation of the algorithm to fix the elementary errors consists of the following stages:

1. The SIMGRAPH uses a a list of lists of lists to build the Network, this can be seen in the code of Figure 10, in which we see that we use a nested cycle of type "for" to travel within the Network, one for the rows, another for the columns and finally one for the depth. The first step consists in passing a list of lists of lists of the sites and the bonds to one-dimensional list, in order to implement the sort method more easily.
2. Once the one-dimensional list of the Sites and Bonds are obtained, they were sort using the Recursive MergeSort algorithm.
3. Once the one-dimensional arrangements are sorted, the Network is reconstructed, but in this case the small sites will be associated with the small bonds and so on until having large sites with large bonds. In this way it was possible to relax the Network so that the intrusion is carried out in a more effective way and that this network is attached to what is a real porous medium. This last capillary process was not implemented in SIMGRAPH but remains as one of the main Future Works of this paper.

In this way the elementary errors are fix and the network is relaxed. After having corrected these errors, we proceeded to correct the geometric errors, in the following Su-Section the algorithm used for its correction is explained.

C. Fix the Geometric Errors

As mentioned earlier, a Geometric error is when the Bonds that surround a Site overlap each other (see Fig. 5). The first step was to use the Pythagorean Theorem to calculate these errors, as explained in Sub-Section II.A. In Fig. 11 we can see the code of the geometric_ERR method that allows to count the geometric errors.

```
public int geometric_ERR(RED_G){
  for(int i=0; i<filas; i++)
    for(int j=0; j<columnas; j++)
      for(int k=0; k<profundidad; k++) {
        //Left-Front
        if(Math.pow(Site_R(), 2) < Math.pow(Left_Bond_R(), 2) +
           Math.pow(Front_Bond_R(), 2)) errors++;

        //Left-Behind
        if(Math.pow(Site_R(), 2) < Math.pow(Left_Bond_R(), 2) +
           Math.pow(Behind_Bond_R(), 2)) errors++;

        //Left-Up
        if(Math.pow(Site_R(), 2) < Math.pow(Left_Bond_R(), 2) +
        Math.pow(Up_Bond_R()(), 2)) errors++;
                  ...
}
```

Fig. 11. Code to count the geometric errors.

After having calculated the number of geometric errors it was noted that having sorted the Net for the correction of elementary errors, the geometric errors are minimal since, having relaxed the Network, there are few bonds that overlap each other.

For the correction of this type of errors, the Bond-Bond Relaxation algorithm was used, which was revised in Sub-Section II.E. In this algorithm the Bonds radius is decremented by a certain percentage (30%) while the geometric errors are modified until reaching the original radius. Since geometric errors were minimal, this method was not the most optimal, since instead of decreasing errors, it increased them in each iteration of the algorithm. So, it was decided to perform shits of the network.

This procedure is carried out through an iterative cycle until the geometric errors are equal to zero. The exchanges of Site with Site or Bond with Bond are made. The operation of the algorithm to fix the geometric errors consists of the following stages:

1. Two positions in the Network (i, j, k) and (x, y, z) are randomly selected.
2. A bond is randomly selected from the position (i, j, k).
3. The bond is exchanged with one of the positions (x, y, z).
4. If the geometric errors decrease, the change is

accepted and the next link is passed, otherwise they are returned to the original position.

5. If the geometric errors are zero, the algorithm is terminated, otherwise it returns to stage 1.

After presenting the algorithms to build the network, fix the elementary errors and geometric errors, we proceeded to test this first version and generate new versions in order to improve it. In the next section we show the tests and results that we perform to obtain different versions as well as the preliminary results of each version. We will finish the Experiments and Results Section showing the visual aspect of SIMGRAPH of the DSBM.

IV. Experiments and Results

As mentioned in Section III of this paper, several versions of SIMGRAPH of the DSBM were created. This in order to test them in different Java language development environments and streamline the process of building the network of a porous medium. In this Section we present what these versions were, as well as their similarities and their differences between them.

A. Version 1.0 of SIMGRAPH of the DSBM

This version follows the construction of the Porous Media Network as described in Section III.A. As well as counting and solving elementary and geometric errors as described in Sections III.D and III.E. Version 1 was developed in the IDE Eclipse

After having the complete program Version 1.0 of the SIMGRAPH of the DSBM, it underwent a series of tests to analyze its operation. The tests that were done were with different sizes of the Network registering the number of geometric errors and the time of the execution of each simulation, these records can be seen in Table I.

