# Metaheuristic approach to sequence product families in reconfigurable manufacturing systems[*]

**Ignacio Eguia, Jesús Racero**

Dpto. de Organización Industrial y Gestión de Empresas. Escuela Superior de Ingenieros. Universidad de Sevilla. Camino de los Descubrimientos s/n. 41092. Sevilla. ies@esi.us.es, jrm@esi.us.es, jmgs@esi.us.es

## Abstract

*This research aims at the development of a heuristic approach to select the best set of product families and their production scheduling in Reconfigurable Manufacturing Systems when exact methodologies cannot offer a solution, due to the complexity of the instances may requires an excessive computing time. For tackling the problem, a specific and quick heuristic variant of the Nearest Neighbour method has been developed and implemented. Solutions to the problem are significantly improved when they are used as part of a tabu search algorithm, within a reasonable computing time. These results have been improved applying a diversification rule to the tabu search procedure.*

**Keywords:** tabu search, scheduling, reconfigurable manufacturing.

## 1. Introduction

In mid-nineties the concept of Reconfigurable Manufacturing Systems (RMSs) emerged, since the need for systems that are capable of being quickly adapted to changing market demands, by providing exactly the needed functionality and capacity at any time, has been realised. A RMS is a manufacturing system designed for rapid changes in structure, hardware and software components in order to adjust production capacity and functionality within a part family in response to sudden changes in market (Koren et al., 1999).

The working of a RMS starts with the classification of products into families, each of which is a set of similar products. Several sets of families can be formed from the products that a company launches to the market. The options for grouping those products are diverse. One of those options is to group products into a dendogram, which represents the families' formation based on product's similarities. The most similar products are grouped together. The dendogram indicates, through a percentage, the similarity among the products in the family. Figure 1 shows an example of dendogram for three products which presents three sets of families, indicated by three levels. The first one (L=1) is composed of three families, each of which is composed of one product. The second one (L=2) is composed of two families, and the last one is composed of one family.

[*] This research has been partly funded by the Spanish Ministry of Education and Science, project DPI2005-09210. The authors wish to acknowledge the Spanish Government for their support.

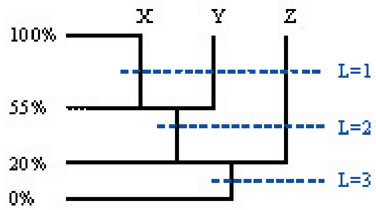


**Figure 1.** Example of dendogram for three products

The manufacturer selects one family to produce, and the RMS is configured to produce the selected family. Once this family is finished, the RMS is configured to effectively produce the following, and so forth (Xiaobo et al., 2000). Thus, the RMS configuration changes for producing different families. The rationale is to minimise the costs associated to configure the manufacturing system from producing one family to the following and the costs associated with the under utilisation of the resources of each machine while producing those families (Racero et al., 2005).

This research aims at the development of a heuristic approach between a variant of the nearest neighbour heuristic and tabu search to selects the best set of product families and their production scheduling when exact methodologies cannot offer a solution, due to the complexity of the instances may requires an excessive computing time.

## 2. Optimal Selection and Scheduling of Product Families in RMS

The RMS is configured to produce one family with both functionality and quantity required. As stated before, once a family is manufactured, the RMS is configured to produce the following one. When the last family has been completed, the system is reconfigured in order to repeat the process again. In each change of the configuration, the manufacturing company incurs in a changeover cost, which depends on the current configuration and the destination configuration (Xiaobo et al., 2000). The manufacturing system has to adapt its capacity and functionality to the production of each family. Therefore, in each configuration of the system, the capacity of the machines and the utilisation of their functionalities are parameters to optimise. A RMS tends to the usage of the full capacity and functionalities of the installed machines.

From the dendogram, the selection of families can be made. There are three different levels in Figure 1, each with different families. Upper levels are composed of several families with few products and high similarity among them. On the contrary, families in bottom levels are composed of few families with lots of products with low similarity.

