An effective hybrid algorithm for the problem of packing circles into a larger containing circle

De-fu Zhang*, An-sheng Deng

Department of Computer Science, Xiamen University, 361005, China

Abstract

Simulated annealing is a powerful stochastic search method, but it still has the disadvantage of blind search. Tabu search (TS) which can prevent cycling and enhance diversification, is an adaptive strategy based on tabu list. By reasonably combining simulated annealing with TS, an effective hybrid algorithm for the problem of packing circles into a larger containing circle is presented. Based on a special neighborhood and tabu strategy, some benchmark problem instances can be well solved by the presented hybrid algorithm, and the computational results can compete with the best literature results.

Keywords: Packing problem; Simulated annealing; Tabu search; NP-hard

1. Introduction

Given \( n \) objects and a bounded space, each with given shape and size, packing problems consist in determining how to best pack these objects into the bounded space without overlapping. In this paper, we consider the problem of packing circles into a larger containing circle.

The problem of packing circles into a larger containing circle has been shown to be NP-hard [1–3], it is unlikely that there exists a polynomial time algorithm to solve it optimally. Hence people turn to obtain heuristic algorithm that is not absolutely rigorous, but is of high speed, reliability, and efficiency.

Packing problems have been studied extensively [4–7]. Here, we briefly give a survey of the problem of packing circles. Several authors have studied the problems of packing equal circles [8–10]. For simple cases of this problem, it is possible to obtain optimal solutions based on lattice patterns [11]. However, for complex cases, we have to depend on fast heuristic algorithms to generate

* Corresponding author.

E-mail address: dfzhang@xmu.edu.cn (D.-f. Zhang).

© 2003 Elsevier Ltd. All rights reserved.

approximate solutions, such as Huang and Kang [12] presented a heuristic quasi-physical strategy for solving disks packing problem.

A few authors have studied the problems of packing different-sized circles as well. During the past decade, some heuristic methods for this problem have been presented. Huang and Zhan [13] presented a fast quasi-physical algorithm, and the work path of this algorithm is to find natural phenomena in the physical world equivalent to the original mathematical problem, and then to observe the evolution of matter in it. Hochbaum and Maass [2] reported a very clever polynomial time approximate algorithm. George et al. [11] developed several heuristic procedures based on a variety of solution building rules that emulate a certain process of packing circles into a rectangle container. Huang and Xu [14] gave a quasi-physical personification algorithm based on combining the quasi-physical approach with the personification strategy. Recently, Wang et al. [3] described an improved algorithm, on the base of the quasi-physical approach, a quasi-human strategy is then proposed to trigger a jump for a stuck object in order to get out of local minima.

Simulated annealing [15] is a general stochastic search algorithm for combinatorial optimization problems. In contrast to other local search algorithms, it provides more opportunities to escape from local minima. However, it often costs too much time for finding a solution, this situation prevents it from being applied to many practical problems. Tabu search (TS) [16] is a local search metaheuristic which relies on specialized memory structures to prevent cycling and enhance diversification. TS has proved remarkably powerful in finding high-quality solutions to computationally difficult combinatorial optimization problems drawn from a wide variety of applications [17]. In this paper, by combining the merits of simulated annealing and TS, we develop an effective hybrid algorithm for the problem of packing circles into a larger containing circle.

The rest of this paper is organized as follows. In Section 2 we give a clear mathematical formulation for the problem of packing circles. In Section 3 we construct a special neighborhood, briefly introduce simulated annealing (SA), and design a tabu strategy, and then a hybrid algorithm is developed. Computational results are described in Section 4. Conclusions are summarized in Section 5.

2. Mathematical formulation of the problem

Given a larger containing circle and \( n \) circles of different sizes (or same sizes), here, \( n \) is a positive integer, we shall ask if these circles can be packed into the larger containing circle without overlapping one another. This problem is stated formally as follows [14].

Take the origin of two-dimensional Cartesian coordinate system at the central point of the larger containing circle with the radius of \( R_0 \) (See Fig. 1.). The coordinates of the center of the \( i \)th circle is denoted by \((x_i, y_i)\), the radius of the \( i \)th circle is \( R_i \), we ask if there exist a set of real numbers \((x_1, y_1, \ldots, x_i, y_i, \ldots, x_n, y_n)\), such that

\[
\sqrt{x_i^2 + y_i^2} \leq R_0 - R_i, \quad i, j = 1, 2, \ldots, n, \quad i \neq j.
\]

\[
\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \geq R_i + R_j,
\]

If such real numbers exist, then please give them.
Definition 1. At any moment, if the positions of \( n \) circles are fixed, we call them a configuration, denoted by \( X = (x_1, y_1, \ldots, x_i, y_i, \ldots, x_n, y_n) \).