TABLE I: Testing of Version 1.0

Size	Geometric errors	Time
10	149	20 min.
15	529	34 min.
20	1330	3 hrs.
25	2258	6 hrs.
30	2948	8 hrs. 19 min.

As can be seen in Table I, the SIMGRAPH takes too long to complete its execution and the geometric errors are significant. This reason was why another solution was proposed for the construction of the Network, which will be described in the following Sub-Section.

B. Version 2.0 of SIMGRAPH of the DSBM

This version was developed with the Netbeans IDE. One of the main differences with respect to Version 1.0 is the creation of the Network. In this version instead of using a list of lists of lists, a dynamic cubic matrix of objects was used.

Another change in this version was the fix of geometric errors. In the previous version, two random links were chosen and exchanged, if the geometric errors diminished the change was accepted, otherwise they were returned to their original position.

The change in this method was that instead of choosing the two Bonds at random, two positions are chosen. It is checked if by exchanging the Bond of the first position with any of the Bonds of the second position, the geometric errors decrease, and the elementary errors do not increase but return to their original position.

In this version tests were also performed and although we observed a significant change in the execution time, it was not the most favorable version for the construction of the Network since, although the method of geometric error correction is much faster, it needed too many unnecessary iterations (there was no control of the solutions visited) to arrive with the solution. Table II shows the tests that were done with the same sizes of the Network of Table I, showing the execution times obtained when fixing the geometric errors.

TABLE II: Testing of Version 2.0

Size	Geometric errors	Time
10	4	2 seg.
15	15	21 seg.
20	13	42 seg.
25	23	16 min.
30	48	2 hrs. 19 min.

C. Final Version of SIMGRAPH of the DSBM

Version 2.0 was used as the basis for this last version. The modifications that were made were with respect to fix of the geometric errors. The approach that was established to fix these errors was in a specific way, that is, one by one solve the geometric errors without making unnecessary changes, something like what is established in a Backtracking Algorithm.

In this method to fix the geometric errors the relaxation of Sites was used (see Section III.E), also used in Version 2.0 of SIMGRAPH. Remembering, in this method the radius of the Sites is increased by a certain percentage, while the shifts are made in the network to correct the errors. The operation of this method consists of the following stages:

1. The Network is copied to an auxiliary network.
2. Geometric and elementary errors are counted.
3. Look for the node where the geometric error is located.
4. If the Geometric errors are greater than 0:
    a. The radii of the sites are increased by a certain percentage.
    b. The necessary changes are made to eliminate the error.
    c. If the geometrical errors have decreased and the elementary errors do not increase, the change is accepted.
    d. Geometric errors are counted again.

As can be seen from the previous steps, the correction is made in a specific manner, that is, knowing where the error is found, the changes are made to correct it. This method is much more optimal than the shifts of the network, since only the necessary changes are made to correct each of the errors and no unnecessary movements are made in the network. Table III shows the tests that were done with a network size of: 50, 100, 150, 200 and 250, showing the execution times

obtained when fixing the geometric errors.

TABLE III: TESTING OF FINAL VERSION

Size	Geometric errors	Time
50	1471989	6 sec..
100	11816529	13 sec.
150	39941942	43 sec.
200	94548640	1 min 9 sec.
250	184537971	5 min. 2 sec..

We can see that the results in Table III are very significant with respect to Versions 1.0 and 2.0. Tests were performed with larger sizes of the network, the maximum size was 250, from which to calculate the total number of entities is done by Equation 5.

$$\#entities = (\#Sites)^3 + 3 \times (\#Sites)^3 \qquad (4)$$

The $\#entries$ is the total number of entities and $\#Sites$ is the number of Sites in the neatwork. In the case of a network size of 250, we have the following number of entities in total.

$$\#entities = (250)^3 + 3 \times (250)^3$$

$$\#entities = 15625000 + 46875000 = 62500000$$

As we can see 62,500,000 entities is a considerable number of carries on the simulation for a traditional computer. However, we can see in Table III that the execution times for ordering them are not so high since we only have to wait more than 5 seconds. We will finish this Section showing the visual part of SIMGRAPH of the DSBM

D. Graphic Application SIMGRAPH of the DSBM

The Visual part of this Simulator, the window and each of its components, was created using the Java Swing graphic library. In this application the user is the one who enters the data of the network (Size, average of sites and average of links), who chooses the type of visualization (Graph of Distribution of Sites and Links, Site in 3D, Network in 2D and Network in 3D).

