SCIENCE CHINA
Information Sciences

• HIGHLIGHT •

April 2018, Vol. 61 048104:1–048104:3
https://doi.org/10.1007/s11432-017-9223-3

Packing unequal circles into a square container based on the narrow action spaces

Kun HE[1,2], Mohammed DOSH[1,3*], Yan JIN[1] & Shenghao ZOU[1]

[1]*School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan* 430074*, China;*
[2]*Shenzhen Research Institute of Huazhong University of Science and Technology, Shenzhen* 518057*, China;*
[3]*Faculty of Education for Girls, Kufa University, Najaf* 00964*, Iraq*

Received 22 March 2017/Accepted 16 August 2017/Published online 16 March 2018

Citation He K, Dosh M, Jin Y, et al. Packing unequal circles into a square container based on the narrow action spaces. Sci China Inf Sci, 2018, 61(4): 048104, https://doi.org/10.1007/s11432-017-9223-3

The single container packing problem consists of allocating a number of items inside one container in order to find the densest packing patterns without overlapping. As an important class of optimization problems, packing problems have numerous applications in industry and academia, such as applied mathematics, material manufacturing, material cutting, logistics, wireless communication and fashion industry. As an NP-hard problem, however, there is no exact algorithm to obtain optimality in polynomial time unless P = NP, and researchers have resorted to heuristics or approximation methods. Hifi and M'Hallah [1] reviewed the most relevant literature on efficient models and methods for packing circular items in different types of containers. The circle packing problem (CPP) is classified into two categories based on whether the circle items are equal [2] or unequal [3]. Other variants exist that consider additional constraints, such as the CPP with equilibrium constraints [4].

Problem definition. We address an important version of the circle packing problem, which aims to find a non-overlapping dense packing pattern for n unequal circle items in a two-dimensional square container (PUC-SC) such that the size of the container is minimized. Given n ($n \in N^+$) circles $C_1, C_2, \ldots, C_n$ with integer radii $r_1, r_2, r_3, \ldots, r_n$, we are asked to find the smallest square container of size L such that all circle items are packed feasibly, i.e., no overlapping between any pair-wise circles ($C_i \bigcap C_j = \phi$) and all circles fit completely inside the container.

Specifically, let the centre of the square container be located at the origin of a two-dimensional Cartesian coordinate system. Denote the centre coordinate of circle C_i ($1 \leqslant i \leqslant n$) as (x_i, y_i). We define a packing pattern as $X = (x_1, y_1, \ldots, x_i, y_i, \ldots, x_n, y_n)$. Let $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ indicate the Euclidean distance between the centres of circle C_i and C_j. The PUC-SC can be formulated as follows:

$$\begin{aligned} &\min \ L, \\ &\text{s.t. } (1)\ \max(|x_i|, |y_i|) \leqslant (L/2 - r_i), \quad 1 \leqslant i \leqslant n, \\ &\qquad (2)\ d_{ij} \geqslant (r_i + r_j), \quad 1 \leqslant i < j \leqslant n. \end{aligned}$$

Constraint (1) imposes that any circle should not be extended outside the container, and constraint (2) imposes no overlap between any pair-wise circles.

The proposed PAS-PCI approach. We propose an effective algorithm based on the partitioned action space and partitioned circle items (PAS-PCI) for solving the PUC-SC. The proposed method is inspired by our previous work. In 2011 and 2012, He et al. [5] defined the conception of action space for the rectangular packing

* Corresponding author (email: i201522007@hust.edu.cn; mohammedh.dosh@uokufa.edu.iq)

problem, and each maximal unoccupied rectangular space is called an action space for a packing pattern. By an innovative idea of approximating each circle as a square, He et al. [6] proposed an action space based global optimization (ASGO) algorithm for the problem that packs unequal circles into a square container. For some given patterns, ASGO utilized the limited-memory Broyden-Fletcher-Goldfarb-Shanno (LBFGS) algorithm [7] for continuous optimization to reach the local minimums. Then, each circular item is approximated as a square item, and an action-space-based basin-hopping strategy is adapted to find the larger unoccupied spaces as candidate places to replace some of the most overlapping items in order to jump from a local minimum.

In this article, PAS-PCI approximates each circular item as a square item by setting its width as $[1 + (1/\sqrt{2})]r_i$, as shown in Figure 1, which is in the intermediate value of $\sqrt{2}r_i$ and $2r_i$. Based on our observation that the small circles and the large unoccupied circle are robustly complementary to each other, PAS-PCI uses a new basin-hopping strategy by partitioning the narrow action spaces and partitioning circle items according to their sizes.

