Bin Packing Solution for Automated Packaging Application

Article *in* Applied Mechanics and Materials · December 2011

DOI: 10.4028/www.scientific.net/AMM.143-144.279

Applied Mechanics and Materials Vols. 143-144 (2012) pp 279-283
Online available since 2011/Dec/08 at www.scientific.net
© (2012) Trans Tech Publications, Switzerland
doi:10.4028/www.scientific.net/AMM.143-144.279

Bin Packing Solution for Automated Packaging Application

Yang Liu[1, 2 a] and Thanh Vinh Vo[1]

[1]Department of Information Technology, Vaasa University of Applied Sciences, Finland

[2]School of Electrical & Electronic Engineering, Hubei University of Technology, China

[a]yang.liu@puv.fi

Keywords: Bin packing, heuristic recursive algorithm, automated packaging, robot

Abstract. This paper introduces the implementation of a new heuristic recursive algorithm for bin packing solution used in automated packaging application. The theoretical method proposed in this paper is successfully implemented on a real ABB robot arm with some important improvements such as added rotation flexibility and removing an added product from the structure. The computational results on a class of benchmark problems have shown that this algorithm not only finds shorter height than the known meta-heuristic ones, but also runs in shorter time. The average running time is very suitable for such kind of automated packaging application.

Introduction

In automated packaging application, when a product is picked from the input box, the application always has to find for it a place in the output box to drop it. Moreover, the position to drop a product always has to be optimal so that the minimal amount of container bin is used. This is a well-known mathematical problem named "bin packing problem" [1]. The purpose of the algorithm is trying to pack objects with different dimensions into bigger fixed-size bins with minimum number of used bins. One variety of this problem is cutting material: given a big dimension material, it needs to be cut it into small pieces with highest material usable ratio.

The problem is known as "strongly NP-hard" (non-deterministic polynomial-time hard). NP-hard type belongs to the set of problem that is not sure if it can be solved in polynomial-time by machine. One relevant problem has been voted as one of Millennium Prize Problems [2] is "P versus NP", better known as "polynomial-time" versus "non-deterministic polynomial-time", in which we have to determine whether questions exist whose answers can be quickly checked, but which require an impossibly long time to solve by any direct procedure. There have been many solutions existing trying to solve the problem by using various method: heuristic, genetic, recursive, etc. Some of the simple and famous methods are best fit decreasing and first fit decreasing [3]. However, none of them can be claimed as perfect or best solution. Earlier research [4,5,6,7] show that varies algorithms can differ from optimal packing by as much as 70% and never suboptimal by more than 22%.

This product packaging problem even falls into the most challenging group:

- It is "online" - in contrast to "offline" problem, in which the program is given the set of objects it need to pack at the beginning and can pack them in any random order. In "online" problem, instead of having sets of objects in advance, they will be given one by one. In fact, the program has the sets of products it needs to pack. However, the next product is not available until the vision recognizes and the robot picks it successfully. As a result, products will come in unpredictable order. Therefore, the problem turns to be "online" bin packing problem.
- The solution has to be near real-time execution which is very challenging to solve "strongly NP-hard" problems.

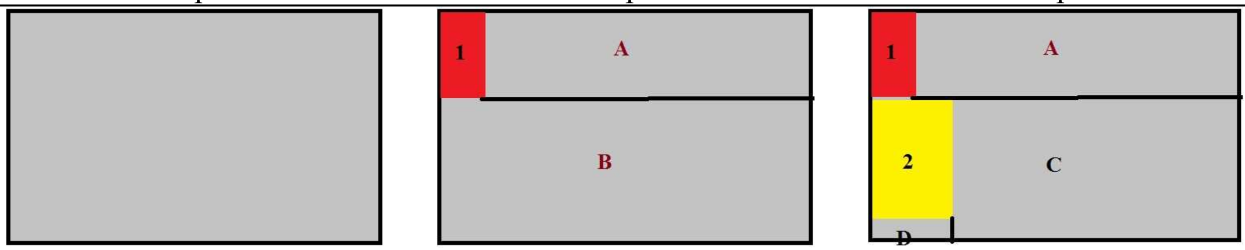
Implementation

The implementation is largely inspired by the algotithm presented in [8] which is mainly based on heuristic strategies and a recursive structure.