For the visualization of the Network in 2D and 3D the class Graphics was used, for the generation of the Graph of Distribution of Sites and Links the library JFreeChart was used and for the visualization of the Site in 3D it was made use of the library Java 3D. Fig. 12 shows the complete execution of this simulator with the different components that were previously described.

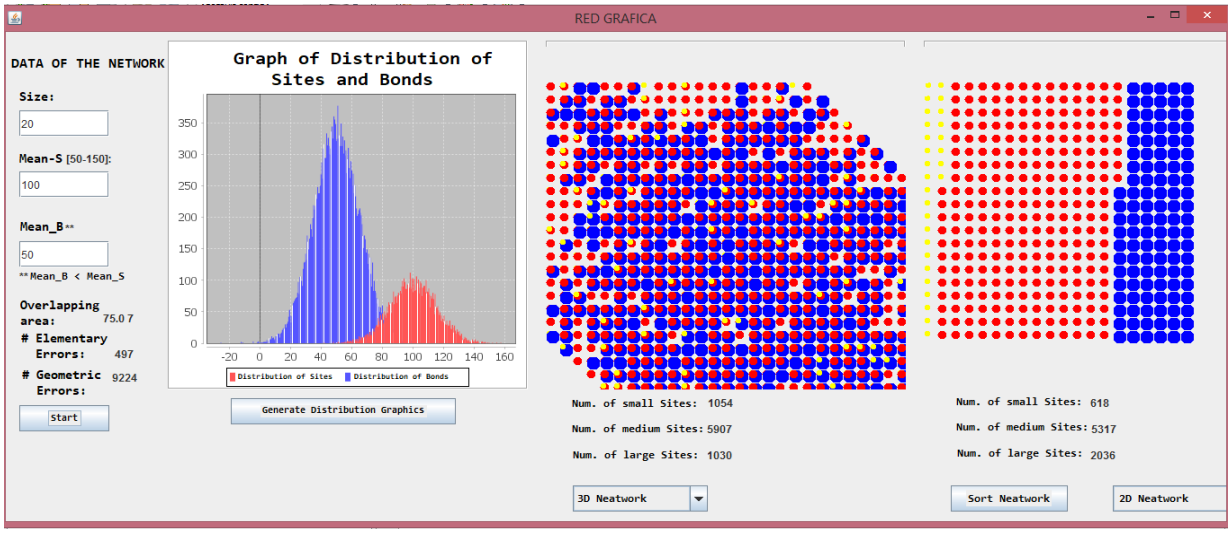

Fig. 12. Complete execution of SIMGRAPH of the DSBM.

V. CONCLUSION AND FUTURE WORK

In this paper, we present a Graphic Simulator for Dual Site - Bond Network Modeling for Porous Media through the Graphics class in Java.

Within the operating engine of SIMGRAPH, a different way of constructing the three-dimensional network of the porous medium was implemented, a different algorithm was also proposed to fix the elementary errors and finally a novel algorithm was proposed to fix the geometric errors.

For the part of the construction of the three-dimensional network of the porous medium, the simple fact of using Java completely changes the programming paradigm of the previous works, since the C language is always chosen because it is one of the most efficient. However, we see that Java also behaves in a good way and the fact of implementing the paradigm of object-oriented programming closer to the reality of the simulation of porous media.

In the algorithm proposed to fix elementary errors, it also has different characteristics than the traditional method of performing random shifts on the network of the porous medium. We considered that performing shifts was an incessant expense of computing resources, so we decided to order the entire network with one of the most efficient algorithms such as the Recursive MergeSort. In this way, the elementary errors in most cases were zero at the beginning of the construction of the Network, reducing the execution times of the traditional algorithms.

In the case of the algorithm specified to fix the geometric errors, it could be seen that it gave better results than all the traditional relaxation methods [9]. We use networks of size 250 with 62,500,000 entities, obtaining execution time of only 5 minutes with 2 seconds. The advantage of this algorithm is based on carrying out specific changes on the geometric errors of the network, this resembles a Backtracking Algorithm in which safe steps are taken in the search for a solution. The traditional relaxation methods generated random changes and could be circling blindly over the same search space, which made the algorithms take too long to fix all the geometric errors (order of hours to fix the

errors).

