

# An algorithm to construct conformable tubular networks that occupy arbitrary regions in $\mathbb{R}^3$

W. Jiang<sup>a</sup>, F. Ghasempour<sup>a</sup>, F. Mendivil<sup>b</sup>, S.D. Peterson<sup>c</sup>, E.R. Vrscaj<sup>a,\*</sup>

<sup>a</sup> Department of Applied Mathematics, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

<sup>b</sup> Department of Mathematics and Statistics, Acadia University, Wolfville, Nova Scotia B4P 2R6, Canada

<sup>c</sup> Department of Mechanics and Mechatronics Engineering, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

## ARTICLE INFO

### Article history:

Received 9 August 2017

Revised 28 March 2018

Accepted 25 April 2018

Available online 6 June 2018

### Keywords:

Tubular networks

Circle packing algorithms

Connection algorithms

Depth-first search algorithm

## ABSTRACT

We describe a general algorithm for the construction of networks of tubes which occupy a specified, and possibly quite complicated, region  $D \subset \mathbb{R}^3$  as fully as possible. Such networks can be important in a variety of industrial applications ranging from heat exchangers to storage vessels. The algorithm is comprised of three fundamental steps: (i) *decomposition* of region  $D$  into subregions  $D_i$ , (ii) *packing* of each subregion with a set of parallel tubes using appropriate circle-packing algorithms and finally, (iii) *connection* of the tubes, both within each subregion  $D_i$  as well as between neighboring regions. Such a design problem is extremely ill-posed: For a given region  $D$ , our algorithm can produce an enormous number of networks, which generally requires that a much smaller number of “best” networks be isolated. Such isolation can be accomplished by ranking the networks based on some application-dependent properties. Some simple examples are presented to illustrate the procedure and output of the algorithm.

© 2018 Elsevier Inc. All rights reserved.

## 1. Introduction

In this paper we describe a general algorithm for the construction of networks of tubes which occupy a specified, and possibly quite complicated, region  $D \subset \mathbb{R}^3$  as fully as possible. Such “conformable tubular networks” (CTNs) can be important in a variety of industrial applications ranging from heat exchangers to fluid distribution networks.

Our algorithm is comprised of three fundamental steps: (i) the *analysis and decomposition* of the three-dimensional region  $D$  into subregions  $D_i$ , (ii) the *packing* of each subregion  $D_i$  with an assembly of tubes and (iii), the *connection* of the tubes, both within each subregion  $D_i$  as well as between neighboring subregions.

In some applications, it may be necessary to check that there are no “dead ends.” This may be viewed as a fourth step in our algorithm. Finally, this design problem is extremely ill-posed: For a given region  $D$ , our algorithm can produce an enormous number of networks, in which case some “best” networks must be selected. This can be accomplished by ranking the networks based on some problems relevant to the application, e.g., flow characteristics.

\* Corresponding author.

E-mail addresses: [ervrscaj@uwaterloo.ca](mailto:ervrscaj@uwaterloo.ca), [p.muthukumar@elsevier.com](mailto:p.muthukumar@elsevier.com) (E.R. Vrscaj).

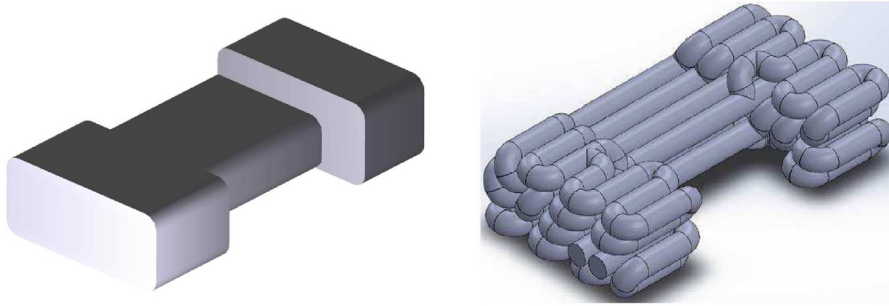


Fig. 1. A “saddle-bag” region  $D$  (left) and a “28-8-28” tubular network produced which occupies the region.

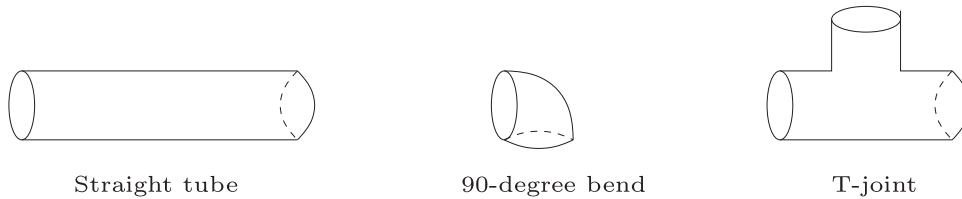


Fig. 2. The three fundamental elements employed in the construction of tubular networks in this paper.

### 1.1. A simple, motivating example: the “barbeque pool heater”

One very simple example which provided a motivation for our early constructions is the so-called “barbeque (BBQ) pool heater” [1]. Water from a pool is pumped into a tubular network situated inside a barbeque via an inlet pipe. As the water travels through the network, it is heated by the flames in the barbeque. The warmer water exits the network through an outlet pipe and re-enters the pool.

Now consider the following simplified pool heater problem. Firstly, we employ tubes of a fixed (outer) radius  $R > 0$ . Secondly, we consider only a special class  $\mathcal{C}$  of one inlet-one outlet tubular networks which occupy the interior region  $D$  of the barbeque. These networks are formed by stacking the tubes in a parallel fashion so that they extend from one side of the barbeque to the other. Neighboring tubes are then connected at both ends by means of connections that are described in Section 2.1. The problem – in this case a combinatorial one – is to generate all possible one inlet-one outlet networks. (In reality, the parallel tubes in the heater should not be touching each other since each one of them should be heated directly by the hot air in the barbeque. We can simply replace each tube (and connection) of radius  $R$  by a tube of radius  $R' < R$  to produce the necessary spacing.)

If only “U-joint” or endcap connections are employed at the ends of the tubes, a particular subset of solutions is obtained, namely *serpentine* tubular networks. These networks consist of a single, unbranched path from the inlet to the outlet. The use of “T-joints” to produce merges (Section 2.1) results in branched networks. In general, a large number of networks are produced. The next step would be to determine which network/networks are “optimal” for the BBQ heater problem, e.g., maximum temperature increase for a given flow rate. Such a discussion, however, is beyond the scope of this paper.

In [2] and [3], we reported on the construction of tubular networks for some slightly more complicated “saddle-bag” regions  $D$ , where two larger rectangular regions – the “bags” – are connected by a narrower rectangular channel.

Fig. 1 shows a particular example of a saddle-bag region (left) along with one of the many one inlet-one outlet networks produced by our algorithm (right). The network shown is a 28-8-28 saddle-bag network, where two side bags, each of which contains 28 tubes, are connected to a middle region containing 8 tubes. In this particular network, the inlet and outlet are situated on the same side of the network (in fact, they are next to each other). In many networks produced by our algorithm, the inlet and outlet are on opposite sides.

## 2. Basic ingredients of algorithm

Currently, we have employed the following three *fundamental elements* to construct tubular networks: (i) straight tubes, (ii) 90-degree bends and (iii) T-joints (or “Tees”). These three elements are sketched in Fig. 2. Note that the sketches in Fig. 2 suggest that the diameters of all tubes are the same. This is not a fundamental restriction of the algorithm, although accommodating multiple tube sizes will require additional elements. Using these elements, we currently employ four basic operations to construct our tubular networks, shown schematically in Fig. 3. Note that the spacing between neighboring tubes as well as connections has been exaggerated somewhat in order to show the bends and T-joints more clearly.