The selection of families can be solved by calculating the cost of each level in the dendogram, and the level with the lowest cost will be selected. Thus, all the possible solutions are evaluated. A high number of products involve lots of calculations, which may require years for solving. Therefore, for the selection of the families, a model that includes the key parameters expressed above and facilitates this selection is required.

This problem is quite similar to the Travelling Salesman Problem (TSP) which seeks to identify a Hamiltonian path (a tour) that minimise the distance travelled by the salesman. The goal of minimising total distance can be changed into minimising total cost or time. At first glance, several similarities with our problem arise. First, cities in TSP can be compared to product families in RMS. Second, the goal in TSP is to minimise the total distance/cost/time required for completing the tour, whereas in RMS the goal is to minimise the total cost. Finally, the salesman in TSP has to finish in the initial city, whereas a RMS is configured for producing the first family when the last one has finished. It is well known that the TSP is a combinatorial optimisation problem of NP class. Thus, the selection and scheduling of product families in RMS is a NP class problem too.

The mathematical model proposed (Racero et al., 2005) solves a TSP in each level of the dendogram, identifying the tour that presents minimum cost in each level. Then, the model compares the best results in each level and selects the best of them. Therefore, the problem to solve is a multi-level TSP. The model is based on the following assumptions: machines are able to support any demand of products, a unique process plan exists for each individual component to manufacture, and each operation or set of operations are carried out in one specific machine. Its objective is to select the level of the dendogram that minimises the cost of the system reconfiguration and the costs due to an under-utilisation of the resources of the machines while product families are being manufactured. The model takes into account the following assumptions:

– One level of the dendogram is only selected.

– All families of the selected level are manufactured, and none from the other levels.

– In each level, families are manufactured one by one carrying out a production order. At the end, the system is reconfigured to manufacture the initial family. It does not matter which the initial family is because cost associated to the production order does not exist.

– Sub-routes that form a non-Hamiltonian path are not allowed.

Due to the lack of existing reconfigurable systems, some instances from the literature regarding Cellular Manufacturing Systems (CMSs) have been modified in order to be taken as RMS instances. Data in CMS instances are machines and parts. Machines have been converted into products, and parts into product components. Besides, the same number of reconfigurable machines than products is selected, and the number of machine modules is twice than machines. Reference (Racero et al., 2005) tested the model in a batch of 35 instances of RMS which was solved optimally applying the mathematical model previously outlined.

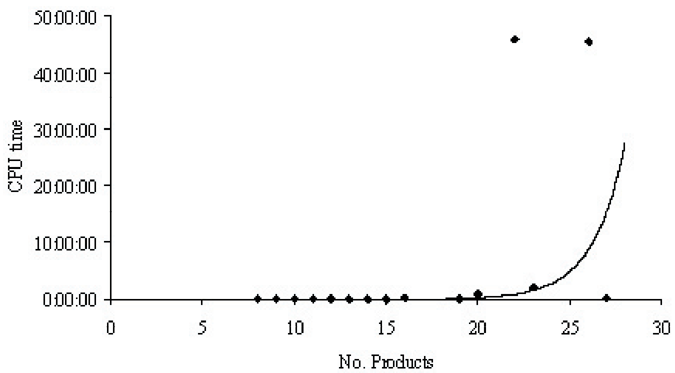


**Figure 2.** Tendency of CPU time with the number of products (Racero et al., 2005)

The CPU time (in hh:mm:ss format) required for solving the model was increasing with the number of products. Those instances were solved with linear programming software, using a CPU at 3.06 GHz PC. Figure 2 presents the tendency of CPU time with the number of products for all the instances previously referred. Results of the objective function are deployed in Table 2 (Optimal method).

As can be observed, the tendency of CPU time with the number of products is increased hardly when the number of products is higher than 25. Therefore, the model is appropriated for problems with 25 products or less. This result shows that an approach based on heuristics must be developed for solving problems with more than 25 different products.

