Some novel circle-packing algorithms

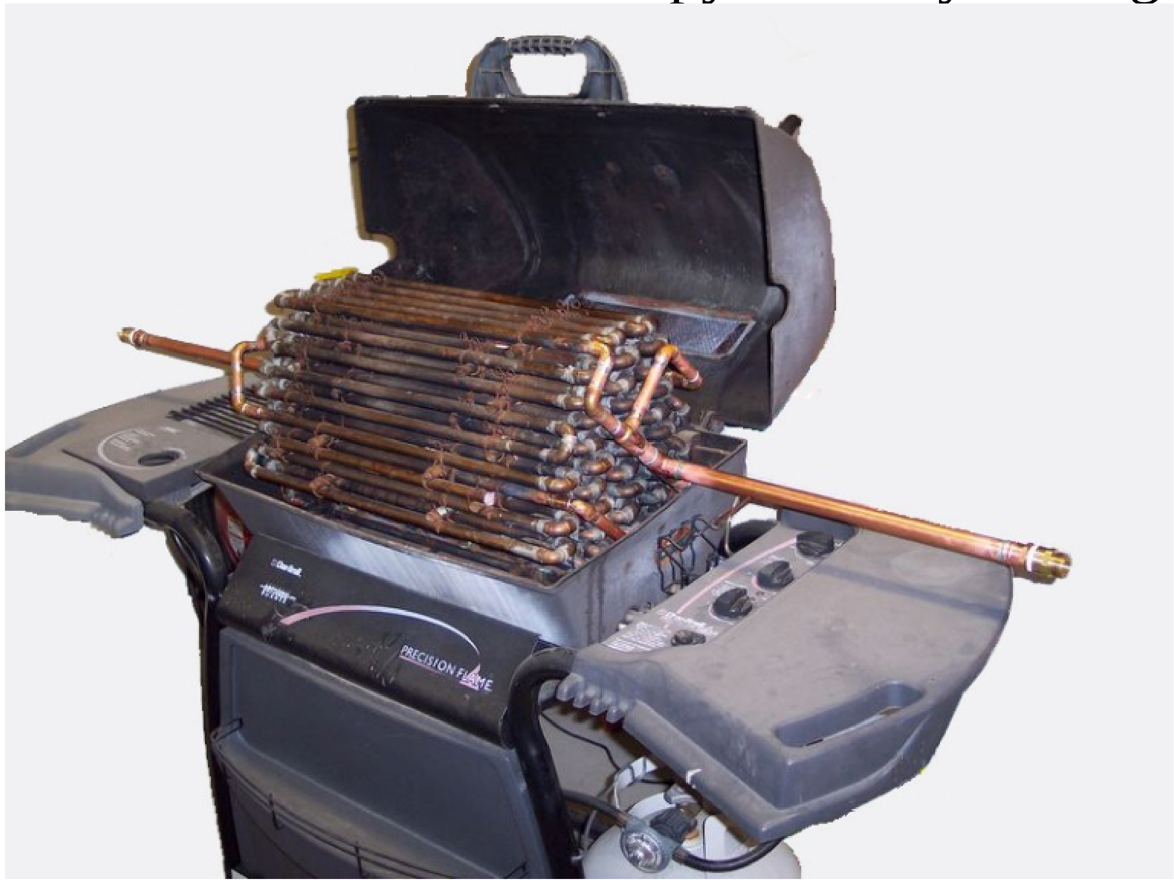
W. Jiang[1], T. Qiao[1], F. Mendivil[2], S. D. Peterson[1] and E.R. Vrscay[1]

[1] *University of Waterloo, Waterloo, Ontario, Canada, {w53jiang,t2qiao,peterson,ervrscay}@uwaterloo.ca*

