European Journal of Operational Research 250 (2016) 615–627

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Innovative Application of O.R.

Iterated Tabu Search and Variable Neighborhood Descent for packing unequal circles into a circular container

Zhizhong Zeng [a,b], Xinguo Yu [a], Kun He [b,*], Wenqi Huang [b], Zhanghua Fu [c]

[a] National Engineering Research Center for E-Learning, Central China Normal University, 152 Luoyu Road, Wuhan 430079, China
[b] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
[c] LERIA, Université d'Angers, 2 Boulevard Lavoisier, 49045 Angers, France

ARTICLE INFO

Article history:
Received 19 November 2014
Accepted 1 September 2015
Available online 8 September 2015

Keywords:
Packing
Tabu Search
Variable Neighborhood Descent
Iterated Local Search

ABSTRACT

This paper presents an Iterated Tabu Search and Variable Neighborhood Descent (ITS-VND) algorithm for packing unequal circles into a circular container (PUCC). The algorithm adapts the Tabu Search procedure of Iterated Tabu Search algorithms and proposes a Tabu Search and Variable Neighborhood Descent (TS-VND) procedure. We observe there are strong complementarities between the small circles and the large vacant places, and propose the insert neighborhood to match up the small circles with the large vacant places. Although the insert neighborhood is inefficient and time-consuming, it is an important supplement to the classic swap neighborhood as it could arrange the small circles properly. Predicated on these features, we employ the insert neighborhood only at chosen local minima of the swap neighborhood that shows promise for an improvement. The traditional Tabu Search procedure is then transformed into a hybrid procedure composed of two alternative parts, namely Variable Neighborhood Descent and Tabu Search respectively. Besides this reformed procedure, ITS-VND also incorporates other new features, such as an adaptive evaluation function, a novel method for accelerating the neighborhood exploration, and the "collision accidents" criterion for evaluating how intensively the area near the current solution has been explored. The computational results on three well established benchmark sets show that the proposed algorithm not only has a good *discovery capability* but also can provide good results within a reasonable time. For a total of 84 benchmark instances, the proposed algorithm improves the best-known results on 23 instances, matches 60, and only misses one.

1. Introduction

In the past decades, the circle packing problem has been an active research issue in the cutting and packing (C&P) problem family. The problem is to place given rigid circles into a container without overlap. The problem is widely encountered in industry and science fields, including areas such as material manufacturing, logistics, wireless communication, architecture layout, and molecular construction analyses (Adickes et al., 2002; Birgin, Martinez, & Ronconi, 2005; Castillo, Kampas, & Pint´er, 2008). As a classic NP-hard problem, it is difficult to obtain exact solutions. So researchers usually resort to heuristic methods. This paper focuses on an important case of it where the shape of the container is circular and the radii of the circles are not uniform, known as the problem of packing unequal circles into a circular container (denoted by PUCC). We describe it formally as follows.

Given a circular container D_0 with undetermined radius R, as well as n ($n \in N^+$) circles D_1, D_2, ..., D_n with known radii r_1, r_2, ..., r_n ($r_1 \le r_2 \ldots \le r_n$). Let the center of D_0 be the origin of the two-dimensional coordinate system and let the coordinate of the center of D_i ($i \in [1, n]$) be (x_i, y_i). The problem is to find a feasible (non-overlapping) solution (R, X) with smallest container radius R, where X is a configuration denoted by $(x_1, y_1, \ldots, x_i, y_i, \ldots, x_n, y_n)$. The objective can be formulated below:

Minimize R, *s.t.*

$$\sqrt{x_i^2 + y_i^2} \le R - r_i, \quad \forall 1 \le i \le n, \tag{1}$$

$$\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \ge r_i + r_j, \quad \forall 1 \le i < j \le n. \tag{2}$$

Formula (1) forbids any overlap between any circle and the container, and formula (2) forbids any overlap between any two circles.

A solution is feasible if it meets the above formulas. An infeasible solution can exist in the intermediate optimization process, and the following potential energy model is used to measure the infeasibility

* Corresponding author. Tel.: +86 13006163312.
E-mail addresses: never_stop_try@163.com (Z. Zeng), xgyu@mail.ccnu.edu.cn (X. Yu), brooklet60@gmail.com (K. He), wqhuang@mail.hust.edu.cn (W. Huang), fuzhanghua1984@163.com (Z. Fu).

http://dx.doi.org/10.1016/j.ejor.2015.09.001
0377-2217/

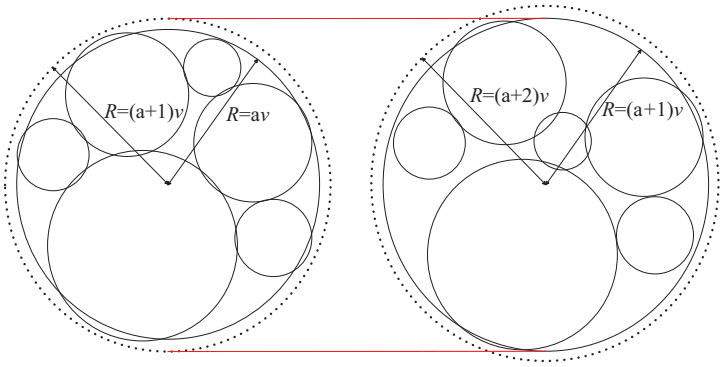

Fig. 1. Adjust the container radii to make feasible solutions comparable.