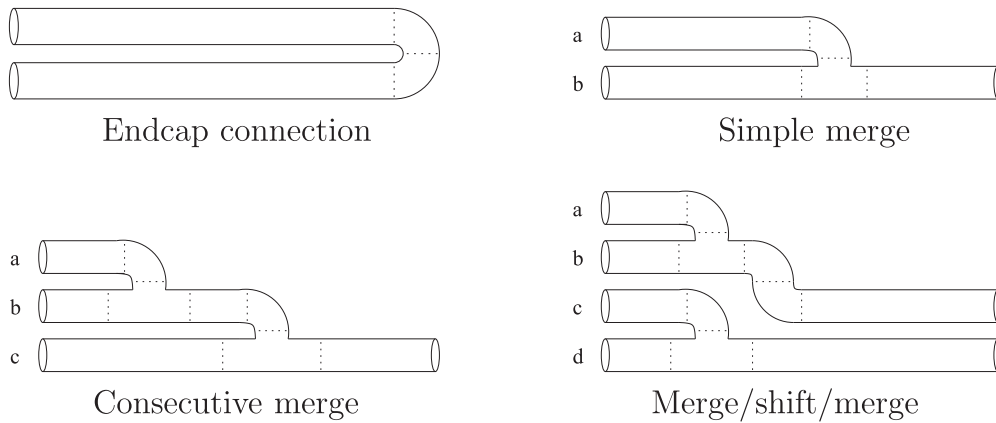


Fig. 3. The four basic connection schemes used to construct the tubular networks.

- *Endcap connection*: A 180-degree bend, formed with two 90-degree bends, which connects two neighboring tubes at their ends. It is used at the extreme ends of the network as well as at the ends of each block  $B_i$ .
- *Simple merge*: Tube  $a$  flows into a neighboring tube  $b$ . The merge is produced with a 90-degree bend and a T-joint. This operation introduces branching in the tubular network. It can be used only at an interface of two neighboring blocks  $B_i$ , producing a 2-to-1 reduction in the number of tubes from one block to the next.
- *Consecutive merge*: Tube  $a$  flows into neighboring tube  $b$ , which then flows into neighboring tube  $c$ . Used only at an interface of two neighboring blocks  $B_i$ , producing a 3-to-1 reduction in the number of tubes. Each of the two merges is produced with a 90-degree bend and a T-joint.
- *Merge/shift/merge operation*: As shown below, tube  $a$  flows into neighboring tube  $b$ . As well, tube  $c$  flows into tube  $d$ . Each of the two merges is once again produced with a 90-degree bend and a T-joint. Tube  $b$  is then shifted to the former position of  $c$ . The shift is formed with two 90-degree bends. Used only at interface of two neighboring blocks  $B_i$ , producing a 4-to-2 reduction in the number of tubes.

The above list of connection schemes is by no means complete. Consecutive merges with higher than a 3-to-1 reduction, as well as merge/shift/merge schemes with higher reduction ratios can be considered if necessary. Additional elements and combinations can also be introduced to accommodate specific geometries, e.g., non-90-degree bends, contractions/expansions, curved tubes, etc.

### 3. First step: analysis and decomposition of region $D \subset \mathbb{R}^3$ to be occupied by the network

Clearly, the simplest situation is when the 3D region  $D$  is a long block with constant cross section (e.g., rectangular, triangular), in which case the tubes can be packed in a parallel manner and connected to each other at the ends. The next simplest case, a variation of the above, is when  $D$  exhibits some kind of longitudinal symmetry, i.e., there exists a principal (but not necessarily straight) axis along which the cross section may vary. In this case, the construction problem is quasi-two-dimensional and the problem of packing tubes in three dimensions is reduced to circle-packing in two dimensions. A decomposition of  $D$  may then be performed by segmentation, i.e., by approximating it as well as possible by a union of blocks  $B_i$  with constant cross sections. A sketch of such a generic decomposition is shown in Fig. 4. Each block  $B_i$  is then packed with a parallel set of tubes as shown schematically in the figure. The saddle-bag region shown in Fig. 1 possesses this longitudinal symmetry and is composed of three blocks. In more complicated situations, it may be necessary to employ a method which can extract the fundamental morphological characteristics of region  $D$ , namely, its shape and structure. For example, if  $D$  possesses a dendritic structure in which several subprincipal axes emanate from a principal location or axis, it is desirable to determine these axes. Such information will be important for both the segmentation/discretization of the region as well as the determination of packing directions.

One way of characterizing principal and/or subprincipal axes of a complicated region is in terms of its “skeleton lines,” which are produced by a method generally known as “skeletonization.” A number of different methods of performing skeletonization currently exist – see, for example, [4–6]. A discussion of these methods is beyond the scope of this paper.

### 4. Second step: packing the subregions with tubes

As mentioned above, each block  $B_i$  comprising the envelope volume is assumed to contain a set of tubes arranged in a parallel fashion. The problem of packing each block  $B_i$  is therefore reduced to the two-dimensional problem of packing circles into its (constant) cross-sectional region. In the simple example presented in Section 1, i.e., the 28-8-28 saddle-bag network, a very simple hexagonal packing of tubes of equal diameters was employed. In general, however, the cross-sectional

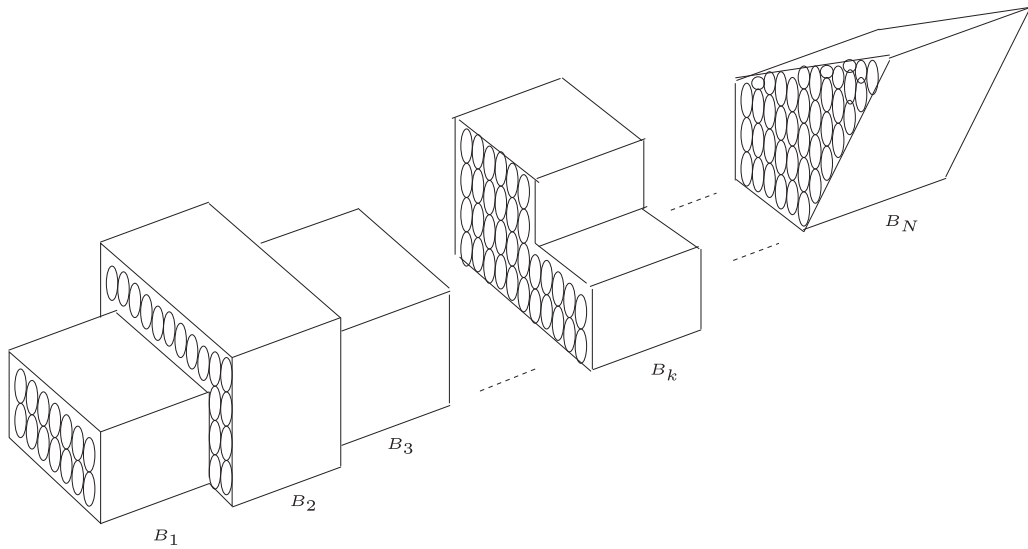


Fig. 4. Generic structure of tubular network exhibiting variation along a principal direction.