[2] *Acadia University, Wolfville, NS, Canada, franklin.mendivil@acadiau.ca*

Introduction

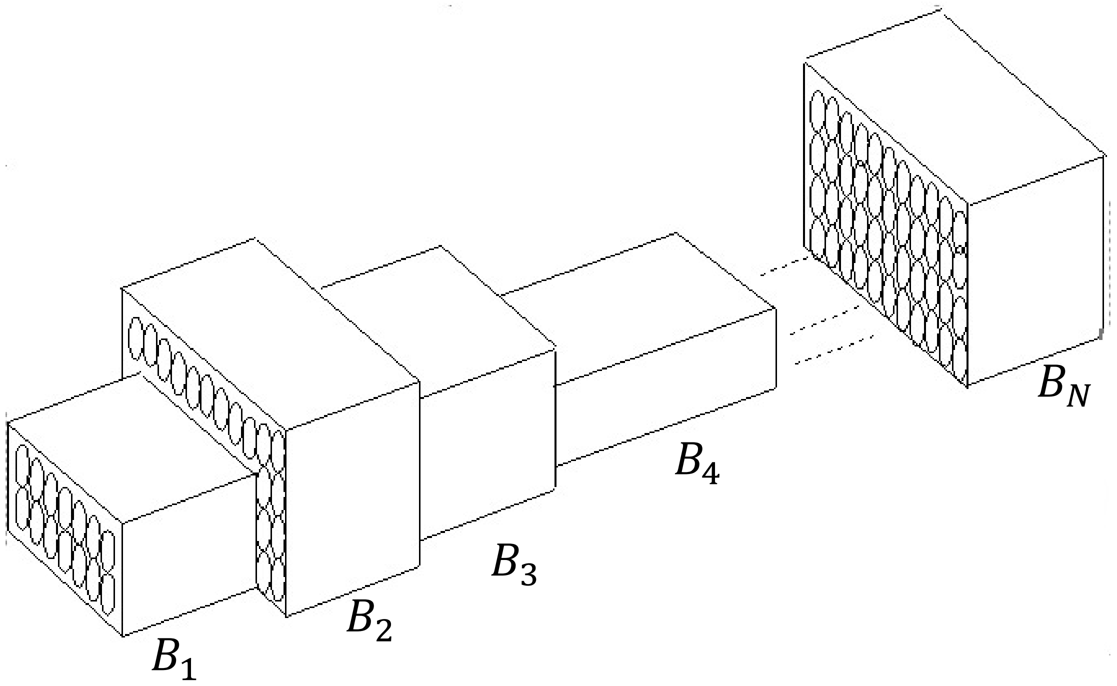
- For constructing tubular networks that occupy arbitrary 3D regions.



- http://www.redneckpoolheater.com/more/2006-11-26_Redneck_Pool_Heater_Manifold/images/100_1180.jpg.

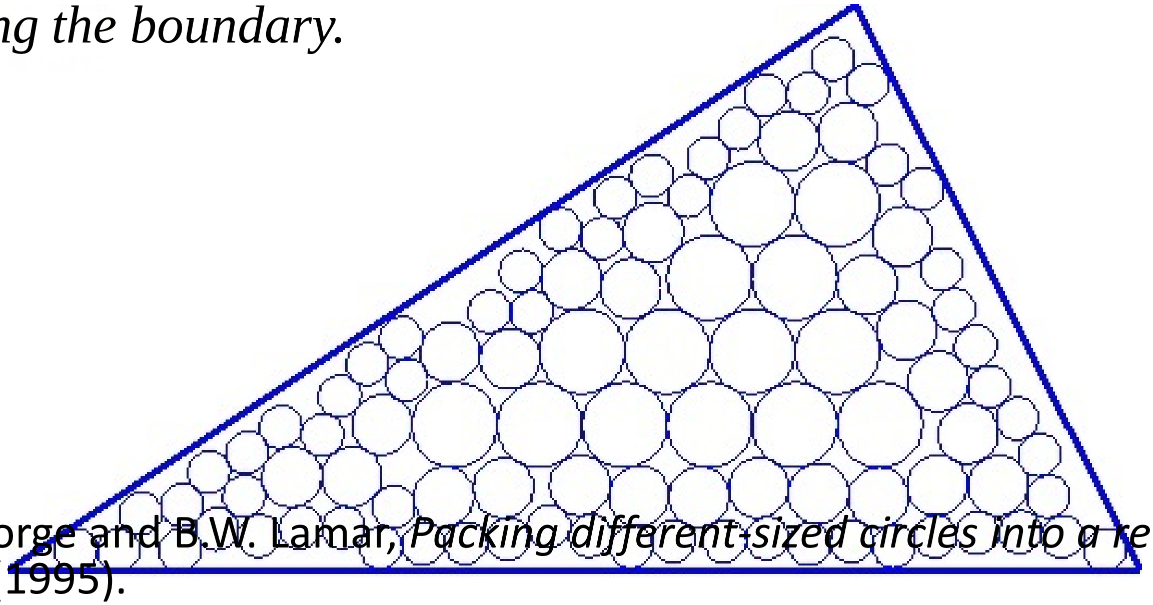
Introduction

- In general, discretize the given volume into blocks B_i and pack tubes in them.
- Packing tubes in 3D reduces to packing circles in 2D.