## 3. Metaheuristic Procedures for Tackling the Problem

Due to the existing difficulty for solving combinatorial problems in an optimal way, the development and implementation of heuristic procedures able to provide acceptable solutions within a reasonable computing time become essential (Adenso-Diaz et al., 1996). Heuristic algorithms frequently use any kind of specific knowledge of the problem to solve in order to build a solution or to improve an existing one. As commented before, the problem we face is similar to the TSP. As reconfiguration from a family (i) to family (j) is different than the opposite way, the problem is an asymmetric TSP (ATSP), and the heuristics that solve it may be divided in the following categories:

- Specific heuristics for the ATSP
- General heuristics or metaheuristics applied to the ATSP

Some of them offer solutions near to the optimal one. Therefore, if a small deviation from the optimal solution is acceptable, these techniques can be used as solution methods (Helsgaun, 2000).

### 3.1. Specific Heuristics for the ATSP

These heuristics are simple algorithms that usually require relatively short computational times. Generally speaking, they may be divided into the following four categories (Helsgaun, 2000): (a) Hamiltonian cycle construction heuristics, (b) Hamiltonian cycle improving heuristics, (c) Heuristics based on patching cycles together, and (d) Hybrid heuristics. Considering a graph composed of nodes (cities) and arcs (distances among cities), a cycle is a path in which the same arc is not travelled twice, and it finishes in the initial city. A Hamiltonian cycle, besides the above requirements, has to cover all the nodes only once.

Tour construction heuristics add a city in each iteration of the algorithm until a Hamiltonian cycle has been covered. In RMS, we have a multi-level TSP problem and therefore each step of the algorithm adds one product family to schedule, in each level of the dendogram. They are very fast algorithms, frequently used for generating an approximate solution when the time is limited, obtaining a starting point for the application of other algorithms, or even obtaining a upper bound for exact algorithms. A famous heuristic is the Nearest Neighbour (Rosenkrantz et al., 1977) which starts choosing a random city and then successively goes to the nearest unvisited city. Some other heuristics are Greedy, random insertion, or Clark-Wright heuristic.

### 3.2. General Heuristics Applied to the ATSP

In the last years, several researchers have concentrated in the development of a specific class of algorithms called metaheuristics. They are based on general frameworks which can be applied to diverse optimisation problems with little modifications (Gambardella and Dorigo, 2000).

Some examples are Simulated Annealing, Genetic Algorithms, Tabu Search, GRASP, Neural Networks and Ant Colonies Optimisation.

To solve the problem of selecting and sequencing product families in RMS a tabu search will be applied because it has demonstrated to be a useful optimisation technique to solve different combinatorial problems.

## 4. Heuristics to Solve the RMS Problem

In order to solve the RMS problem, a specific heuristic based on the nearest neighbour method has been developed and implemented. Results have been compared to those obtained with the exact method outlined in section 2. Besides, a general metaheuristic based on tabu search has been implemented using the solutions obtained with the specific heuristic as initial solutions of the tabu search algorithm.

### 4.1. Variant of the Nearest Neighbour Heuristic

This procedure starts by evaluating the reconfiguration costs between each pair of product families. Then, the minimum cost is chosen to be used as the starting point. Once scheduled the first pair of families, the following family to add to the sequence is the not-yet-scheduled-family with minimum reconfiguration cost. The procedure finishes when all product families are scheduled in one single sequence. For example, considering four product families to schedule {A, B, C, D} with the reconfiguration costs between them presented in Table 1.

**Table 1.** Reconfiguration costs between families

| | A | B | C | D |
|---|---|---|---|---|
| A | - | 5 | 2 | 7 |
| B | 2 | - | 3 | 1 |
| C | 9 | 5 | - | 8 |
| D | 6 | 4 | 6 | - |

The sequence that presents the minimum reconfiguration cost is {B-D} and it is chosen as the first pair of families to schedule. From D, the following possible family with the minimum reconfiguration cost is chosen. There are two possible families to choose {A, C} and both have the same cost, so one of them is chosen randomly for example family A, and the current sequence is {B-D-A}. As there is only one family to sequence, it is allocated at the end. Therefore, the final sequence obtained with this heuristic is {B-D-A-C}.