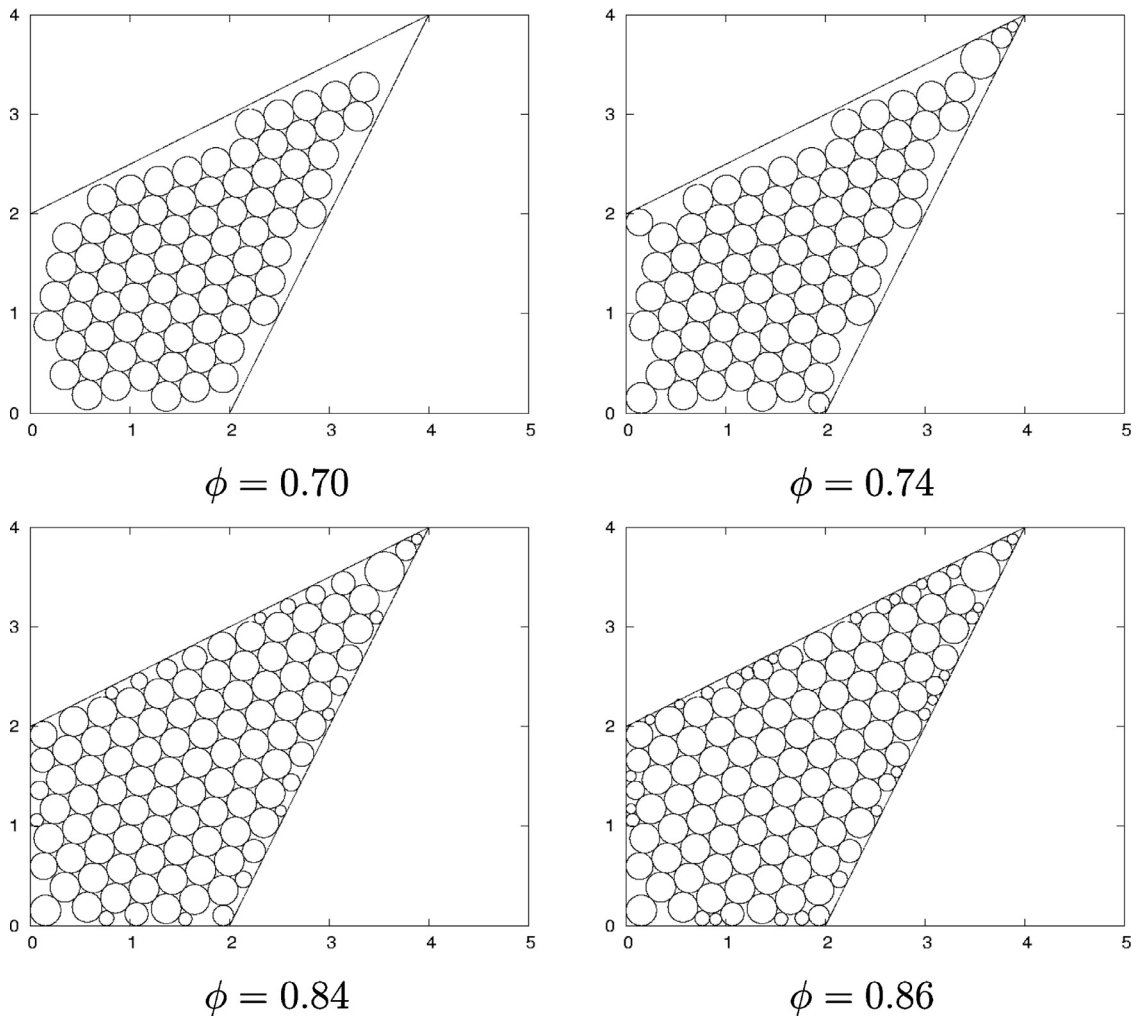
regions are more complicated, i.e., polygonal. There is also the requirement of packing these regions as efficiently as possible which will most probably require tubes/circles with different sizes.

Circle-packing has received a great deal of attention in the literature from a variety of perspectives. The starting point of our work was the algorithm by George et al. [7], hereby to be referred to as the GGL algorithm. One of the original motivations of the GGL paper was the efficient packing of pipes or bottles in a rectangular box. Proceeding in a rather intuitive and pragmatic manner, the packing starts at the two bottom corners and proceeds upwards by using two basic packing schemes: (i) packing a new circle so that it touches an already-packed circle and a side of the box (“side packing”), (ii) packing a new circle so that it touches two already-packed circles (“two-circle packing”). Only packings that (a) do not overlap already-packed circles and (b) do not extend outside the box are acceptable. The reader is referred to [7] for details and technicalities associated with the original GGL packing scheme.

Since the aims of our tubular network problem differ from those that led to the GGL algorithm, a number of differences between our algorithms and the GGL algorithm naturally arose. First, unlike the GGL algorithm, our problems do not necessarily have a preferred direction (i.e., the “bottom” of a box), nor are our regions necessarily rectangular boxes. We therefore adapted the GGL scheme to pack circles into arbitrary polygonal regions, starting at any of the corners and proceeding inward. Depending on the application, however, it may be more desirable to have the tubes packed in the interior, only touching the boundaries when necessary. This led to a number of “reversed-GGL” schemes in which the packing started in the interior of a polygonal region and proceeded outward toward the boundary. With an eye toward particular applications, some of these schemes packed the largest circles (corresponding to the largest tubes) in the interior, with smaller circles being employed as the packing proceeded outward toward the boundaries. As well, a number of “jiggling” methods, in which a given packing scheme was perturbed for the purpose of obtaining more available space for packing, were devised. (The original GGL paper [7] also employed some “shaking” methods.) A detailed discussion of all of these early packing methods developed for the tubular network problem is given in [8].

In this paper, we outline an improved packing scheme in which the sizes of circles are, after a first stage, no longer prescribed but adjusted to fit spaces that are available for packing. The packing of a bounded region  $D \subset \mathbb{R}^2$  with polygonal boundary curve  $C$  is accomplished using four fundamental steps:

- **Step 1:** Perform a hexagonal (or rectangular, if desired) packing of the interior region of  $D$  with circles of a prescribed radius  $R$ , starting at a reference point  $(\bar{x}, \bar{y})$  from which a set of (hexagonal) lattice points,  $(x_k, y_k)$ , separated by distance  $2R$  and oriented at an angle  $\theta$  to the “lab frame” horizontal axis, can be generated. (This method is essentially the first step of a “reversed-GGL” scheme.) Identify all “boundary circles,” i.e., the outermost layer of Step 1 circles which lie closest to the boundary  $C$ .
- **Step 2 “Corner packing”:** It is possible that there are significant unpacked regions near the corners of the polygon  $C$ . There are a number of possible strategies to pack circles into these regions. One method is to find the packed Step 1 circle  $C_p$  closest to a convex vertex  $P$  and then construct, if possible, the largest circle  $C$  which touches both boundary segments emanating from  $P$  as well as  $C_p$ . (This can be done analytically – we omit a presentation of the formulas.)
- **Step 3:** Use the boundary circles from Step 1 and the two-circle packing method from the GGL scheme to pack a second layer of smaller boundary circles which touch the boundary circles as well as the boundary  $C$ . (Once again, this can be done analytically and we omit a presentation of the formulas.)



