Variable neighborhood search for the 2D orthogonal packing *

Yuri Kochetov, Yuliya Velikanova

Abstract

We develop the VNS algorithm for the two dimensional rectangular packing problem. Our approach is based on a Transitive Closure Graphs representation for feasible packing. The VNS uses three neighborhoods to modify these graphs and two empirical methods to improve the packing. Computational results for test instances from the benchmark library GSRC are discussed.

1 Introduction

In this paper we consider the following orthogonal packing problem on two dimensional plane. Let $I = \{1, \ldots, n\}$ be a finite set of rectangular items whose width and height are denoted by $w_i > 0, h_i > 0$, $i \in I$. Let (x_i, y_i) denote Cartesian coordinates of the bottom-left corner of item $i \in I$. Packing P of items on the plane is feasible if side of each item is parallel to a coordinate axis and there are no two items overlap. Rotations of items by 90° are not allowed. We wish to find a feasible packing with minimal area of bounding rectangle, i.e. minimize

$$F(P) = \max_{i \in I}(x_i + w_i) \max_{i \in I}(y_i + h_i).$$

It is NP-hard combinatorial problem [1] which arises in VLSI design.

To solve this problem we use idea of the Variable Neighborhood Search algorithm (VNS) [2]. Our approach is based on a Transitive Closure Graphs representation (TCG) for feasible packing [4]. The VNS uses three neighborhoods to modify these graphs and two empirical methods to improve the packing. The first method deals with mirror reflections of packing with respect to one or two axes. It can produce a packing with better value of the objective function and open new directions for local descent algorithm. The second method uses an auxiliary objective function to overcome the plateaus of the search landscape. This auxiliary function indicates the length of touching for the current packing and bounding rectangle. Our computational results for test instances from the benchmark library GSRC [6] show that VNS finds near optimal solutions with small relative deviation from the primitive lower bound.

*The research is supported by Russian Foundation for Basic Research, project 04-07-90096.

2 Representation of packing by TCG

The TCG representation describes geometrical relations among items by two directed acyclic graphs [4]: horizontal transitive closure graph C_h and vertical transitive closure graphs C_v. For this representation, solution space is finite and the best solution corresponds to an optimal packing.

2.1 Encoding

Let us show how feasible packing P is encoded by graphs C_h and C_v.

Definition 1 For two non-overlapping items i and j, the item i is said to be horizontally related to item j and denoted by $i \rightarrow j$ if i is on the left side of j and their projections on the y axis overlap.

Definition 2 For two non-overlapping items i and j, the item i is said to be vertically related to item j and denoted by $i \uparrow j$ if i is on the bottom side of j and their projections on the x axis overlap.

Definition 3 For two non-overlapping items i and j, the item i is said to be diagonally related to the item j if i is on the left side of j and their projections on the both axes do not overlap.

We treat a diagonal relation between i and j as a horizontal one unless there exist a chain of vertical relations from i to j. Otherwise we treat the diagonal relation as a vertical relation. Hence, we have a vertical or a horizontal relation for each pair of items in the feasible packing.

Graphs C_h and C_v are defined as follows. The set of vertices in both graphs is the set of items. The weight of vertex i is w_i for C_h and h_i for C_v. If $i \rightarrow j$ we include arc (i, j) into the arc set for graph C_h. Similarly, we include arc (i, j) into the arc set for C_v if $i \uparrow j$. Note that for each pair of items there exist exactly one arc either in C_h or in C_v.

2.2 Decoding

Let us add two additional vertices, source s and sink t with zero weights into the both graphs C_h and C_v. Let $E^+(i)$ be a set of incoming arcs and $E^-(i)$ be a set of outgoing arcs for vertex i. We construct arc (s, i) for graphs C_h and C_v for each vertex i with $E^+(i) = \emptyset$. Similarly, we add arc (i, t) for each vertex i with $E^-(i) = \emptyset$. Now the coordinate (x_i, y_i) of the bottom-left corner of item i can be found by the longest path algorithm. More exactly, $x_i = L_h(i), y_i = L_v(i)$ where

$$L_h(i) = \begin{cases} 0, & i = s; \\ \max\limits_{j \in E_h^+(i)} (L_h(j) + w_j), & i \neq s \end{cases}$$

$$L_v(i) = \begin{cases} 0, & i = s; \\ \max\limits_{j \in E_v^+(i)} (L_v(j) + h_j), & i \neq s \end{cases}$$

The area of packing is given by $L_h(t)L_v(t)$.