The heuristic does not take into account the under utilisation cost because it is independent of the generated sequence.

Table 2 (SH columns) shows the results obtained with the implementation of the developed procedure in the same batch of instances used in (Racero et al., 2005) which optimal solutions are known. Besides, deviation percentages of the solutions gained with the heuristic regarding optimal solutions are shown too. In all the cases, the computing time required to solve the instances are less than one second.

Results showed in Table 2 (SH columns) can be considered as satisfactory because the heuristic offers good solutions in less than one second to NP problems. In 19 of the 35 instances the deviation from the optimum is less than 5%, in 28 instances deviation is less than 10% and all of them present a deviation lower to 18%. As an average value, the deviation of the instances from optimum is 5.85%. It must be noticed that to obtain results with the application of the optimal method, the computing time required becomes not feasible with more than 25 products to group. This heuristic can offer a solution very quickly to the RMS problem with any number of products to group.

## 4.2. Tabu search

Tabu search selects the best possible movement in each step, allowing a solution worse than the actual one that permits to escape from a local optimum and to continue the search for better solutions. In order to avoid the return at a former local optimum and to create a cycle, some movements are classified as "tabu" in the next iterations.

The neighbourhood of the solution has been generated with inserting movements, which consists of including a scheduled family in a different position of the sequence. For example, in the sequence {A-B-C-D} family A can be inserted in the third position, being the new sequence {B-C-A-D}. In order to generate the neighbourhood, each family is inserted in the other positions of the sequence and therefore, the size of the neighbourhood for a set of $n$ product families is $n(n-1)$. As intensification rule, the algorithm searches in the neighbourhood and selects as the new solution the sequence of families with the fewest cost, though it may be worse than the actual sequence. Two possible ways to finish the algorithm have been implemented: a certain number of operations and a certain number of iterations without improving the solution. The maximum number of operations has been fixed according with the number of families: $n^3+n^2$, and the number of iterations without improving is a percentage of $n^3+n^2$. But even this number may be quite small for escaping from a local optimum (even $n^n$ may be not enough). The escape from huge valleys has been developed with the use of a diversification rule. This rule continues the search in different areas by generating random sequences and choosing the best one.

It is well known that tabu list size affects the performance of the heuristic (Glover, 1989). The literature presents some researches referring to the determination of the optimal size of tabu list for the symmetric TSP with $n$ cities. Glover (1989) investigated four different tabu list sizes, ranging from $n/4$ to $n$. The tabu list size to use in this research will be the number of products to group.

Tabu list does not store the whole sequence, but some attributes of the manufacturing sequence. They are the family that changes its position and the position to insert the family.

Note that a RMS instance is composed of several sets of families, in different levels of the dendogram. Therefore, an ATSP in each level of the dendogram must be solved, applying the above approach. As all possible sequences in the same level have the same under-utilisation costs, they are not used in the described procedure but it is added at the end. Once costs have been obtained for each level, the procedure finishes selecting the best of them.

The implemented procedure tries to improve the solutions offered by the specific heuristic. Once this has been obtained, the intensification process starts by generating the neighbourhood of the solution. The new sequences are evaluated and the best one is selected in each step. The process finishes when $n^3+n^2$ iterations have been completed. The algorithm has been storing the best solution found in each step, and gives it back when finishing. Results are shown in Table 2.

Results in Table 2 show that the implementation of tabu search improves, or make equal in the worst case, the solution obtained with the specific heuristic for each instances. The instance average deviation from the optimum gained with the tabu search approach is 1.83%, quite better than the 5.85% obtained with the specific heuristic. An optimal stopping condition is fundamental to obtain a satisfactory solution with low computing time.