For any given configuration \( X = (x_1, y_1, \ldots, x_i, y_i, \ldots, x_j, y_j, \ldots, x_n, y_n) \), it is said that two circles \( i, j \) \( (i \neq j) \) embed each other, if
\[
\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} < R_i + R_j.
\]
Otherwise, the two circles do not embed each other. If the two circles embed each other, then the embedding depth \( d_{ij} \) between the \( i \)th circle and the \( j \)th circle is
\[
d_{ij} = R_i + R_j - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}.
\]
Otherwise, \( d_{ij} = 0 \). Similarly, it is said that the \( i \)th circle and the larger containing circle embed each other, if
\[
\sqrt{x_i^2 + y_i^2} > R_0 - R_i.
\]
Otherwise, \( d_{0i} = 0 \). Similarly, it is said that the \( i \)th circle and the larger containing circle embed each other, then the embedding depth \( d_{0i} \) between the \( i \)th circle and the larger containing circle is
\[
d_{0i} = R_i - R_0 + \sqrt{x_i^2 + y_i^2}.
\]
Otherwise, \( d_{0i} = 0 \).

By the quasi-physical approach [14], we convert the problem of packing circles to an optimization problem that minimized on the following potential energy function:
\[
U(X) = U(x_1, y_1, x_2, y_2, \ldots, x_n, y_n) = \sum_{j=1}^{n} U_j. \tag{1}
\]
Here, $U_j$ is the extrusion elastic potential energy of the $j$th circle and can be calculated as

$$U_j = \sum_{i=0, i \neq j}^{n} u_{ij}, \quad j = 1, 2, \ldots, n.$$  

$u_{ij}$ is the extrusion elastic potential energy between two smooth elastic circles and is proportional to the square of their mutual embedding depth

$$u_{ij} = kd_{ij}^2, \quad i, j = 0, 1, \ldots, n, \quad i \neq j.$$  

Generally, $k = 1$. Clearly, $U(X) \geq 0$. Hence the problem of packing circles has been converted to an optimization problem on the known potential energy function (1), that is to say, a configuration $X^* = (x_1^*, y_1^*, x_2^*, y_2^*, \ldots, x_n^*, y_n^*)$ of the potential energy function should be found. Therefore, if $U(X^*) = 0$, then $X^*$ is one solution for the problem, whereas if $U(X^*) > 0$, then the problem has no solution.

3. The proposed hybrid algorithm

In this section, we first present a heuristic strategy for constructing a special neighborhood. Then briefly discuss SA and design a tabu list. At last, a hybrid algorithm is developed.

3.1. Neighborhood structure

**Definition 2.** For any given configuration $X = (x_1, y_1, \ldots, x_i, y_i, \ldots, x_n, y_n)$, we call $X' = (x_1, y_1, \ldots, x'_i, y'_i, \ldots, x_n, y_n)$ as a neighboring configuration of $X$, where the position of the $i$th circle in the configuration $X'$ is randomly generated and different from that of the $i$th circle in the configuration $X$, and the positions of other circles in the configuration $X'$ are the same as those of $X$. The set of all these neighboring configurations of $X$ is called a neighborhood of $X$, denoted by $N(X)$.

By Definition 2, there exist many configurations in $N(X)$ because the position of the $i$th circle in the configuration $X'$ is randomly generated. Clearly, the range of search is too large, it can easily result in blind search and waste too much time, so the efficiency of computation is very low. According to these analyses, in order to make iterative process converge fast, it is necessary to reduce search space. Therefore, how to generate $X'$ for further constructing the neighborhood is very crucial for enhancing the efficiency of the annealing process. According to the characteristics of the problem of packing circles, we present the following heuristic strategy for generating $X'$ by simulating the physical moving process of circles.

For any given $X = (x_1, y_1, \ldots, x_i, y_i, \ldots, x_n, y_n)$, $X' = (x_1, y_1, \ldots, x'_i, y'_i, \ldots, x_n, y_n)$ can be determined by the vector sum of embedding depth of the $i$th circle. Here, we regard the embedding depth as a vector. Thus, the displacement of the $i$th circle is the module of the vector sum of its embedding depth. The moving direction of the $i$th circle is the direction of the vector sum of the extrusion elastic forces acted on it. In detail, the way of generating $X'$ is as follows:
For a given $X = (x_1, y_1, \ldots, x_i, y_i, \ldots, x_n, y_n)$, we consider the change of the position of the $i$th circle under the extrusion elastic forces. With the help of Fig. 2, we have

$$\frac{x_i - x_j}{dx_{ij}} = \frac{D_{ij}}{d_{ij}},$$
$$\frac{y_i - y_j}{dy_{ij}} = \frac{D_{ij}}{d_{ij}},$$