Introduction

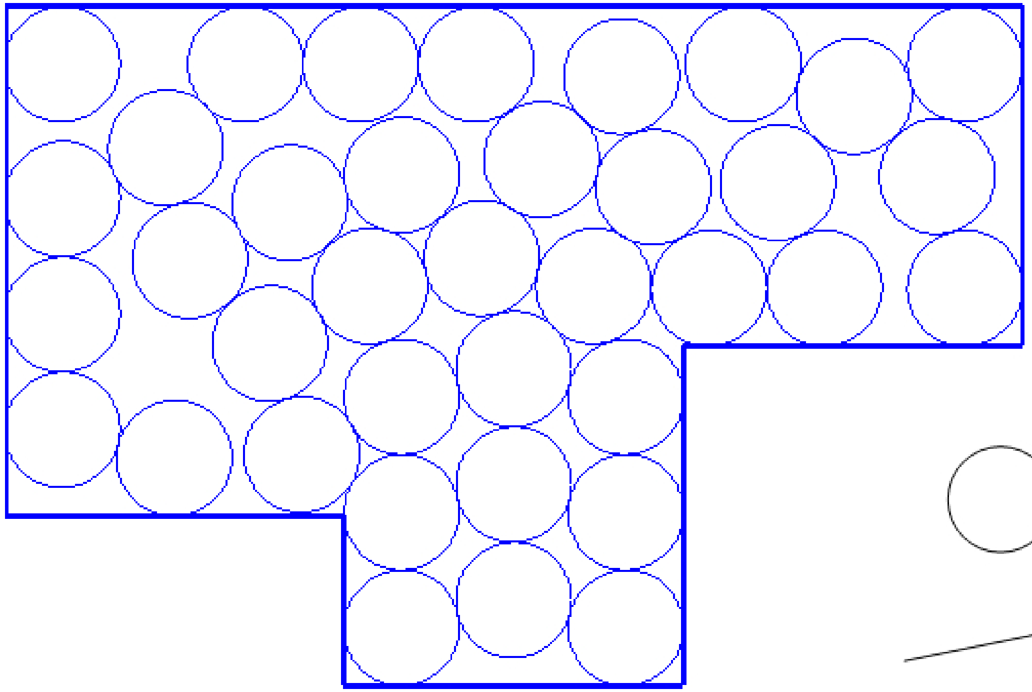
- Pack different-sized circles into an arbitrary polygonal region maximizing covered area.
- Basic method is from [1], which we call GGL circle-packing algorithm.
- A series of algorithms that satisfy our specific constraints for construction of tubular networks in arbitrary 3D space.
    - *Big circles in the central region, small circles near boundary.*
    - *Empty space along the boundary.*

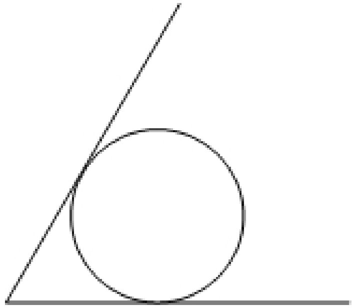


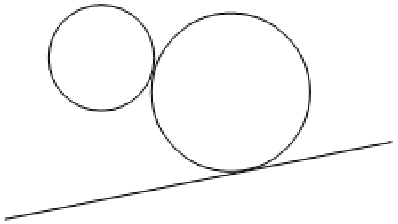
- [1] J.A. George, J.M. George and B.W. Lamar, *Packing different-sized circles into a rectangular container*, Eur. J. Op. Res. **84**, 693-712 (1995).

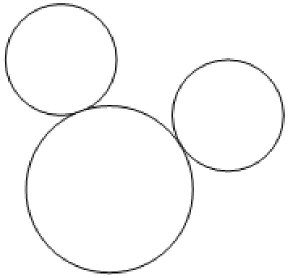
Algorithm 1. GGL-based circle-packing

- Start packing at corners and grow by “one-side” or “two-circle”.





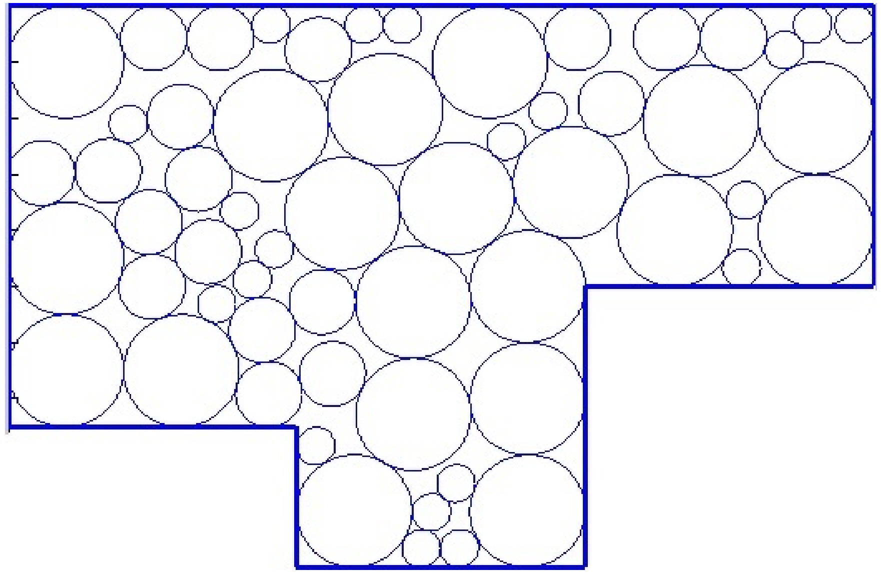




Algorithm 1. GGL-based circle-packing

Multiple sizes

- Run the algorithm for a certain number of iterations and keep record of the best one.