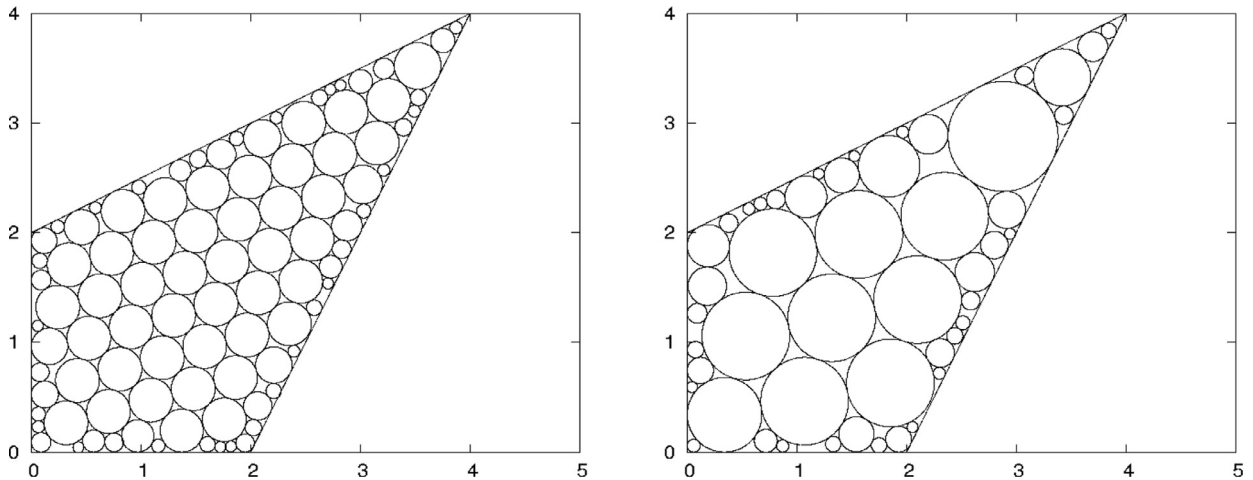
**Fig. 5.** Results obtained by applying Steps 1–4 successively to the simple trapezoidal region in Fig. 4. The packing fraction  $\phi$  obtained after each step is presented under each figure. **Top left:** Step 1 hexagonal packing with circles of radius  $R = 0.15$ , starting at centroid. Orientation angle  $\theta = \frac{\pi}{10}$  radian. 79 circles packed.  $\phi = 0.70$ . **Top right:** Step 2 corner packing performed on the result of Step 1. After four iterations, 6 circles are added.  $\phi = 0.74$ . **Bottom left:** Step 3 packing. 26 circles are added.  $\phi = 0.84$ . (Note the significant increase in packing fraction.) **Bottom right:** Step 4 packing. 15 circles are added.  $\phi = 0.86$ . In Steps 2–4, no circles with radii less than 0.05 were packed. Rectangular region shown in above figures:  $0 \leq x \leq 5$ ,  $0 \leq y \leq 4$ .

- **Step 4:** Pack a third layer of boundary circles which touch the boundary segments, using boundary circles from Steps 1 and 2 (and which are not necessarily touching) and the two-circle packing scheme. This packing is performed numerically: Circles which touch two packed circles are “grown” from a minimum radius  $R_{\min}$  (see below) in discrete increments  $\Delta r$ . The growth is stopped before the circles extend beyond the boundary. If desired, this procedure may be iterated since it is possible that some vacant spots are missed during the first application.
- **Minimum feasible radius  $R_{\min}$ :** For reasons of practicality (e.g., limits in construction), it may be desirable to prescribe a minimum radius  $R_{\min}$  so that no circles with radii smaller than  $R_{\min}$  are accepted in Steps 2–4.

To illustrate this scheme, the packings of a simple trapezoidal region  $D$  in the plane obtained by a successive application of these steps are shown in Fig. 5. A minimum feasible radius  $R_{\min} = 0.05$  (see below) was employed in Steps 2–4. The packing efficiencies  $\phi$  (fraction of total area covered by packed circles) obtained after each step are also presented below each figure.

In an effort to find the best possible packing of a region  $D$ , one may wish to optimize over ranges of the Step 1 parameters:

- For a fixed  $R$ -value, Steps 1–4 can be performed for a variety of orientation angles  $\theta$  as well as starting points  $(\bar{x}, \bar{y})$ . It is sufficient to consider translations of the starting point within the basic (horizontally-oriented) parallelogram unit cell



**Fig. 6.** Results of optimized packing scheme applied to the simple trapezoidal region in Fig. 4. **Left:**  $R = 0.2$ . 96 circles employed in packing.  $\phi = 0.88$ . **Right:**  $R = 0.4$ . 49 circles employed in packing.  $\phi = 0.88$ .

with sides of length  $2R$  and lower left vertex at  $(\bar{x}, \bar{y})$ . For hexagonal packing it is also sufficient to consider orientation angles  $0 \leq \theta \leq \pi/3$ .

- It may also be desirable to perform the above optimization over a range of Step 1  $R$ -values.

In Fig. 6 are shown the results of the first optimization procedure as applied to the trapezoidal region  $D$  of Fig. 5, using  $R = 0.2$  and  $R = 0.4$ . In each case, the starting reference point  $(\bar{x}, \bar{y})$  was the centroid of  $D$ . Each result represents the best packing (in terms of packing efficiency) obtained from 1000 possible reference points. These were obtained by considering 100 possible translations of the centroid (obtained from a grid of  $10 \times 10 = 100$  points within the unit cell) and 10 orientation angles  $\theta$  between 0 and  $\pi/3$ . In both cases, a minimum radius of  $R_{\min} = 0.05$  was employed. For both  $R$  values, a packing efficiency of  $\phi = 0.88$  was obtained, suggesting that after the Step 1 circles of radius  $R$  are packed, the algorithm performs quite well in packing the remaining unpacked region as efficiently as possible in Steps 2–4. Indeed, packing efficiencies of  $\phi = 0.88$  to  $0.90$  were obtained for a range of  $R$  values from  $0.1$  to  $0.9$ .

As we discuss in the next section, in order to facilitate connections of tubes in adjacent cross-sections it is advantageous to look for an interior (planar) region  $\mathcal{R}$  that is common to several, if not all, adjacent block cross-sections. These regions are packed first. In this way, it is possible that some tubes will extend continuously from one block into another possibly through all blocks comprising the network. Such tubes are clearly visible in the 28-8-28 network shown in Fig. 1. (There are 8 of these tubes in total.) Each non-common region is then packed. The packing of each such region begins with circles that touch already-packed circles from the common region. The packing then proceeds outward to the boundaries.

### 5. Third step: connecting the tubes to produce tubular networks

With reference to the general network shown in Fig. 4, after all of the blocks  $B_i$  have been packed with tubes (including tubes that extend continuously from one block to the next, possibly from one end of the network to the other), it remains to connect them to produce a feasible tubular network with a specified number of inlets and outlets. As will be shown below, there are, in general, a significant number of possible connections at each end and each block-to-neighboring-block interface which can result in an enormous number of possible networks.

In general, two neighboring blocks  $B_i$  and  $B_{i+1}$  will have a common area, to be denoted as region  $\mathcal{R}_i$ , as sketched in Fig. 7. Circles in this region represent tubes that can run continuously from  $B_i$  to  $B_{i+1}$ . Tubes that are in block  $B_i$  but not completely in region  $\mathcal{R}_i$  clearly cannot be continued into block  $B_{i+1}$ . As such, they must be either (1) turned back into block  $B_i$  via an endcap connection or (2) merged into a neighboring tube which lies in  $\mathcal{R}_i$ . There may also be the possibility of shifting a tube into a neighboring position which extends into region  $\mathcal{R}_i$  by means of a merge/shift/merge operation. The same comments apply to tubes in block  $B_{i+1}$  which cannot run into block  $B_i$ .

The 28-8-28 saddle-bag network shown in Fig. 1 illustrates these connection schemes. The bags of the saddle-bag network,  $B_1$  and  $B_3$ , contain 28 tubes whereas the interior connector region  $B_2$  contains only 8 tubes. As such, 8 tubes run from one end, i.e.,  $B_1$ , through the connector region  $B_2$ , to the other end  $B_3$ . The other tubes 20 tubes in  $B_1$  and  $B_3$ , however, must remain in their respective bags.