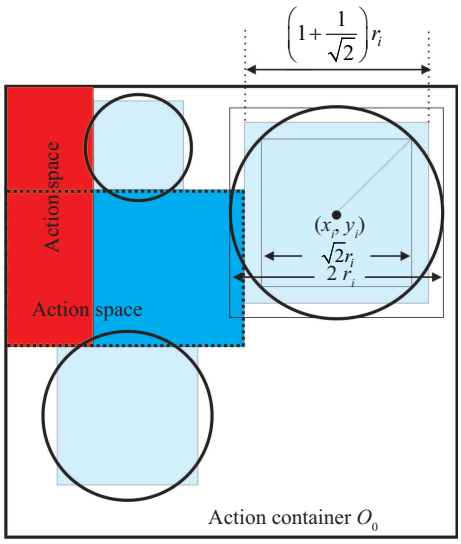

Figure 1 (Color online) An example of the narrow action spaces.

Definition 1 (Partitioned circle items)**.** For a pattern (X, L) with n circle items, we partition the items into four sets, S_1 to S_4, based on their radii. Without loss of generality and assuming the circles are numbered from 1 to n from small to large, we obtain [1, int$(n/4)$], [int$(n/4)$ +1, int$(n/2)$], [int$(n/2)$ +1, int$(3n)/4$], and [int$(3n/4)$ +1, n].

Definition 2 (Narrow action space)**.** An action space i is called narrow if the long side is at least double the short side, i.e., either $h_i \leqslant 0.5w_i$, or $w_i \leqslant 0.5h_i$.

PAS-PCI includes three phases: the initial continuous optimization phase, the iterative processing phase and the post-processing phase. The details are shown in Algorithm 1.

Algorithm 1 The proposed PAS-PCI algorithm

Require: Container size L_0, item sizes $r_1, \ldots, r_n$, number of patterns G, selection number m;
Ensure: Feasible pattern (X, L) or Null;
repeat
  Randomly generate G patterns, store in array A;
  for each pattern A[i] **do**
    A[i] ← LBFGS (A[i]);
    if U_e(A[i]) $< 10^{-20}$ **then**
      (A[i],L)←post-processing algorithm (A[i],L);
      return (A[i],L);
    end if
  end for
  $k_b \leftarrow 0$;
  repeat
    for $k_p = 1$ **to** 20 **do**
      Select m patterns with the lowest U_e;
      Generate 39 new patterns using the basin-hopping algorithm for each selected pattern;
      for each of the new $39 \times m$ patterns **do**
        $(X', L) \leftarrow$ LBFGS(X, L);
        if $U_e(X') < 10^{-20}$ **then**
          $(X', L) \leftarrow$ post-processing(X', L);
          return (X', L);
        end if
      end for
    end for
    Select m patterns with the lowest U_e, do perturbation operator, $k_b \leftarrow k_{b+1}$;
  until $k_b > 5$;
until reach the limitation of time;
return NULL.

The initial continuous optimization phase is to generate patterns with local minimum potentials and to select good patterns for the iterative processing phase. The container size L_0 is initialized by the current best record published by ASGO. We randomly generate G patterns ($G = 32$) to increase the diversification and then use LBFGS on each pattern for continuous optimization to reach their local minima. We will switch to the post-processing phase if a feasible solution is found.

The iterative processing phase calls the basin-hopping algorithm to jump out of the local minima. We select m patterns ($m = 3$) with the lowest total elastic potential energy from the set of patterns G_i ($i = 1, 2, \ldots, 32$) and apply the basin-hopping algorithm to the selected patterns to obtain a total of $39 \times m = 117$ new patterns to increase the diversification. Afterwards, we apply k_p successive iterations ($k_p = 20$) of basin hopping. At each iteration, we run the basin-hopping algorithm to generate new patterns that inherits some good properties of the old patterns. Then, LBFGS

is applied to reach the local minima. We check the feasibility at the end of each iteration and switch to the post-processing phase if there exists a feasible pattern. If we execute the basin-hopping algorithm for $k_p = 20$ times and cannot obtain a feasible solution, we then believe that the current patterns cannot be improved using the basin-hopping strategy alone. We therefore modify the perturbation operator proposed by Fu et al. [8] to perturb each of the $m = 3$ selected patterns to jump out of the local minimum trap. The perturbation operator will guide the patterns to some promising area by constructing an updated pattern instead of destroying the pattern. Then, we run the 20 iterations of basin hopping plus continuous optimization again. The outer iteration with the perturbation will run for at most $k_b = 5$ times. If we still cannot find a feasible solution in k_b attempts, we consider that the initial patterns make it difficult to reach a feasible solution and restart the whole process.