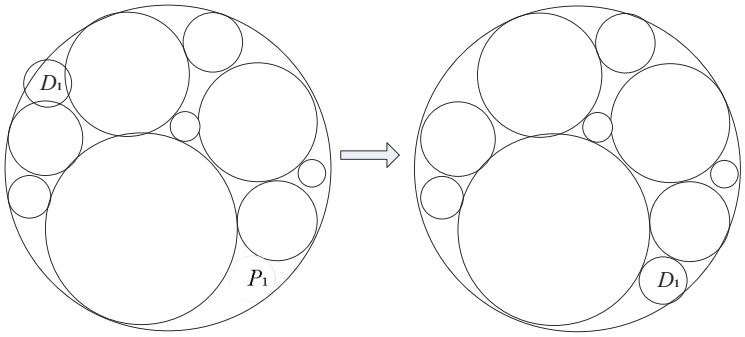

Fig. 2. Insert neighborhood illustration.

of an infeasible solution (Huang & Xu, 1999):

$$U(X) = \sum_{i=0}^{n-1} \sum_{j=i+1}^{n} \frac{1}{2} O^2{}_{ij}. \tag{3}$$

O_{ij} stands for an overlap. When $i = 0$, O_{ij} is the overlap between the container and circle D_j:

$$O_{0j} = Max\{\sqrt{x_j{}^2 + y_j{}^2} + r_j - R, 0\}. \tag{4}$$

When $i > 0$, O_{ij} is the overlap between circle D_i and circle D_j:

$$O_{ij} = Max\{r_i + r_j - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, 0\}. \tag{5}$$

Previous algorithms which are based on above model fix the container radius on a proper value, and use the potential energy U alone as the evaluation function. However, since there is no way to compare two feasible solutions by U, when the current solution becomes feasible they have to end the process and reduce the container radius. To evaluate the solutions more adaptively, we combine R and $U(X)$ in lexicographical order to form a new evaluation function. When comparing two solutions, we consider one is better than the other only in two conditions: first, the container radius of the former is smaller; second, they have equal container radii but the former has a smaller potential energy. To make the new evaluation function work, we limit the candidate container radii R to some discrete values kv ($k \in N^+$and v is a predefined small constant like 0.1). When we reach a feasible solution (R, X), we reduce R to the largest kv which cannot make the solution feasible again by LBFGS (Liu & Nocedal, 1989) optimization. For example, for the left solution illustrated in Fig. 1, by setting $R = (a + 1)v$ we can get a feasible solution through LBFGS optimization, but by setting $R = av$ we cannot. So we set $k = a$ and adjust the container radius to av. Similarly the container radius of the right solution is adjusted to $(a + 1)v$. By this means, even without knowing the final potential energy U of these two solutions, we still can judge the first solution as better by simply comparing their final container radii. This example illustrates how R works in the evaluation function.

Then we can first minimize the evaluation function $\langle R, U(X) \rangle$ iteratively and then further optimize the final solution into a feasible and compact one by using techniques such as SNOPT (Addis, Locatelli, & Schoen, 2008a). Following this approach we propose the Iterated Tabu Search and Variable Neighborhood Descent algorithm (ITS-VND) to minimize $\langle R, U(X) \rangle$. Compared with the existing Iterated Tabu Search algorithms for this problem (Fu, Huang, & Lü, 2013; Huang, Fu, & Xu, 2013; Huang, Zeng, Xu, & Fu, 2012; Ye, Huang, & Lü, 2013), our algorithm has two major new features: a new neighborhood and a mechanism to combine different neighborhoods.

The first new feature is the new neighborhood. Tabu Search (Glover, 1989, 1990) is the basic search engine of Iterated Tabu Search, and how to define moves to construct proper neighborhoods is one of its most important issues. The existing Iterated Tabu Search algorithms for the PUCC problem use the swap moves or the union of the swap and the random relocate moves to construct neighborhoods for their Tabu Search procedures (Huang et al., 2013 , 2012; Ye et al., 2013). They have obtained state-of-the-art results recently. However, although the swap moves are powerful, they cannot efficiently handle some occasions. For example, considering the left solution in Fig. 2, one can relieve its infeasibility considerably by placing circle D_1 into a vacant place P_1 (see the right solution in Fig. 2). But this is difficult to accomplish by swapping two circles or randomly relocating a small circle. To handle this kind of occasions efficiently, we introduce the insert neighborhood. The new neighborhood is developed from the random relocate neighborhood (Huang et al., 2013) and is inspired by the Vacancy Search algorithm for packing equal circles into a square (Huang & Ye, 2010). A solution in the insert neighborhood is formed by picking out a small circle (such as D_1 in Fig. 2) from the current solution, and placing it back at a large vacant place (such as P_1 in Fig. 2), and further continuously optimizing the resulting solution (The vacant places are calculated beforehand). Comparing with the existing swap and random relocate moves, one can observe that the insert moves can make the best of the complementarities between the small circles and the large vacant places.