#### 5.1. An illustrative example: determining a particular 9-4-9 network solution “by hand”

Because of the complexity of the problem, in particular the one of connecting tubes at block interfaces, it is instructive to work out a relatively simple example involving a small number of tubes. Here, we consider the 9-4-9 saddle-bag network



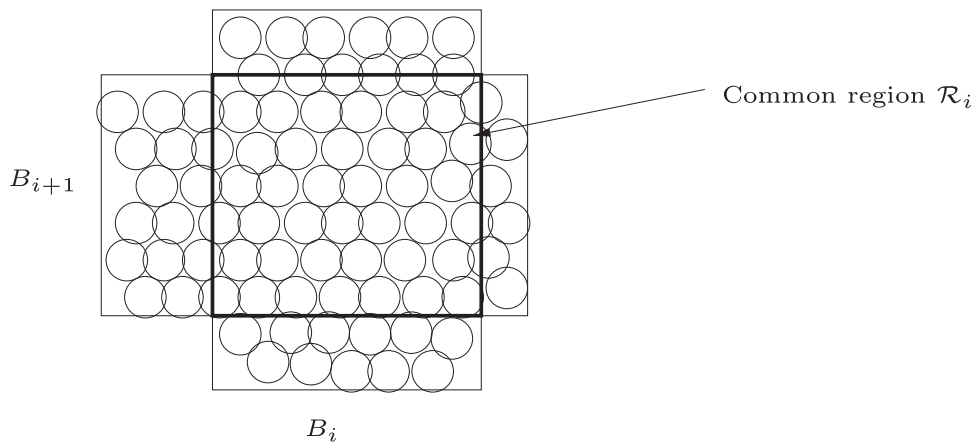


Fig. 7. Two packed neighbouring blocks  $B_i$  and  $B_{i+1}$  with common region  $\mathcal{R}_i$ .

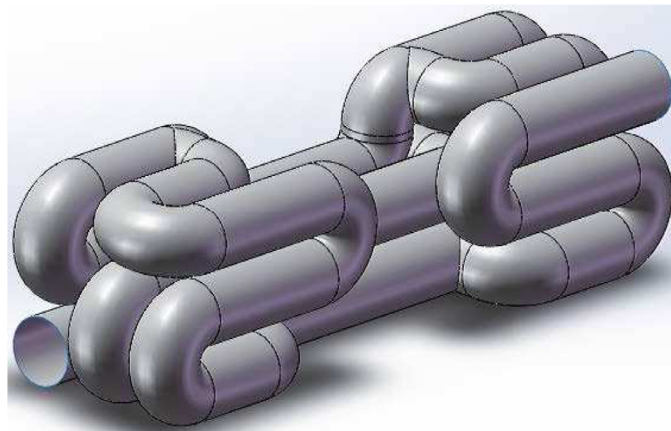


Fig. 8. A 9-4-9 network produced by the algorithm outlined in this paper.

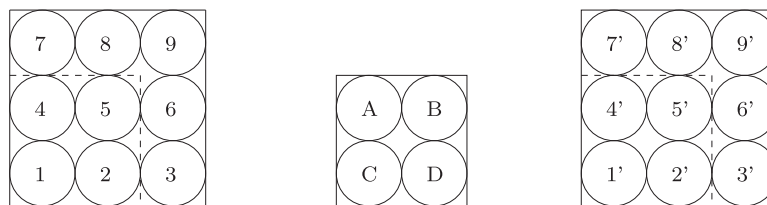


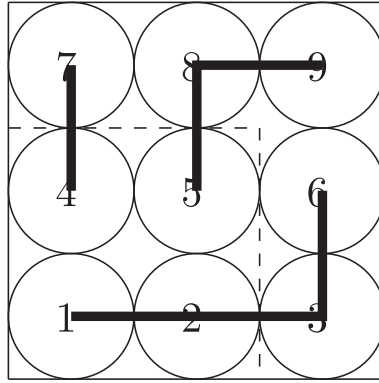
Fig. 9. Cross sections of 9-4-9 saddle-bag network. **Left:**  $B_1$  saddle bag “L” with 9 tubes. **Middle:** Connector region  $B_2$  with 4 tubes. **Right:**  $B_3$  saddle bag “R” with 9 tubes.

problem. One such network is shown in Fig. 8. The cross-sectional regions of the two bags,  $B_1$  and  $B_3$ , as well as the connector region  $B_2$  of a generic 9-4-9 network are shown in Fig. 9. Note that the tubes are packed in a rectangular fashion. All merge, shift and endcap connections are made between nearest rectangular neighbours.

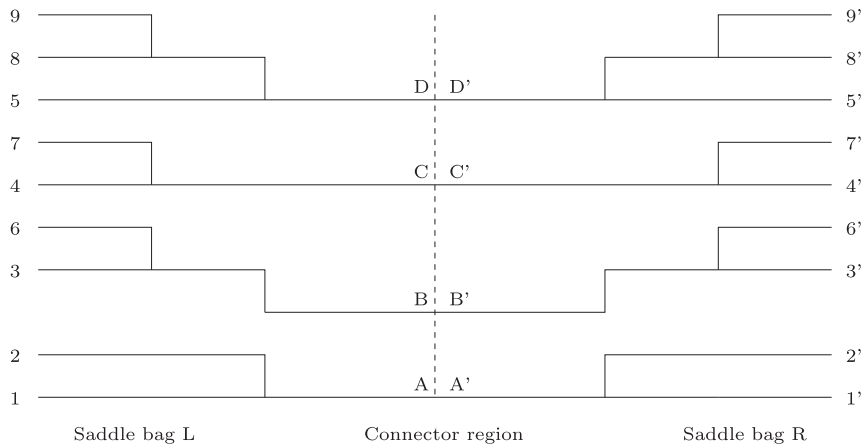
#### Interblock connections:

Using the ideas described at the beginning of this section, we present one solution for the connections at the  $B_1 - B_2$  and  $B_2 - B_3$  interfaces. We start at side L and determine how the nine tubes in the saddle bag  $B_1$  will become Tubes A,B,C and D in the connector region  $B_2$ .

- Tube A: Tube 7 merges into Tube 4, which then becomes Tube A. (This implies that Tube 4 is not available for any other operation.)
- Tube B: Tube 9 merges into Tube 8. Tube 8 then merges into Tube 5, which then becomes Tube B. (This implies that Tube 5 is not available for any other operation.)
- Tubes 6 and 3 must still be “removed.” This means that Tube 2 will have to be moved.
- Tube C: Tube 2 merges into Tube 1, which then becomes Tube C.



**Fig. 10.** Diagrammatic representation of the resulting set of connections produced by the procedure outlined above.



**Fig. 11.** Graphical representation of shift/merge connections between saddle bags and connector region.

- Tube D: Tube 6 merges into Tube 3, and then Tube 3 shifts to Tube 2 (after Tube 2 has merged to Tube 1).

The result of the above series of steps is a particular *merge-shift configuration* for the 9-4-9 saddle-bag network which is conveniently represented by the diagram shown in Fig. 10.

Of course, similar shifting and merging operations must be performed on Tubes 3', 6', 7', 8' and 9' in order to move from saddle bag R into the connection region. For simplicity, we consider the same merging and shifting operations on saddle bag R as was done for saddle bag L in the steps above. Notationally, we simply replace all tube numbers with primes, i.e., 3 with 3', etc.. The resulting, almost-finished network may be represented by the diagram in Fig. 11.

*Endcap connections:*

The final step is to make appropriate endcap connections between neighboring tubes at both ends of the network produced in the previous step. A complete endcap connection scheme for the 9 tubes at the end of a 9-4-9 network will consist of four endcap connections between neighboring tubes, leaving one tube unconnected. The unconnected tube will serve as either an inlet or outlet for the resulting 9-4-9 network.