50 iterations; 58 circles packed; Packing ratio=0.788

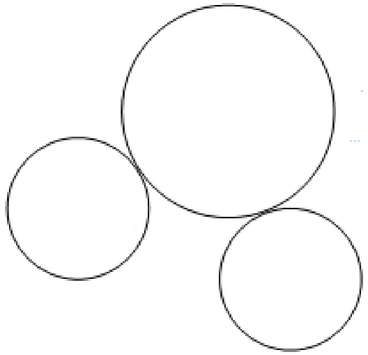
Algorithm 1. GGL-based circle-packing

Placing circles by position numbers

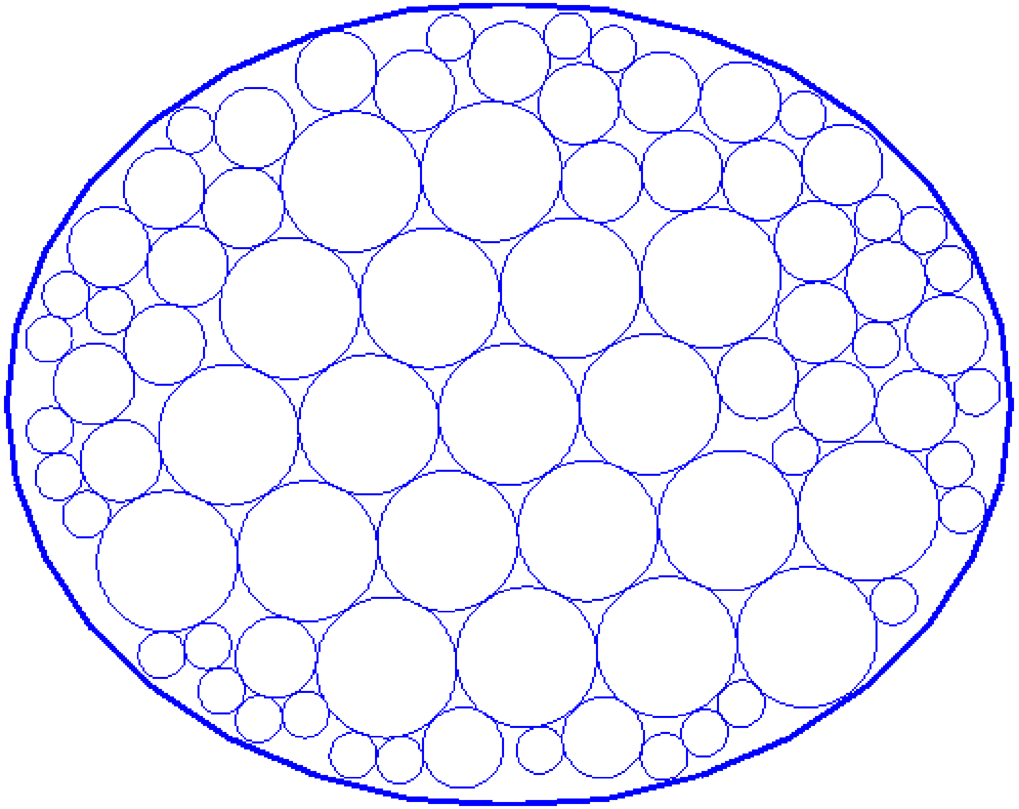
- Position numbers 1 to n_{pc} represents n_{pc} packing corners.
- Position numbers defined for circle *k* with circles *i* and *j* already placed.
- **A1** does not meet our needs, but this method can be extended to other algorithms.

	Side 1	⋯⋯	Side n	j=1	j=2	j=3
i=1	$n_{pc} + 1$	⋯⋯	$n_{pc} + n$			
i=2	$n_{pc} + n + 1$	⋯⋯	$n_{pc} + 2n$	$n_{pc} + 2n + 1$		
i=3	$n_{pc} + 2n$ +2	⋯⋯	$n_{pc} + 3n + 1$	$n_{pc} + 3n + 2$	$n_{pc} + 3n + 3$	
i=4	$n_{pc} + 3n$ +4	⋯⋯	$n_{pc} + 4n + 3$	$n_{pc} + 4n + 4$	$n_{pc} + 4n + 5$	$n_{pc} + 4n + 6$