(2)

where $D_{ij}$ denotes the distance from the center of the $i$th circle to that of the $j$th circle, $d_{ij}$ is the same as previous definition, $dx_{ij}$ is the projection of $d_{ij}$ in the horizontal axis, $dy_{ij}$ is the projection of $d_{ij}$ in the vertical axis. From (2), we have

$$dx_{ij} = \frac{x_i - x_j}{D_{ij}} d_{ij},$$
$$dy_{ij} = \frac{y_i - y_j}{D_{ij}} d_{ij}.$$

Specially, we have

$$dx_{i0} = -\frac{x_i}{D_{0i}} d_{0i},$$
$$dy_{i0} = -\frac{y_i}{D_{0i}} d_{0i},$$

where $D_{0i}$ denotes the distance from the center of the larger containing circle to the center of the $i$th circle, $d_{0i}$ is the same as previous definition, $dx_{i0}$ is the projection of $d_{0i}$ in the horizontal axis, $dy_{i0}$ is the projection of $d_{0i}$ in the vertical axis. Therefore, the next position of the $i$th circle is

$$x'_i = x_i + dx_{ij} + dx_{i0},$$
$$y'_i = y_i + dy_{ij} + dy_{i0},$$

$j \neq i, \ i = 1, 2, \ldots, n.$
SA( )
Begin
Generate an initial configuration $X_0$, let $X = X_0$;
Obtain an initial temperature $T_0$, let $T = T_0$;
While the stop criterion is not yet satisfied
{   Perform the following loop $M$ times
   {   Randomly pick one configuration $X'$ from $N(X)$;
       Let $\Delta U = U(X') - U(X)$;
       If $\Delta U \leq 0$ then set $X = X'$;
       If $\Delta U > 0$ then set $X = X'$ with probability $\exp(-\Delta U/T)$;
   }
   Set $T = \alpha T$;
}
Return $X$;
End.

Fig. 3. The SA algorithm.

With the help of the configuration $X = (x_1, y_1, \ldots, x_i, y_i, \ldots, x_n, y_n)$, we obtain a configuration $X' = (x_1, y_1, \ldots, x'_i, y'_i, \ldots, x_n, y_n)$ by simulating the physics moving process of the $i$th circle. Similarly, we can construct other neighboring configurations of $X$. Clearly, the number of the configurations in $N(X)$ is $n$. Since the way of generating $X'$ is determinate, the range of search is significantly reduced. Therefore, SA based on this neighborhood can avoid blind search to some extent, and it allows iterative process to converge fast.

3.2. SA

SA introduced by Kirkpatrick [15] is based on the analogy between the annealing of solids and the solving of large-scale optimization problems. Solutions in a combinatorial optimization problem are equivalent to the states of a physical system, and the cost of a solution is equivalent to the energy of a state. In the process of search, SA accepts not only better but also worse neighbor solutions with a certain probability. The temperature determines the probability of accepting worse solutions. The probability of accepting a worse solution is large at higher temperatures. As the value of the temperature declines, the probability of accepting worse solutions also decreases as well. This feature implies that SA, in contrast to other local search algorithms, has more opportunity to escape from local minimum trap. The annealing process first raises the system temperature to a sufficiently high level so that the system can be transferred to all possible states. The temperature is then maintained for a certain time at each level and is gradually decreased until the desired state is attained.

Generally, one can use the annealing procedure (see Fig. 3.) as follows to obtain a solution for optimization problem [15,18,19].

Since the performance of SA is significantly impacted by the choice of $T_0$, $\alpha$ etc., these parameters should be selected rationally by referring to [18]. In the problem of packing circles, we choose $T_0 = n/20.0$, $\alpha \in [0.9, 0.95]$, the stop criterion can refer to $(U(X) < 10^{-6}$ or $T < 0.0001)$, and $M = n$. 
3.3. Tabu list strategy

For SA, when the temperature $T$ tends to zero at the end of the process, the probability of accepting worse neighboring configurations is approximately zero. In that case, SA loses its feature to accept worse configurations, thereby becoming identical to other local search algorithms. In this way the optimization process may get stuck in a local optimum. In addition, the way of generating $X'$ may lead SA to get stuck in a local optimum during the course of execution as well. Under this circumstance, a promising approach is to put forward some good heuristic strategies for jumping out of the local minimum trap by taking the calculating point out of local minima and placing it in a position with better prospects. Then new SA process can be carried out over again.