One of the things that seemed very strange to us and that we still follow in the search of a solution with foundation is the change between the IDE's of work, this case between Eclipse vs NetBeans. When we presented the results of Table II, we saw that despite being very similar algorithms between Version 1.0 and Version 2.0, the execution times decreased by 25% with Version 2.0. In general terms it is the same compiler for Java, but we assume that the administration of the RAM memory can be different between both IDE's.

Fig. 12 shows the main components of the SIMGRAPH, in which we can graphically manipulate the data for the generation of the network of the porous medium. We can also view network sites in 3D, the views of the network in 2D and 3D. One of the important components that we show in the SIMGRAPH is the graph of the distribution of Sites and Bonds, since with this graph we can see how much the network is concerned by the overlap of both distributions.

One of the main Future Works is to implement this simulation in a parallel programming environment. We have studied this detail because of the large number of elements that are managed in the networks, for example for a network of size 500, there would be 500 million entities. Trying to order all these entities in a sequential scheme is too complex a task. That is why it is proposed to migrate this version to a programming language of new generation such as Python, Pearl, Ruby or Elixir and thus implement a more robust parallel scheme that allows the simulation of this network size and even larger.

Finally, another Future Work that we visualize of this work is to carry out the main capillary processes of intrusion and retraction of mercury [5], that has been one of the main research topics in the last two decades. With this incorporation we could check in a more specific way the behavior that these networks are having that we are generating with SIMGRAPH of the DSBM

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Benjamín Moreno-Montiel carried out the initial approach of the case study, the writing of the paper, as well as the implementation of some SIMGRAPH modules and the analysis of the data obtained in this paper. Jacqueline Luna-Sánchez had the task of implementing some SIMGRAPH modules. Carlos-Hiram Moreno-Montiel and René MacKinney-Romero supported the writing of the article, as well as the analysis of the data presented in this paper; all authors had approved the final version.

References

[1] F. Lionetto and M. Frigione, "Mechanical and natural durability properties of wood treated with a novel organic preservative/consolidant product," *Materials & Design*, vol. 30, issue 8, pp. 3303-3307, 2009.

[2] V. Mayagoitia and I. Kornhauser; "Capillary processes in porous networks: 1. models of porous structures" in *Principles and Applications of Pore Structural Characterization*, J. M. Haynes, P. Rossi-Doria, J. W. Arrowsmith, Eds. Bristol, pp. 15-26, 1985.

[3] D. B. Pinniger and R. E. Child, "Woodworm-A necessary case for treatment? New techniques for the detection and control of furniture beetle," in *Proc. the Second International Conference on Urban Pests*, 1996.

[4] A. Dominguez, H. Pérez-Aguilar, F. Rojas, and I. Kornhauser; "Mixed wettability: A numerical study of the consequences of porous media morphology," *Colloid and Surfaces A: Physicochemical and Engineering Aspects*, pp. 187-188, vol. 415-424, 2001.

[5] C. H. Moreno, F. Rojas, G. Román, S. Cordero, M. A. Castro, and M. Aguilar, "A parallel simulator for mercury (Hg) porosimetry," in *Proc. ParSim 2009, EuroPVM/MPI*, C Trinities and M Schulz, Eds. Springer-Verlag LNCS series, Heidelberg 2009.

[6] F. Rojas, I. Kornhauser, C. Felipe, J. M. Esparza, S. Cordero, A. Dominguez, and J. L. Riccardo, "Capillary condensation in heterogeneous mesoporous networks consisting of variable connectivity and pore-size correlation," *Phys. Chem. Chem. Phys.*, vol. 4, pp. 2346-2355, 2002.

[7] P. M. Adler, "Porous media: Geometry and transports," *Butterworth-Heinemann*, 1992.

[8] V. Mayagoitia, F. Rojas, I. Kornhauser, and H. Pérez-Aguilar; "Modeling of porous media and surfaces structures: Their true essence as networks," *Langmuir*, vol. 13, no. 5, pp. 1327-1331, 1997.

[9] A. González-Méndez, G. Román-Alonso, F. Rojas-González, M. A. Castro-Garcia, M. Aguilar-Cornejo, and S. Cordero-Sanchez, "Constructionof porous networks subjected to geometric restrictions by using OpenMP," in *Proc. the 28th IEEE International Parallel & Distributed Processing Symposium Workshops*, 2014, pp. 1189-1197.

[10] S. Cordero, I. Kornhauser, C. Felipe, J. M. Esparza, F. Rojas, A. Domínguez, and J. L. Riccardo, "Topological analysis of Heterogeneous Three-Dimensional Porous Networks: The Case of variable connectivity and pore-size correlation," *Annales Universitatis Mariae Curie-Sklodowska, Lublin-Polonia*, vol. 56, no. 6, section AA, pp. 79-99, 2001.