3 Neighborhoods

Let us consider three neighborhoods for feasible packing P [3, 4].

Definition 4. An arc (i, j) is called the reduction arc, if there is no path from i to j except the arc (i, j) itself.

Definition 5. Feasible packing P' is called neighboring solution for P if its TCG representation (C'_h, C'_v) can be obtained from (C_h, C_v) by one of the following operations:

— Reverse a reduction arc in C_h or C_v.
— Move a reduction arc from one graph to another.
— Swap two vertices with their weights in graphs C_h and C_v simultaneously.

The set of neighboring solutions generated by the Reverse operator we denote by $N_r(P)$. Similarly, $N_m(P)$ is the set of neighbors generated by the Move operator and $N_s(P)$ is the set of neighbors generated by the Swap operator.

The Reverse and Move operators correspond to changing the geometrical relations of the two touching items in the packing.

To reverse a reduction arc (i, j) in the transitive closure graph we should:

— delete arc (i, j) from the graph;
— add reverse arc (j, i) to the graph;
— for each pair of vertices $k \in E^+(j) \cup \{j\}$ and $l \in E^-(i) \cup \{i\}$, we check whether arc (k, l) exists. If it is not, we add this arc and delete it from another graph.

To move a reduction arc (i, j) from the one transitive closure graph G to other G', we should:

— delete arc (i, j) from the graph G;
— add arc (i, j) to the graph G';
— for each pair of vertices $k \in E^+(i) \cup \{i\}$ and $l \in E^-(j) \cup \{j\}$, we check whether arc (k, l) exists in the graph G'. If it is not, we add this arc to G' and delete it from G.

The Swap operator does not change the topology of TCG. Note that TCG is closed under the Reverse, Move, and Swap operators.

Theorem For arbitrary feasible packings P_1 and P_2 and their TCG representations (C_h, C_v) and (C'_h, C'_v), we can reach (C_h, C_v) from (C'_h, C'_v) by at most $O(n^2)$ steps using the Reverse, Move, and Swap operators.

4 Local descent algorithm

Standard local descent algorithm starts from a feasible solution. It moves iteratively from one solution to a better neighboring solution and terminates at a local optimum. The number of steps of the algorithm and value of the local optimum are determined by the starting point, neighborhood, and pivoting rules. One of the open question in this field is the running time of the algorithm. Is the number of steps polynomially bounded? It is clear that our local search problem belongs to the class PLS [5]. Is it PLS-complete? Can we find an example of the packing problem when the standard local descent algorithm spends exponential number of steps to reach local optimum?

Another interesting question is the quality of local optimum or average relative deviation from the lower bound. According to idea of VNS we study two strategies to use the neighborhoods N_r, N_m, N_s:

— find local optimum for the first neighborhood, use it as a starting point for local descent and find local optimum for the second neighborhood, and then apply local descent for the third neighborhood; repeat this stage until we get local optimum with respect to all neighborhoods;

— change neighborhood after k steps of the local descent.

We compare these strategies for different $k \geq 1$ on random test instances and observe that the first strategy is better. We use it in our VNS algorithm.

Two pivoting rules are the most popular in the local search heuristics: the best improvement rule and the first improvement rule. Our computational study shows that the first improvement rule produces better local optimum from random starting points, spend less efforts per step, but require more steps than the best improvement rule for the neighborhoods N_s, N_m. For neighborhood N_r the best improvement rule is better. Further we use the first improvement for N_s, N_m and the best improvement for N_r neighborhood.

Figure 1. Change the orientation of C_h and C_v

4.1 Change of orientation

To find better local optimum we apply two empirical methods. The first one changes the orientation of all arcs in graph C_h or C_v. This procedure corresponds to rotation of the packing by 180° and compressing it to the bottom-left corner. We can reduce the packing area by this procedure and discover better neighboring solutions for the same neighborhoods. In Figure 1 we illustrate these rotations. Graph $C^{-1} = (V, E^{-1})$ is obtained from graph $C = (V, E)$ by changing the orientation of every arc $(i, j) \in E$.