In this paper, when search process gets stuck in local minima, it selects a circle with maximum $U_i/R_i$ to jump out of local minima. However, during the process of jumping out of local minima, cycling is possible, namely, a circle may be selected repeatedly. In order to improve the efficiency of the exploration process and avoid cycling, we make use of tabu list strategy [20] of TS. During the process of jumping out of local minima, TS keeps track not only of local information but also of some information related to the exploration process. This systematic use of the stored information is the essential feature of TS. The search process uses this information to avoid cycling and explore new directions in the neighborhood. During the process of each jumping out of local minima, we select one circle, then randomly put it to a new position. However, these circles that have been put randomly during the last $L$ times jumping time are considered tabu, which should be avoided to select in the next jumping time. The stored information, represented by a data structure named tabu list, is based on the last events and will partly prevent cycling and enhance diversification. To realize this data structure, an array, defined by JumpTime $[i]$, $i = 1, 2, \ldots, n$ which record the time of jumping out of trap when the $i$th circle is selected. In current time, one circle is tabu or not depends on whether it meets the following tabu condition. If JumpTime $[i] + L \geq \text{currentTime}$, then this circle is tabu in the next jumping time, otherwise it is not tabu. For example, the $i$th circle is selected in time 179, then the value of JumpTime $[i]$ is modified as 179, others do not change. If $L = 3$, then the next three times jumping time, namely time 180, 181, 182, the $i$th circle should be tabu. Due to the circle selected to jump out of the trap is limited by the information stored in the tabu list. Therefore, the length of the list plays an important role in the jumping process. Nevertheless, if $L$ is too long, the jumping process can be inhibited, whereas if it is too short, cycling cannot be avoided. The experimental evidence has shown that $L = 3$ is appropriate for our testing problem instances. It is noted that the tabu list in this paper is a given parameter, its aim is to prevent cycling. We also tried other classical tabu List, for example, with variable neighborhood size, and observed little difference in the results, however, the tabu list in this paper is simple, and we still use it in our experiments.

3.4. Statement of the hybrid algorithm

In order to implement the hybrid algorithm on a PC, a program is written, which can be used to verify the proposed algorithm. Let the set of circles be $C$. Integrating previous tabu list strategy into SA, the hybrid algorithm (SATS) for the problem of packing circles can be given as follows:
SATS()
Begin
Do
{   Running the SA procedure: SA();
   If \( U > 10^{-6} \)
   {   Flag=1;
       While(Flag=1)
       {   Selecting circle \( c \) with the maximal \( U_{/c'}, c \in C \);
           If JumpTime[\( c \)+L] \geq\text{CurrentTime};
               \( C = C - \{c\} \);
           Else\{  \( c \) is selected, and then it be randomly put a new position;
               JumpTime[\( c \)] =\text{CurrentTime};  \text{Flag=0};  \}
       }
   }\}
CurrentTime ++;
}while (stop criterion is not reached);
End.

Fig. 4. The hybrid algorithm SATS.

From Fig. 4, SATS reasonably combines the merit of SA and TS. SA provides an initial solution for TS, while TS is to prevent cycling during jumping out of local minimum and obtain a new initial solution with better prospects for SA. Then new SA process can be carried out over again.

4. Computational results

To test the performances of the hybrid algorithm, we compare SATS with QuasiPQuasiH [3], TS and SA. In comparing different algorithms for solving problems of packing circles, both speed and quality of solution are important measures of their performances [3]. In this paper, quality of solution of both algorithms is same, namely \( U \leq 10^{-6} \), the calculation is stopped, and then we only compare these algorithms from computational speed. All these algorithms have been implemented with C language on a Dell GX260 to perform large amounts of calculation with benchmark instances. These benchmark instances with different scale are confined to solvable instances [3], for those instances in which no feasible solution exists, all these algorithms will not stop. The nine benchmark problem instances in Table 1, which include five instances of packing equal circles and four instances of packing unequal circles, are typical representatives. \( I_1 \) to \( I_4 \) are cited from [3], \( I_5, I_6 \) are cited from [14], \( I_7-I_9 \) are cited from [10]. Here, \( I_2-I_9 \) are harder benchmark instances. Due to an initial configuration being generated randomly, each benchmark instance is randomly run five times for each algorithm in order to have a more meaningful and rational comparison. The execution time, the
average execution time ($\bar{t}$) and the geometric morphologies of the solutions are shown in Table 2. Here, $I$ denotes a benchmark problem instance, other symbols are as before.