The theoretical method proposed in [8] is successfully implemented with some important improvements such as added rotation flexibility which means the robot can rotate products if necessary, and the ability to remove an added product from the structure. The picking/placing process might have problem, so the structure need to be able to cancel the added products. The computational results on a class of benchmark problems have shown that this algorithm not only finds shorter height than the known meta-heuristic ones, but also runs in shorter time. The average running time is $T(n)=\Theta(n^3)$, which is very suitable for this application. The recursive process is illustrated in Table 1 and described as follow:

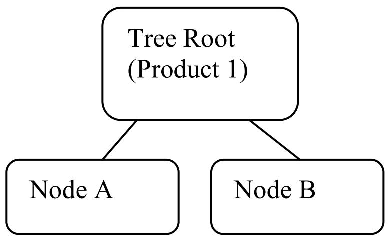
Table 1. Steps of recursive process

Step 1	Step 2	Step 3

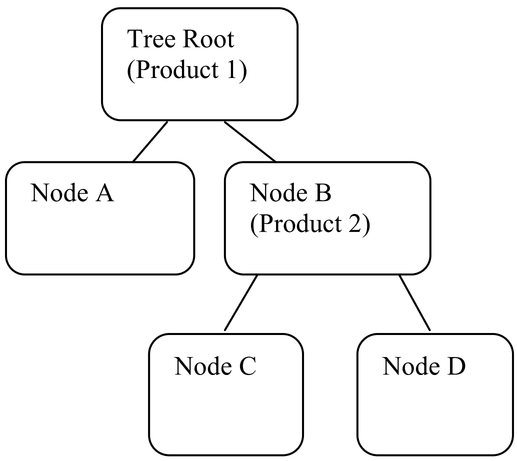


Step 1: Starting with an empty box, the tree only contains the root node.

Step 2: After inserting one product, the root node of the tree now holds the first product. At the same time, the remaining free area is separated into 2 smaller areas. Therefore, the tree structure looks like this:



Step 3: Keep adding another product. When inserting another product, the application first checks if the product can fit node A and if the product cannot, the application will check node B. If the product fits into node B, it is inserted into node B and B area is divided into 2 sub-areas similar with inserting the first product.



This procedure will be repeated again and again starting from the root. If the product cannot be inserted anymore, it means that a new bin needs to be used.

If neither area A nor area B can host the product, the application will try to rotate the product and insert it again. If no solution found after rotation it means that the product cannot be inserted. There are some cases when added product is required to be removed from the tree. The application sees the product, calculates coordinates (including current product's coordinates and its coordinates in the output bin) and sends coordinates to the robot so that the robot can go, pick it up and place it to the output bin. At the same time, proximity sensor will check if the picking process has been carried out successfully. It the process fails, the bin packing application has to remove the newly added product from the tree. The newly added product is always hosted in branch of the root. When the product is added, it is always assigned an ascending identification number. If one inserted product needs to be removed, the application will goes recursively from the root node to find the node which hosts the product with same identification number, then set that node to null value. Since the deleted node is a branch node, it does not affect the root structure.

Testing

The algorithm has been successfully implemented on a real ABB robot arm to perform the following real world tests. The testing setup is shown in Figure 1. The container bin has a fixed size of 700 x 300 mm and the products are rectangular boxes which have random size of ranging from [0 →100] x [0 →50] mm. The testing conditions are listed in Table 2. A sample of calculated bin packing solution is shown in Figure 2. The test of packing random sized products has been repeated for 10 times and the results of occupation ratio are listed in Table 3 which shows better performance than declared in [7]. The time required to execute the routine is around 4~5 ms for a large set of products.

Fig.1. The testing setup

Table 2. Testing conditions

Testing conditions for packing random size boxes	
Max width (mm)	200
Max height (mm)	100
Container width (mm)	700
Container height (mm)	300
Number of boxes trying to insert	200