For the moment, we simply state that associated with the merge-shift scheme determined in the previous section and shown in Fig. 10, there are five feasible endcap connection schemes. These five schemes are presented in Fig. 12. More details on the determination of these schemes – as well as why other schemes are unfeasible – will be given in the next section. If we now use the fourth and first connection schemes of Fig. 12 on, respectively, the left (L) and right (R) ends of the unfinished saddle bag network represented in Fig. 11, we obtain the finished saddle-bag network shown in Fig. 13.

## 5.2. Brief description of the general connection algorithm

### 5.2.1. Connection schemes at block interfaces

The problem of connecting tubes at block interfaces is more complicated than that of connecting them at the extreme ends of the network. As such, we first consider the interface problem, with reference to Fig. 7. For simplicity, our discussion below is restricted to the circles (which represent tubes) that are in Block  $B_i$  of the figure. Firstly, circles of  $B_i$  that lie entirely in the common region  $\mathcal{R}_i$  are referred to as *common circles*. The remaining circles of  $B_i$  are referred to as



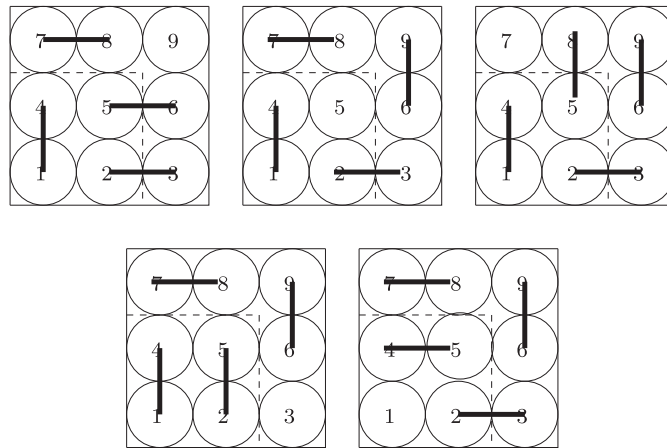


Fig. 12. The five feasible endcap connection schemes associated with the merge-shift connection scheme shown in Fig. 10.

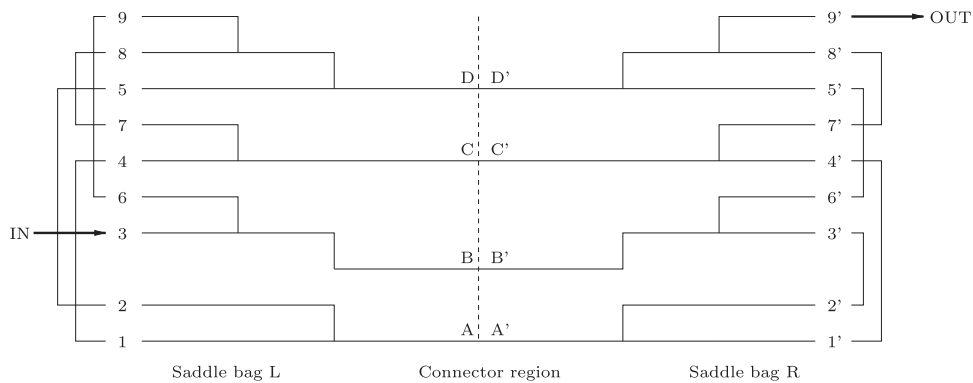


Fig. 13. A finished 9-4-9 network formed by endcapping the unfinished network of Fig. 11 with two feasible sets of endcap connections.

boundary circles (BCs). It is the boundary circles which must be “processed,” i.e., endcapped, merged or shifted so that they disappear when  $B_i$  is exited. The farther a circle lies from the common region, the higher the priority that it be considered for connection/shifting. This is accomplished by assigning a “layer-rank” index  $L_r$  to each circle, as follows:

- First-layer BCs, also called 1B circles, touch common circles.  $L_r = 1$ .
- All other BCs are called 2B circles,  $L_r = 0$ .
- First-layer common circles,  $L_r = 2$ , or 1C circles, touch at least one BC.
- Second-layer common circles,  $L_r = 3$ , or 2C circles, touch at least one 1C circle.
- All other common circles are not involved in this algorithm since it is not possible for them to be connected at an interface using any of the merge operations shown in Fig. 3.  $L_r = \infty$ .

Circles are then processed in increasing order of layer rank index  $L_r$ .

There is another important factor. Since it is necessary that all BCs be “processed” and not ignored in any step, it is advisable to consider those BCs with a lower number of available connections first. As such, we define the *connectivity degree*,  $CD$ , of a BC as the number of (neighboring) circles to which it may be connected.

Finally, it is convenient to define the *connectivity value*  $CV$  of a circle/tube, the maximum number of tubes to which it can be connected in one of the four operations shown in Fig. 3:

- A 1B tube can be the middle of a consecutive merge (tube  $b$ , connected to both tubes  $a$  and  $c$ ). A 1C tube can be at the end of a consecutive merge or a merge/shift/merge (tube  $c$ , connected to tubes  $a$  and  $b$  in both cases).  $CV = 2$ .
- 2B or 2C tubes can be connected to only one tube.  $CV = 1$ .

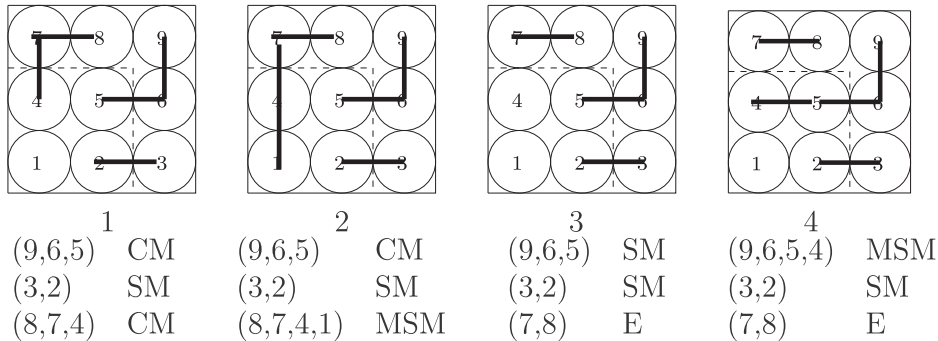
The indices associated with each tube, along with the set of tubes to which each tube can be connected, comprise an “availability list”. For example, the availability list for interface between Saddle Bag L and the connector region of the 9-4-9 network shown in Fig. 9 is presented in Table 1.

Starting with such an availability list, our algorithm examines all possible connections at an interface in a systematic manner. Given the different natures of the circles/tubes involved (i.e., BCs vs. common circles) as well as the connection

**Table 1**

Availability list for the 9-4-9 network Saddle bag L/connector interface shown in Fig. 9.

Tube	$L_r$	CV	Available tubes	CD
1	3 (2C)	1	2,4	2
2	2 (1C)	2	1,3,5	3
3	1 (1B)	2	2,6	2
4	2 (1C)	2	1,5,7	3
5	2 (1C)	2	2,4,6,7	4
6	1 (1B)	2	3,5,9	3
7	1 (1B)	2	4,8	2
8	1 (1B)	2	5,7,9	3
9	0 (2B)	1	6,8	2

**Fig. 14.** Four of the 22 distinct connection schemes for a 9-to-4 (or 4-to-9) block transition produced by the depth-first search algorithm. The connections comprising each scheme are shown below it. CM: consecutive merge, SM: simple merge, MSM: merge/shift/merge, E: endcap.

operations in Fig. 3, there is a rather complicated set of rules for making connections. Furthermore, after a connection is made, the availability list is updated according to another set of rules. (For example, the CV values of the tubes involved in the connection are decreased and if the CV of any tube becomes zero, it must be removed from the availability list.) We avoid a discussion of these rules here and refer the interested reader to [8] (Section 3.4.2, p. 62) for details.