From the results, as shown in Table 2, we observed that the results obtained by SATS are much better than those obtained by SA and TS applied alone, and observed little difference in speed between SATS and the QuasiPQuasiH algorithm for $I_1$, $I_2$, $I_4$; For $I_3$, $I_5$–$I_9$, the speed of SATS is five times that of the QuasiPQuasiH algorithm; For rather difficult instances, for example, $I_6$, $I_9$, such an increase in speed is especially notable.

### 5. Conclusions

In this paper, we reasonably combined simulated annealing with tabu search to develop a hybrid algorithm for the problem of packing circles. The key of this algorithm lies in a powerful means for getting out of local minima. Simulated annealing was introduced to escape from local optima with probability mechanism. Tabu search is mainly used for preventing cycling and enhancing diversification. The computational results based on some benchmark instances showed that, the hybrid algorithm was very effective and robust, and almost outperform TS, SA and QuasiPQuasiH for all benchmark instances.

Our research was partly motivated by industrial applications, such as the optic-fiber communication and the transportation of the steel pipes in shipping containers, SATS has been applied to a practical container packing software designed by us. In addition, SATS is easily extendable to other problems of packing circles in other bounded space (for example, rectangle or triangle). We hope to find highly efficient algorithm for other packing problem of even greater practical significance in the near future.

**Acknowledgements**

The authors would like to thank the referees for their helpful comments and suggestions that help to improve this paper.
Table 2
Comparisons of TS, SA, QuasiPQuasiH and SATS

<table>
<thead>
<tr>
<th></th>
<th>Execution time of five times and $\bar{t}$ (s)</th>
<th>The geometric morphologies of the solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TS</td>
<td>SA</td>
</tr>
<tr>
<td>1</td>
<td>0.00, 0.00, 0.00, 0.00, 0.00</td>
<td>0.00, 0.00, 0.00, 0.00, 0.00</td>
</tr>
<tr>
<td>2</td>
<td>73.74, 51.37, 91.98, 27.36, 37.81</td>
<td>79.62, 75.77, 39.23, 259.18, 120.99</td>
</tr>
<tr>
<td>3</td>
<td>2.15, 5.66, 17.58, 3.69, 9.73</td>
<td>4.51, 3.13, 1.92, 1.26, 22.69</td>
</tr>
<tr>
<td>4</td>
<td>4.45, 8.27, 2.86, 81.92, 6.48</td>
<td>5.11, 8.08, 40.66, 15.22, 9.89</td>
</tr>
<tr>
<td>5</td>
<td>0.9, 1.35, 0.67, 1.04, 3.00</td>
<td>0.82, 0.22, 0.22, 0.22, 0.22</td>
</tr>
<tr>
<td>6</td>
<td>24.74, 17.69, 58.91, 97.82, 34.01</td>
<td>54.78, 9.73, 40.77, 28.52, 4.78</td>
</tr>
<tr>
<td>7</td>
<td>0.63, 0.80, 0.77, 0.29, 0.25</td>
<td>0.16, 0.22, 0.16, 0.22, 0.05</td>
</tr>
<tr>
<td>8</td>
<td>3.51, 7.02, 4.82, 7.71, 1.96</td>
<td>0.55, 0.55, 0.77, 0.71, 2.03</td>
</tr>
<tr>
<td>9</td>
<td>15.99, 33.30, 38.9, 138.52, 34.67</td>
<td>9.78, 9.23, 11.87, 2.42, 10.88</td>
</tr>
<tr>
<td></td>
<td>$\bar{t}$ = 0</td>
<td>$\bar{t}$ = 0.0</td>
</tr>
<tr>
<td></td>
<td>$\bar{t}$ = 123.25</td>
<td>$\bar{t}$ = 96.26</td>
</tr>
<tr>
<td></td>
<td>$\bar{t}$ = 7.76</td>
<td>$\bar{t}$ = 6.70</td>
</tr>
<tr>
<td></td>
<td>$\bar{t}$ = 20.79</td>
<td>$\bar{t}$ = 15.79</td>
</tr>
<tr>
<td></td>
<td>$\bar{t}$ = 46.63</td>
<td>$\bar{t}$ = 27.72</td>
</tr>
<tr>
<td></td>
<td>$\bar{t}$ = 0.55</td>
<td>$\bar{t}$ = 0.16</td>
</tr>
<tr>
<td></td>
<td>$\bar{t}$ = 5.00</td>
<td>$\bar{t}$ = 0.92</td>
</tr>
<tr>
<td></td>
<td>$\bar{t}$ = 72.27</td>
<td>$\bar{t}$ = 8.84</td>
</tr>
</tbody>
</table>
References