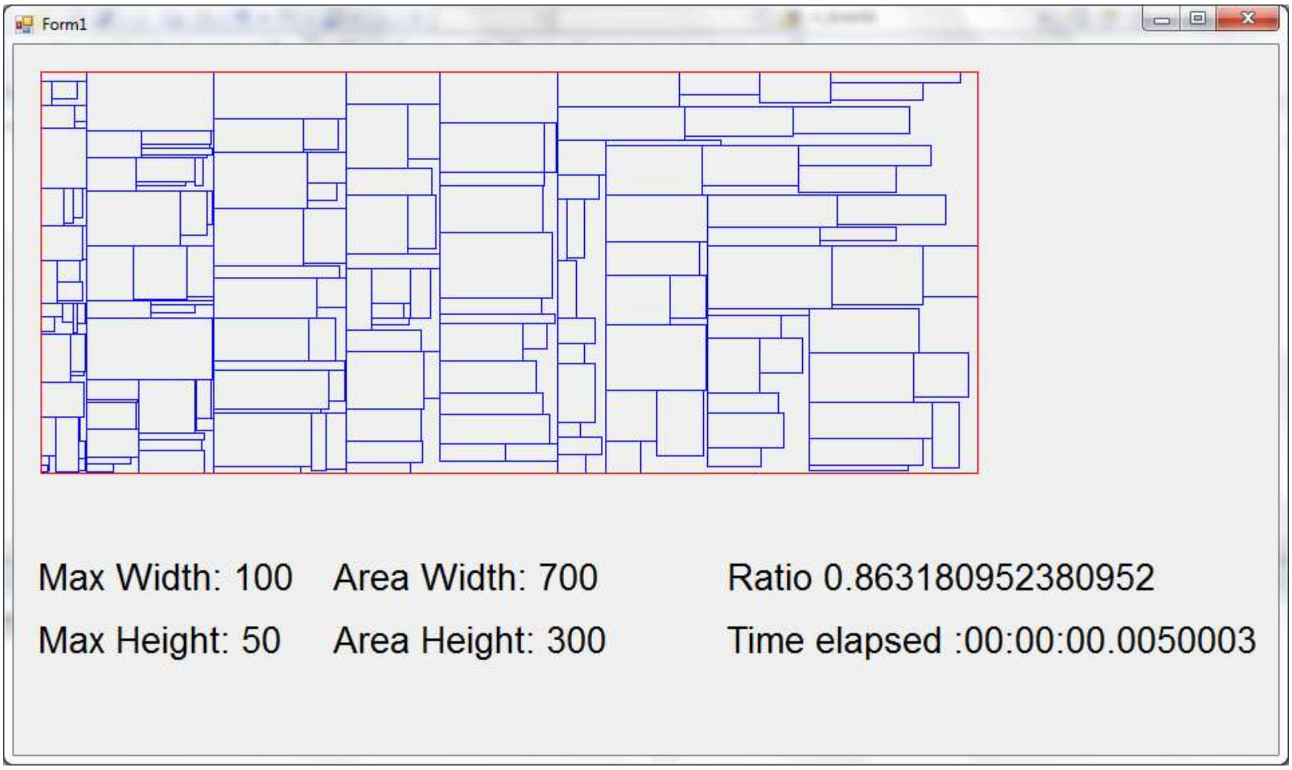


Fig.2. A calculated bin packing solution

Table 3. Testing results

Testing sequence no.	Occupation ratio
1	0.92840
2	0.93424
3	0.94944
4	0.95437
5	0.95132
6	0.94932
7	0.96235
8	0.93594
9	0.95562
10	0.93786
Average	0.94589

Summary

In this paper, a familiar binary tree structure is implemented based on a new heuristic recursive algorithm to solve a complex problem relatively well. The theoretical method proposed in this paper is successfully implemented on a real ABB robot arm with some important improvements such as added rotation flexibility and removing an added product from the structure. Based on the test result, this solution has been proved to be good enough to solve the bin packing problem within strict timing requirement of automated packaging application.

Acknowledgements. The author gratefully acknowledges the financial support of this research from China Hubei Provincial Science & Technology Department (Project No. 2010BFA007), and Vo Thanh Vinh for his valuable work in this project.

References

[1] S. Martello, D. Pisinger and D. Vigo: The three-dimensional bin packing problem. Operations Research 48(2), 256-267 (2000)

[2] Clay Mathematics Institute: *Millennium Prize Problems*. (Accessed 2010) Available at: http://www.claymath.org/millennium/

[3] S. Albers and M. Mitzenmacher: Average-case analyses of first fit and random fit bin packing. Random Structures & Algorithms 16(3), 240–259 (2000)

[4] M. R. Garey, R. L. Graham and J. D. Ullman: An analysis of some packing algorithms. In: *Combinatorial Algorithms*. Algorithmics Press, New York (1973) 39-47

[5] D. S. Johnson: Approximation algorithms for combinatorial problems. Journal of Computer and System Sciences 9(3), 256-278 (1974)

[6] E. G. Coffman Jr., M. R. Garey and D. S. Johnson: Approximation algorithms for bin-packing: An updated survey. In G. Ausiello, M. Lucertini and P. Serafini, eds. : *Algorithm Design for Computer System Design*. Springer Verlag, Vienna (1984) 49-106

[7] P. Hoffman: *The Man Who Loved Only Numbers: The Story of Paul Erdos and the Search for Mathematical Truth*. Hyperion, New York (1998)

[8] D. Zhang, Y. Kang and A. Deng: A new heuristic recursive algorithm for the strip rectangular packing problem. Computers & Operations Research 33(8), 2209-2217 (2006)

Electrical Information and Mechatronics and Applications
10.4028/www.scientific.net/AMM.143-144

Bin Packing Solution for Automated Packaging Application
10.4028/www.scientific.net/AMM.143-144.279