Algorithm 2. Reversed-GGL circle-packing



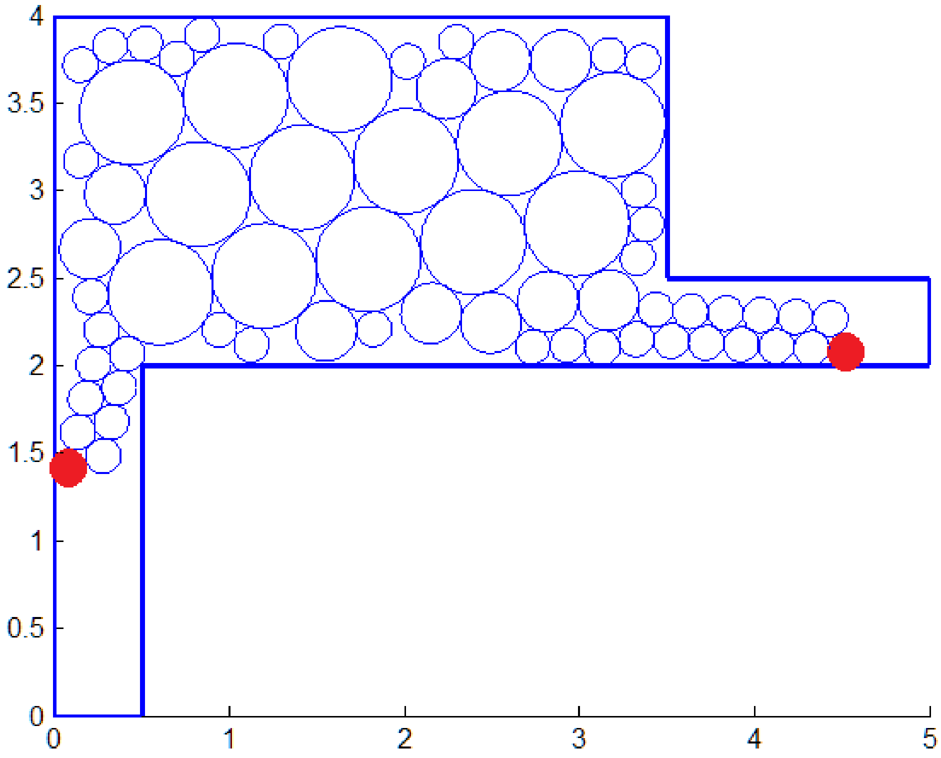
- Start packing with big circles with radius R_b.
- Pack the first circle at the centroid of the region: $(\bar{x}, \bar{y})$.
- Pack the second circle right next to it: $(\bar{x} + 2R_b, \bar{y})$
- Grow only by two-circle packing;



Algorithm 2. Reversed-GGL circle-packing

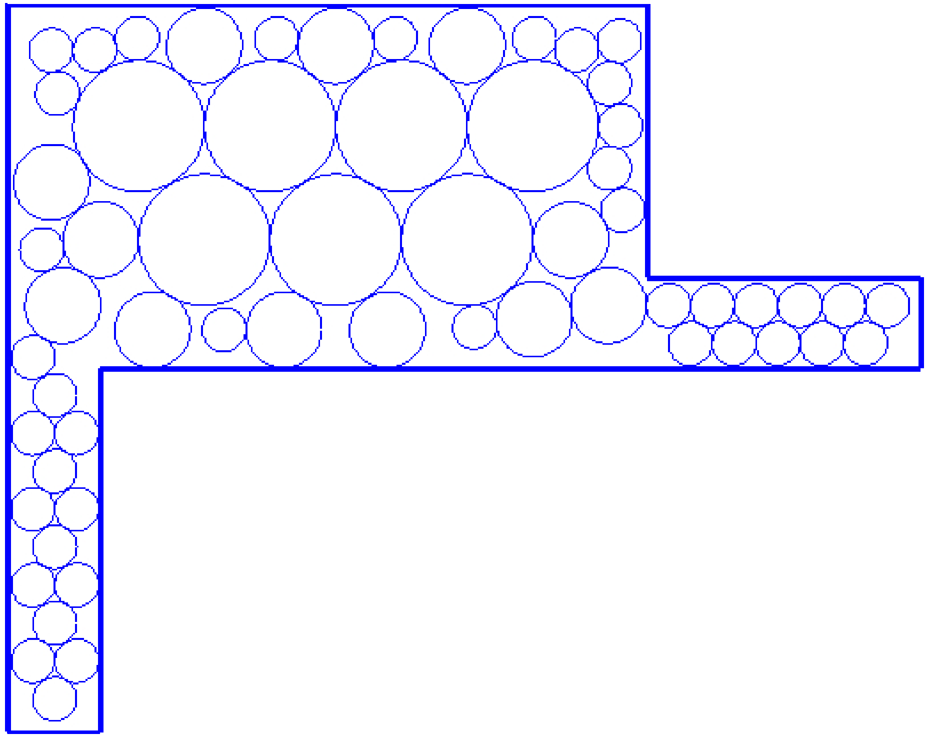
Limitation

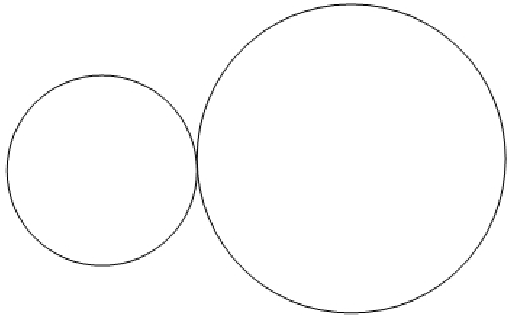
- Sometimes cannot reach corners because only by two-circle packing is allowed.

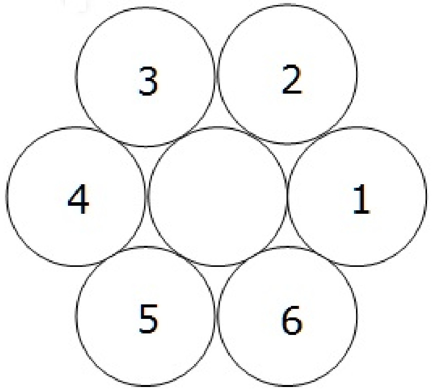


Algorithm 3. Reversed-GGL with one-circle packing

- Add one-circle packing to Reversed-GGL.
- Degree: the number of circles that could be placed around a circle.