At each step of the algorithm, after one or more connections have been made and the availability list updated accordingly, all tubes available for connection according to the list are examined, starting with those of lowest layer-rank index  $L_r$  and proceeding with increasing  $L_r$  value. For a given tube, all tubes available for connection are examined. As such, this *depth-first search algorithm* is recursive in nature, examining all possible connections in a tree-like manner. If, at any step, all boundary circles of the interface are connected, the resulting set of connections are feasible, then the connection scheme is stored.

There are 22 distinct schemes for a 9-4 (or 4-9) connection problem. The first five of these schemes, as produced by our depth-first algorithm, are shown in Fig. 14. Beneath each scheme is presented a list of the connections in the scheme and how they are coded in terms of a string of tube indices. Note that schemes 3 and 5 have connections that were not discussed earlier – endcap connections that simply redirect a tube from a 9-tube saddlebag back into itself. (Aside: The merge-shift scheme shown in Fig. 10 is produced as No. 14 in our algorithm.) Given the rather large number of schemes for this quite simple 9-4 problem, along with the fact that connection schemes from several interfaces will have to be patched together to form the tubular networks, one expects that a large number of networks can be produced. The computational complexity of this procedure and the need for a computer-based algorithm should be evident.

### 5.2.2. Connecting the interfaces

The above connection procedure must be performed at all interfaces between neighboring blocks  $B_i$  and  $B_{i+1}$ ,  $1 \leq i \leq N-1$ . Suppose that at each  $B_i - B_{i+1}$  interface,  $1 \leq i \leq N-1$ , there are  $n_i$  possible connection schemes. Selecting one scheme from each of the  $n_i$  connection schemes for each  $1 \leq i \leq N-1$  will produce an unfinished tubular network which extends from block  $B_1$  to  $B_N$ . As such, there are  $\prod_{i=1}^{N-1} n_i$  such unfinished networks. For example, in the 9-4-9 problem, where  $N = 3$ ,  $n_1 = n_2 = 22$ , so that there are 484 unfinished networks.

The final step is to “cap” each of these unfinished networks at their ends  $B_1$  and  $B_N$  via endcap connections to produce a tubular network. This is discussed in the next section.

### 5.2.3. Connections at ends of network

The endcap connection problem is somewhat easier than the block interface problem of the previous section. As mentioned earlier, the endcap connection schemes which are possible at one end of a network, say block  $B_1$ , are determined by the merge-shift connection scheme between that block and its neighbour,  $B_2$ .

**Table 2**

Availability list for an end of an 9-4-9 network when the merge-shift connection scheme of Fig. 10 is used at the next interface.

Tube	Available tubes	CD
1	4	1
2	3,5	2
3	2	1
4	1,5	2
5	2,4,6,8	4
6	5,9	2
7	8	1
8	5,7	2
9	6	1

In what follows, we consider, without loss of generality, the endcap connection problem at block  $B_1$  when a given merge-shift connection scheme between blocks  $B_1$  and  $B_2$  is considered. The first step is to construct the *availability list* for tubes in  $B_1$  as determined by the merge-shift scheme. (This list differs from the availability list for block interface connections.) For each tube in  $B_1$ , we consider all of its possible neighbours, but then delete any tubes that will produce undesirable loops. The *connectivity degree* (CD) of each tube will be the resulting number of tubes available to it.

For example, let us return to the merge-shift configuration pictured in Fig. 10. We first consider Tube 1. There are two tubes adjacent to it: Tubes 2 and 4. But Tube 2 merges into Tube 1 at the  $B_1 - B_2$  interface. Connecting Tubes 1 and 2 with endcaps would produce an undesirable dead-end loop through which no fluid can flow. Since Tube 4 is not involved in any merging or shifting with Tube 1, it is available to Tube 1. The connectivity degree of Tube 1 is 1.

Let us now skip to Tube 5. Situated in the center, it has the greatest number of adjacent tubes: Tubes 2, 4, 6 and 8. Tubes 2, 4 and 6 are “uninvolved” with Tube 5 from a merge-shift perspective and are therefore available. Although Tube 8 merges into Tube 5, the two are endcap connectible since this is the second part of a three-to-one merge – see Fig. 11. Therefore, all four adjacent tubes are available to Tube 5 and its connectivity degree is 4. Proceeding in this way for all Tubes 1–9 in block  $B_1$ , we obtain the results presented in Table 2. Note that a tube can be connected to only one other tube – a fundamental difference between this scheme and the block interface connection problem. This difference greatly simplifies the algorithm.

Starting with such an initial availability list, the endcap connection algorithm produces all possible connections in a systematic way. We start with tubes of lowest connectivity degree and consider all possible connections from the availability list. After a connection is made, the availability list is updated by deleting the two tubes that have been connected. The procedure is repeated with the updated availability list, starting with tubes of lowest connectivity degree, until all tubes are connected. As in the block interface case, this algorithm is naturally recursive in nature, producing all possible connections in a tree-like manner.

This algorithm produced the five endcap schemes shown in Fig. 12, which are associated with the merge-shift connection scheme of Fig. 10.

## 6. Fourth step: determining the feasibility of the networks

In the algorithm described above, the connection schemes at each  $B_i - B_{i+1}$  interface are constructed independently of each other. Tubular networks are then formed by considering all possible combinations of these connection schemes. We do not expect that all resulting networks constructed by this “greedy algorithm” will be feasible, i.e., that they will possess characteristics that are deemed desirable for a particular application. Two such characteristics – there may be more, according to the application – are the following:

**Connectedness:** The entire network must be *connected*. Given any two distinct points,  $A$  and  $B$ , in the network (including the inlet and the outlet), there exists a continuous path within the network from  $A$  to  $B$ . The network cannot contain any isolated loops such as the one shown in Fig. 15.

**No “dead ends” or closed cycles:** Fluid must be able to flow through each point in the network. This implies that there must be no closed loops or dead ends in the network such as those shown in Fig. 16. Each point in the network lies on a simple, i.e., nonintersecting, path that extends from the inlet to the outlet.

Whether or not a network exhibits these properties can be determined from a graph which characterizes all of its connections. An example of an appropriate graphical representation of the network of Fig. 11 is presented in Fig. 17. Such graphical representations of our tubular networks may be constructed by means of the following procedure, which makes reference to the above example.

1. To each of the two ends of each tube in the network is assigned a node/vertex. These two vertices are connected by an edge which represents the tube.
2. If Tube  $A$  merges into Tube  $B$ , then the node representing the end of  $A$  which merges into  $B$  is placed on the edge representing  $B$ . An example in Fig. 17 is the tube (9,23) which merges into tube (5,22).

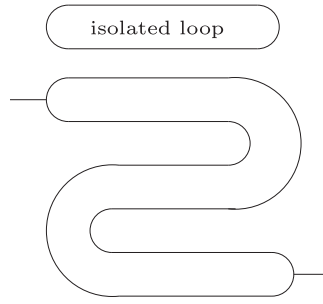


Fig. 15. Example of an isolated loop.



Fig. 16. Some examples of dead ends.