[11] F. Rojas, I. Kornhauser, C. Felipe, and S. Cordero, "Everett's sorption hysteresis domain theory revisited from the point of view of the dual site-bond model of disordered media," *Journal of Molecular Catalysis A: Chemical*, vol. 167, pp. 141-155, 2001.

[12] M. J. Cruz, V. Mayagoitia, and F. Rojas; "Mechanistic studies of capillary processes in porous media. Part 2.- construction of porous networks by monte-carlo methods," *J. Chem. Soc., Faraday Trans. 1*, vol. 85, no. 8, pp. 2079-2086, 1989.

[13] A. Dominguez, S. Bories, and M. Prat, "Gas cluster growth by solute diffusion in porous media; experiments and automaton simulation on pore network," *Internacional Journal of Multiphase Flow*, vol. 26, pp. 1951-1979, 2000.

[14] V. Mayagoitia, A. Dominguez, and F. Rojas, "Twofold description of the morphology of colloid aggregates and other complex structures," *Journal of Non-Crystalline Solids*, vol. 147–148, pp. 183-188, 1992.

[15] G. B. Kuznetsova, V. Mayagoitia, and I. Kornhauser, "Twofold description of the morphology of polymers," *Intern. J. Polymeric Mater.*, vol. 19, pp. 19-28, 1993.

[16] J. Salmones, E. Garciafigueroa, V. Mayagoitia, F. Rojas, and I. Kornhauser, "Twofold description of the texture of aluminium oxides prepared by the sol-gel method," *Adsorption Science & Technology*, vo. 15, no. 9, pp. 661-675, 1997.

[17] V. Mayagoitia, F. Rojas, and I. Kornhauser, "Twofold description of porous media and surface structures: A unified approach to understand heterogeneity effects in adsorption and catalysis," *Langmuir*, vol. 9, no. 10, pp. 2748-2754, 1993.

[18] T. J. Barton, L. M. Bull, W. G. Klemperer, D. A. Loy, B. McEnaney, M. Misono, P. A. Monson, G. Pez, G. W. Scherer, J. C. Vartuli, and O. M. Yaghi, "Tailored porous materials," *Chem. Mater*, vol. 11, pp. 2633-2656, 1999.

[19] D. Zhao, J. Feng, Q. Huo, N. Melosh, G. H. Fredrickson, B. F. Chmelka, and G. D. Stucky, "Triblock copolymer syntheses of mesoporous silica with periodic 50 to 300 åpores," *Science*, vol. 279, pp. 548-552, 1998.

Benjamıin Moreno-Montiel received the BSc and MSc and PhD degrees in science and technologies of information from Universidad Autónoma Metropolitana –Unidad Iztapalapa, Mexico City, México, in 2007 2009 and 2017, respectively. He has been a professor of computer sciences since 2015 in Universidad Autónoma Metropolitana where he took his undergraduate courses in computer sciences. He teaches artificial intelligence and machine learning for undergrads and postgrads and has some publications in international conferences and journals.

Carlos-Hiram Moreno-Montiel received the BSc and the MSc degrees in science and technologies of information from Universidad Autónoma Metropolitana - Unidad Iztapalapa, Mexico City, México, in 2006 and 2010, respectively. He is currently working towards the PhD degree at the Posgrado en Ciencias y Tecnologías de la Información in Universidad Autónoma Metropolitana – Unidad Iztapalapa, México City, México. His research interests are parallel computing, software engineering, and artificial intelligence.

Jacqueline Luna-Sánchez she is currently working towards the BSc degree at the Licenciatura en Computación in Universidad Autónoma Metropolitana – Unidad Iztapalapa, Mexico City, México. His research interests are parallel computing, software engineering, and artificial intelligence.

René MacKinney-Romero received the BSc degree from Universidad Autónoma Metropolitana – Unidad Iztapalapa, Mexico City, Mexico, in 1993, the MSc degree in computation from the University of Oxford, England, in 1994, and the doctorate degree in computer sciences from the University of Bristol, England, in 2002. He has been a professor of computer sciences since 1990 in Universidad Autónoma Met-ropolitana where he took his undergraduate courses in computer sciences. He teaches artificial intelligence and machine learning for undergrads and postgrads and has many publications in international conferences and journals.