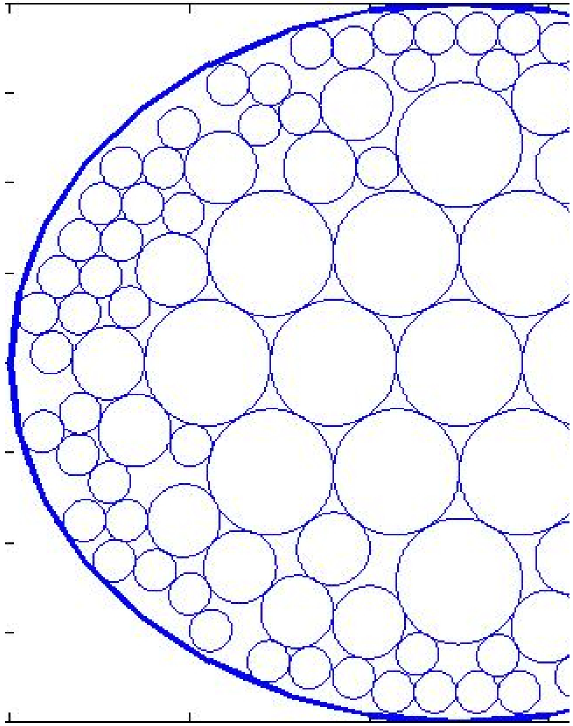


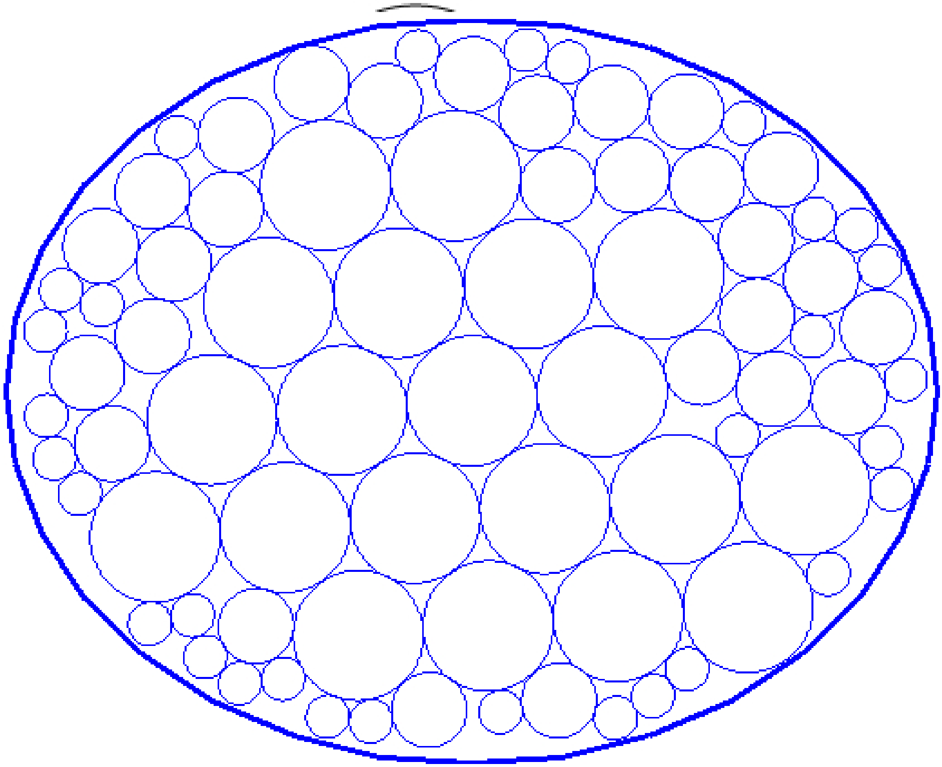


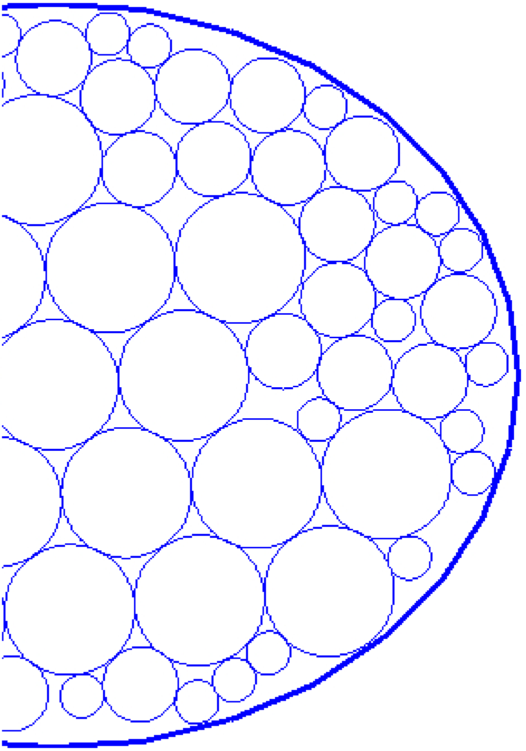


Algorithm 4. Reversed GGL with one-circle packing and additional constraints.