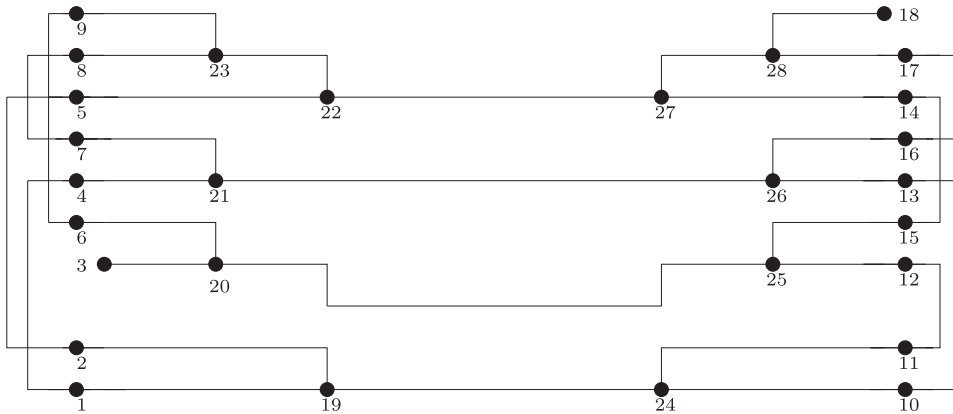


Fig. 17. Graphical representation of the 9-4-9 tubular network of Fig. 11.

3. All endcap connections are represented by edges which connect the appropriate vertices. Examples are (6,9) and (11,12) in Fig. 17.
4. In the case of internal blocks, some additional vertices may need to be added so that only one edge can connect two vertices. This is not the case in Fig. 17.

In order to determine whether or not a network is connected and whether or not there exist closed loops/dead ends, we examine the incidence matrix  $\mathbf{A}(G)$  associated with the graph  $G$  that represents the network [9]. Recall that if  $G$  is a graph with  $n$  vertices and  $m$  edges, its (vertex-edge) incidence matrix  $\mathbf{A}(G)$  is an  $m \times n$  matrix such that

$$a_{ij} = \begin{cases} 1, & \text{if edge } i \text{ is incident to vertex } j \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Here we simply state the main results. For details, see [8], Section 3.5, p. 66.

- **Connectedness:** First of all, if  $m < n - 1$ , then the graph is disconnected. Otherwise, we employ the following result.

**Theorem 1.** *If  $\text{rank} \mathbf{A}(G) < n - 1$ , then the graph  $G$  must have at least two connected components.*

- **Dead ends/Closed cycles:** A bridge in an undirected graph is an edge the removal of which increases the number of connected components of the graph. Therefore, if a graph has a bridge it has dead ends.
- **Step 1:** Using modulo 2 arithmetic, compute a set of basis vectors for the null space  $N$  of  $\mathbf{A}(G)^T$ , the transpose of the incidence matrix of  $G$ . Denote the basis vectors as  $\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_p$ , where  $p = \dim(N)$ .

- **Step 2:** Form the  $n \times p$  matrix  $\mathbf{C} = [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_p]$ .
- **Step 3:** If the matrix  $\mathbf{C}$  contains an entire row of zeros, then the graph  $G$  must have a bridge.

Computing the basis for the null space  $N$  of  $\mathbf{A}(G)^T$  may be accomplished by Gaussian elimination (mod 2) which also yields  $\text{rank} \mathbf{A}(G) = m - \dim N$ . As such, the matrix  $\mathbf{A}(G)^T$  may be used to check for both connectedness and dead ends.

### 6.1. Example: The 9-4-9 saddle-bag problem

We now state some results for the simple 9-4-9 network problem – one inlet and one outlet – in order to give an idea of the complexity of the tubular network problem.

- As mentioned earlier and shown in Fig. 10, there are 22 feasible sets of connections at the interface of a bag and the connector region. For each of these connection schemes there are 4–6 endcap schemes for a total of 118 possible merge-shift connection and endcap schemes. Since a 9-4-9 network is formed by making the union of two possible schemes,  $118^2 = 13,924$  possible networks are generated.
- Of these 13,924 networks, 13,100 are connected.
- Of these 13,100 connected networks, 10,066 have no dead ends or closed cycles.

In summary, our algorithm has generated 10,066 feasible tubular networks for the 9-4-9 problem.

## 7. Concluding remarks

In this paper, we have outlined an algorithm for the construction of conformable tubular networks which occupy, as fully as possible, a given three-dimensional region  $D$ . The algorithm decomposes the region (with assumed longitudinal axis) into a series of blocks with fixed cross-sections. Circles are subsequently packed into each cross-section and connected via a set of rules to produce a tubular network.

The circle-packing method outlined in Section 4 was designed to pack regions with polygonal boundaries. It works very well for convex and nonconvex polygonal regions in which the lengths of the segments comprising the boundary are large compared to the sizes of circles being packed. In the case of more complicated boundaries, for example, irregular curves that could be encountered in practice, there are limitations to this method because it relies on a knowledge of the line segments comprising the boundary. One may wish to employ polygonal approximations of such curves but this may compromise packing efficiency. As such, we have developed another method which can be used in the case of irregular boundaries. This method, however, must work with a discretization of the planar region to be packed. It is described in [10].

More recently, we have also focussed on an extension of the work in Step 1 to accommodate more complicated regions which may not possess a single longitudinal axis. This necessitates the use of morphological methods such as skeletonization to extract principal and subprincipal axes of more complicated regions and to then use this information to devise efficient packing strategies.

Our framework is flexible and can accommodate the specific requirements that may arise in industrial applications (e.g., flow through the tubes, heat transfer, etc.) through the imposition of additional constraints either within the algorithm itself, or during the selection of viable tubular networks.

## Acknowledgments

This research has been supported primarily by a Collaborative Research and Development Grant from the [Natural Sciences and Engineering Research Council](#) (Grant no. [CRDPJ 453649-13](#)). Additional support, in the form of teaching assistantships (JW), was provided by the Department of Applied Mathematics and the Faculty of Mathematics, University of Waterloo.

## References

- [1] <http://www.redneckpoolheater.com>.
- [2] W. Jiang, B. Kettlewell, T. Qiao, F. Mendivil, S.D. Peterson, E.R. Vrscaj, The barbeque pool heater: An algorithm to construct tubular networks that occupy arbitrary regions in  $\mathbb{R}^3$ , in: Proceedings of AMMCS-CAIMS Congress, Waterloo, Ontario, Canada, 2015.
- [3] W. Jiang, T. Qiao, F. Mendivil, S.D. Peterson, E.R. Vrscaj, Some novel circle-packing algorithms devised for the construction of tubular networks in  $\mathbb{R}^3$ , in: Proceedings of AMMCS-CAIMS Congress, Waterloo, Ontario, Canada, 2015.
- [4] R. Gal, D. Cohen-Or, Salient geometric features for partial shape matching and similarity, *ACM Trans. Graphics* 22 (1) (2006) 83–105.
- [5] Y.S. Liu, K. Ramani, Robust principal axes determination for point-based shapes using least median of squares, *Comput. Aided Design* 41 (4) (2009) 293–305.
- [6] P.K. Saha, G. Borgefors, B.S. de Baja, A survey on skeletonization algorithms and their applications, *Patt. Rec. Lett.* 76 (2016) 3–12.
- [7] J.A. George, J.M. George, B.W. Lamar, Packing different-sized circles into a rectangular container, *Eur. J. Oper. Res.* 84 (3) (1995) 693–712.
- [8] W. Jiang, Construction of optimal tubular networks in arbitrary regions in  $\mathbb{R}^3$ , Department of Applied Mathematics, University of Waterloo, 2015 M. Math. Thesis.
- [9] J.L. Gross, J. Yellen, *Graph Theory and its Applications*, second ed., CRC Press, Boca Raton, 2005.
- [10] F. Ghasempour, W. Jiang, F. Mendivil, S.D. Peterson, E.R. Vrscaj, Constructing tubular networks that occupy arbitrary regions in  $\mathbb{R}^3$ , *J. Phys. Conf. Ser.* (2017). In press.