If we obtain a feasible solution from the above two phases, we then use the post-processing strategy to further reduce the container size while maintaining the feasibility to improve the solution.

Computational results. PAS-PCI was coded in C++ and run on a personal computer with 3.0 GHz and 4.0 GB memory. To assess the efficiency of the proposed approach, we tested it on two groups of benchmark instances: $r_i = i$ and $r_i = \sqrt{i}$ (r_i is the radius of circle i, $1 \leqslant i \leqslant n$). The instances for $r_i = i$ are large-variation instances, and the second group $r_i = \sqrt{i}$ contains small-variation instances. The range of n is from 1 to 72 on the Packomania website. Given its stochastic nature, we run PAS-PCI independently 10 times to solve each instance. For each instance, the search process terminates when it reaches the time limit or we obtain a feasible solution that is better than or equal to the current best result. All the parameters are set based on a small number of trials or referred to parameters of the previous algorithm ASGO. We compared PAS-PCI with ASGO [6] and the best results downloaded from the Packomania website in October 2015, which were collected from different experts[1)]. The best records obtained in these experiments are maintained[2)] by Eckard Specht. In general, the proposed PAS-PCI equipped with its particular features attains a highly competitive performance while comparing with state-of-art algorithms. In comparison with ASGO on the 68 tested instances, PAS-PCI achieves smaller containers for 64 instances and matches the other 4. In comparison with the best records of the Packomania website for 94 instances in October 2015 when PAS-PCI uploaded the new results to this website, PAS-PCI finds smaller containers for 82 instances and matches the other 12. In particular, PAS-PCI can find improved records for 19 instances (47–48, 51–54, 57, 61–72) that had remained unchanged since 2013. Even comparing with the current results on the Packomania website in March 2017, PAS-PCI can find smaller containers for 10 instances.

Conclusion. This article proposes a PAS-PCI algorithm for packing unequal circles in a two-dimensional square container (PUC-SC). The PAS is used to partition the narrow action space on the long side to get two equal action spaces to make a maximum utilization of the unoccupied spaces. While the PCI is used to partition the circle items into four groups based on item sizes for the basin-hopping strategy. Experimental evaluations on two sets of benchmark instances showed that the proposed algorithm is highly competitive in comparison with the state-of-art ASGO algorithm and the best records of the Packomania website.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61472147, 61370183, 61602196, 61772219), and Shenzhen Science and Technology Planning Project (Grant No. JCYJ20170307154749425).

References

1 Hifi M, M'Hallah R. A literature review on circle and sphere packing problems: models and methodologies. Adv Oper Res, 2009. doi: 10.1155/2009/150624
2 Szabó P G, Markót M C, Csendes T, et al. New Approaches to Circle Packing in a Square: With Program Codes. Berlin: Springer, 2007
3 Liu J F, Zhang K W, Yao Y L, et al. A heuristic quasi-physical algorithm with coarse and fine adjustment for multi-objective weighted circles packing problem. Comput Ind Eng, 2016, 101: 416–426
4 He K, Mo D Z, Ye T, et al. A coarse-to-fine quasi-physical optimization method for solving the circle packing problem with equilibrium constraints. Comput Ind Eng, 2013, 66: 1049–1060
5 He K, Huang W Q, Jin Y. An efficient deterministic heuristic for two-dimensional rectangular packing. Comput Oper Res, 2012, 39: 1355–1363
6 He K, Huang M L, Yang C K. An action-space-based global optimization algorithm for packing circles into a square container. Comput Oper Res, 2015, 58: 67–74
7 Liu D C, Nocedal J. On the limited memory BFGS method for large scale optimization. Math Program, 1989, 45: 503–528
8 Fu Z H, Huang W Q, Lü Z P. Iterated tabu search for the circular open dimension problem. Eur J Oper Res, 2013, 225: 236–243

1) Packomania website 2012-2015: www.packomania.com.
2) www.packomania.com.