- Additional constraints on the distance from the center of big/medium circle to the boundaries. (flexible)





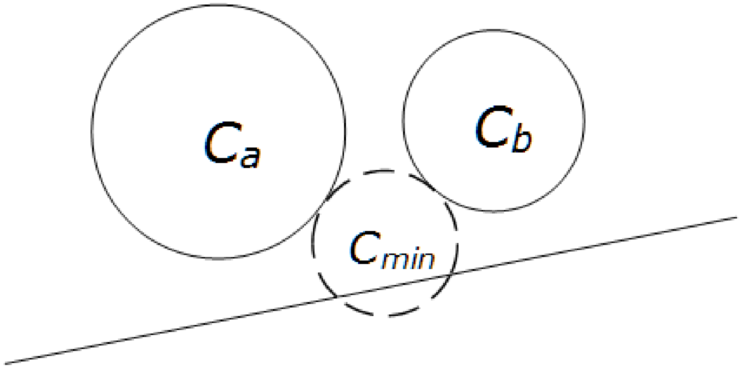


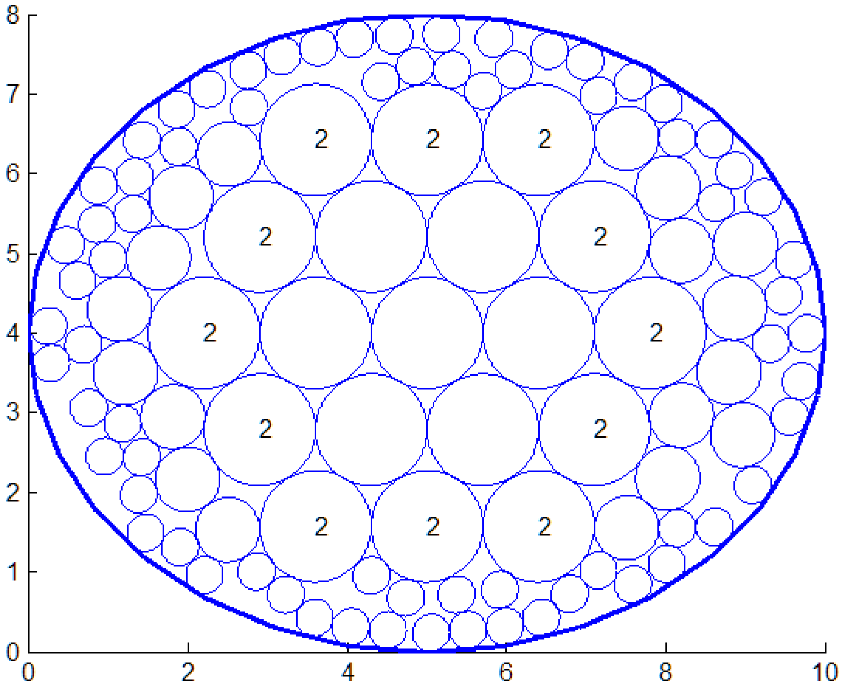
Algorithm 4

A2. Reversed GGL

Algorithm 5. Hybrid circle-packing algorithm

- Step 1. Use Reversed-GGL algorithm to fill the polygon.
- Step 2. Use a smallest circle to identify and remove the outermost layer of circles.
- Step 3. Use GGL to pack smaller circles between the boundary and the second layer.