4.2 Auxiliary objective function

For feasible packing P let us consider two sets

$$B_x = \{i \in I \mid x_i + w_i = \max_{j \in I}(x_j + w_j)\},$$

$$B_y = \{i \in I \mid y_i + h_i = \max_{j \in I}(y_j + h_j)\}.$$

These sets contain the items which define the width and height of the packing.

Figure 2. Auxiliary objective function

The auxiliary objective function is defined as follows

$$f(P) = \sum_{i \in B_x} h_i + \sum_{i \in B_y} w_i.$$

Function $f(P)$ indicates the length of touching for the packing and the bounding rectangle without axes x and y, see Figure 2.

For this example, $|B_x| > 1, |B_y| > 1$. So, we can not improve the packing by moving one item only. But we can improve the value of auxiliary objective function and reduce the packing area after several steps. Now local descent algorithm uses two objective functions. First of all, it tries to find a neighboring solution with smaller packing area. If it does not exist, algorithm tries to find a neighbor with better value of the auxiliary objective function. It stops in local optimum with respect to both objective functions.

In Figure 3 we present the influence of these two empirical methods. Deviation from the lower bound which is area of all items, grows rapidly for the standard local descent algorithm. Changing the orientation of the digraph C_h and C_v gives us some improvements. But the most effect we get by combining two methods. Average deviation decreases for such combination. We denote the improved variant of local descent algorithm by ILD.

5 Variable Neighborhood search

Let $N^k(P)$ be a set of feasible solutions which are derived from solution P by applying k times one of operators Reverse, Move, or Swap. Basic scheme of VNS can be present as follows.

Figure 3. Influence of empirical methods

Basic VNS

1. Select an initial solution P and stoping condition.
2. Repeat until the stopping condition met:
   2.1 Set $k := 1$;
   2.2 Repeat the following steps until $k = k_{max}$:
      (a) Select $P_1 \in N^k(P)$ at random;
      (b) Apply ILD algorithm and get local optimum P_2;
      (c) If $F(P_2) < F(P)$, then $P := P_2, k := 1$, else $k := k + 1$.

We randomly select initial solution and use $k_{max} = 15$.

6 Computational experiments

The developed VNS algorithm is coded in Pascal for Delphi environment and tested on examples from GSRC Floorplan Benchmark Suite[6]. The instances have dimensions $n = 100$ and 200. Typical behavior of ILD algorithm is presented in Figure 4. Notice that the algorithm faces with plateaus (horizontal lines) and overcomes them quickly.

Figure 4. Behavior of the ILD algorithm.

The final packing is shown in Figure 5, for $n = 100$. For all instances we observe average relative deviation from 4.5% to 5.7% if we call the ILD algorithm 15 times. For large instances, $n = 200$, we have got the best known packing with deviation 5.74% against 5.825% [6].

Figure 5. Final solution, n=100.

Acknowledgements

Authors thank Nenad Mladenović and Pierre Hansen for fruitful discussions and useful suggestions which help us to improve preliminary variant of the VNS algorithm. This research was supported by Russian Foundation for Basic Research, grant 04-07-90096.

References

[1] B. S. Baker, E. G. Coffman and R. L. Rivest. Ortoganal Packing in Two Dimensions. *SIAM J. Comput.,* 9(4): 846–855, 1980.

[2] P. Hansen and N. Mladenović. Variable Neighborhood Search: Principles and Applications. *European J. Oper. Res.,* 130: 449–467, 2001.

[3] J.-M. Lin and Y.-W. Chang. TCG–S: Orthogonal Coupling of P^* Representation for General Floorplans. *IEEE Trans. Computer–Aided Design,* 23(6): 962–967, 2004.

[4] J.-M. Lin and Y.-W. Chang. TCG: A Transitive Closure Graph-Based Representation for General Floorplans. *IEEE Trans. on Very Large Scale Integration Systems,* 13(2): 288–292, 2005.

[5] M. Yannakakis. Computational Complexity. In: E. Aarts, J. K. Lenstra, editors, *Local Search in Combinatorial Optimization*, pages 19–55. Chichester: Wiley, 1997.

[6] *GSRC Floorplan Benchmark Suite* http://www.cse.ucsc.edu/research/surf/GSRC/