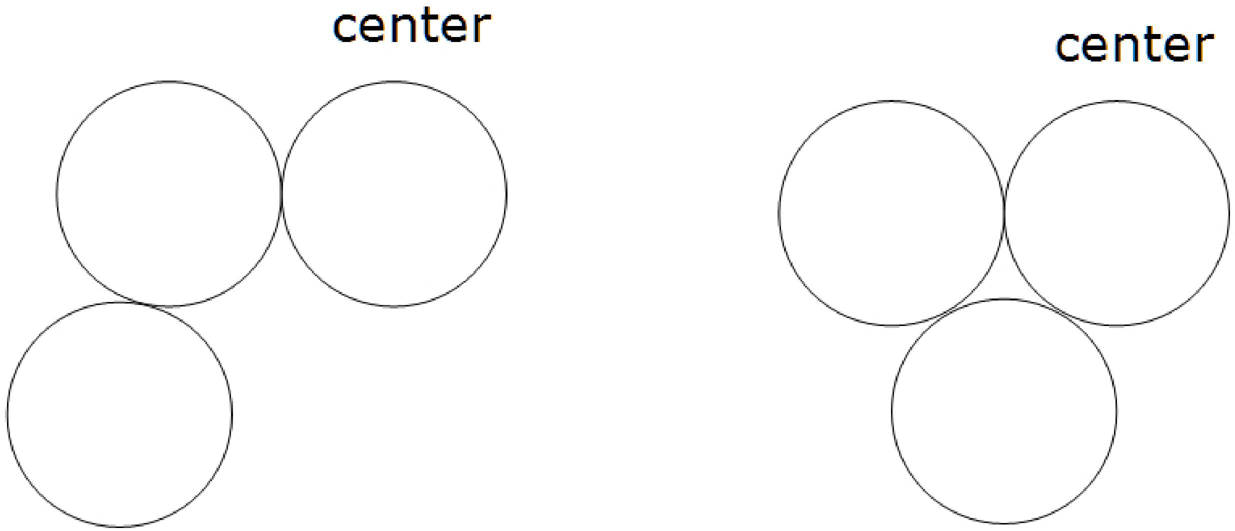




Algorithm 6. Jiggling

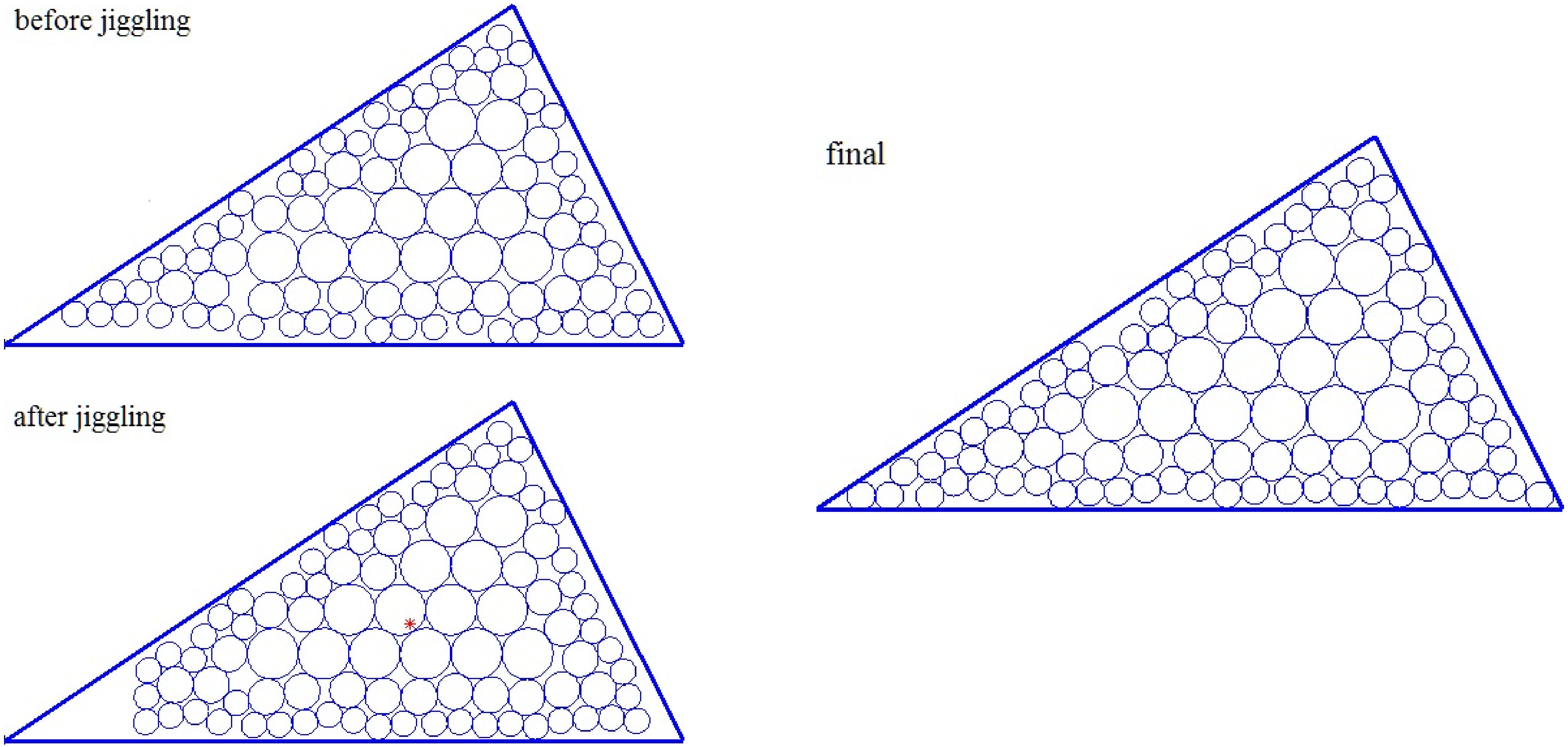
Improve a given circle-packing

- Calculate the center of mass of a packing.
- “Black hole”: moves circles near the boundary towards the center of mass by two-circle packing.



Algorithm 6. Jiggling

Improve a given circle-packing



Conclusions

- Extended GGL circle packing algorithms to arbitrary polygons.
- Developed Reversed-GGL circle packing algorithm which starts from the interior of the region and achieves the following goals:
  - *Big circles in the central region, small circles near boundary.*
  - *Empty space along the boundary.*
- Designed a simple algorithm that can simulate “black hole” for circle packing.

Thank you!