**Table 2.** Solution of instances applying specific heuristic and tabu search

| Reference | Optimal method † | Specific heuristic (SH) | Tabu search (TS) | % deviation SH from optimum | % deviation TS from optimum | CPU time (TS) |
|---|---|---|---|---|---|---|
| SHA95A | 85 | 87 | 85† | 2.35 | 0.00 | 0:00:01 |
| SHA95B | 88 | 88† | 88† | 0.00 | 0.00 | 0:00:02 |
| COA88 | 89 | 89† | 89† | 0.00 | 0.00 | 0:00:04 |
| AKT96 | 126 | 129 | 128 | 2.38 | 1.59 | 0:00:08 |
| CHE96C | 128 | 138 | 128† | 7.81 | 0.00 | 0:00:05 |
| SHA95C | 158 | 167 | 159 | 5.70 | 0.63 | 0:00:04 |
| SEI89 | 143 | 144 | 143† | 0.70 | 0.00 | 0:00:07 |
| MCC72A | 79 | 79† | 79† | 0.00 | 0.00 | 0:00:12 |
| VAK90 | 166 | 166† | 166† | 0.00 | 0.00 | 0:00:12 |
| CRA96 | 150 | 152 | 150† | 1.33 | 0.00 | 0:00:24 |
| ASK87 | 159 | 163 | 159† | 2.52 | 0.00 | 0:00:34 |
| CHA82 | 98 | 102 | 98† | 4.08 | 0.00 | 0:00:44 |
| SHA95D | 78 | 82 | 78† | 5.13 | 0.00 | 0:00:45 |
| CHE95 | 136 | 148 | 136† | 8.82 | 0.00 | 0:00:46 |
| BOC91A | 340 | 367 | 350 | 7.94 | 2.94 | 0:01:18 |
| BOC91B | 242 | 250 | 244 | 3.31 | 0.83 | 0:01:08 |
| BOC91C | 211 | 224 | 212 | 6.16 | 0.47 | 0:01:07 |
| BOC91D | 302 | 315 | 302† | 4.30 | 0.00 | 0:01:19 |
| BOC91E | 240 | 240† | 240† | 0.00 | 0.00 | 0:01:15 |
| BOC91F | 252 | 260 | 252† | 3.17 | 0.00 | 0:01:11 |
| BOC91G | 286 | 297 | 286† | 3.85 | 0.00 | 0:01:09 |
| BOC91H | 294 | 301 | 301 | 2.38 | 2.38 | 0:01:08 |
| BOC91I | 241 | 275 | 253 | 14.11 | 4.98 | 0:01:08 |
| BOC91J | 222 | 255 | 232 | 14.86 | 4.50 | 0:01:12 |
| SRI90 | 243 | 255 | 243† | 4.94 | 0.00 | 0:01:07 |
| KIN80 | 353 | 368 | 362 | 4.25 | 2.55 | 0:01:07 |
| ASK91 | 118 | 122 | 120 | 3.39 | 1.69 | 0:03:19 |
| VEN90A | 305 | 322 | 305† | 5.57 | 0.00 | 0:04:32 |
| CHE96A | 341 | 392 | 364 | 14.96 | 6.74 | 0:04:17 |
| CHE96B | 278 | 325 | 299 | 16.91 | 7.55 | 0:04:10 |
| NG96 | 248 | 292 | 290 | 17.74 | 16.94 | 0:04:08 |
| VEN90B | 126 | 134 | 128 | 6.35 | 1.59 | 0:07:52 |
| KUM86 | 364 | 412 | 369 | 13.19 | 1.37 | 0:09:44 |
| ADI97 | 497 | 529 | 516 | 6.44 | 3.82 | 0:20:49 |
| MCC72B | 541 | 596 | 559 | 10.17 | 3.33 | 0:25:28 |

Studying the quality of these solutions, it has been realised that the best solutions are obtained in the early iterations. Thus, thirteen instances (composed of a total sum of 202 levels) have been selected randomly in order to study the iteration in which the best solution is found. Results have shown that the best solution has been found doing less than 12% of the upper bound iterations number ($n^3+n^2$). Therefore, the batch of instances can be solved with the stopping criteria of a maximum number of iterations without improving the best solution found. For assuring more iterations than the 12% obtained experimentally, this condition has been set to 30%. Results have demonstrated that the same results are obtained, and consequently saving computing time. Figure 3 shows how difference in computing time with and without the stopping condition, and how it increases together with the number of products to group.

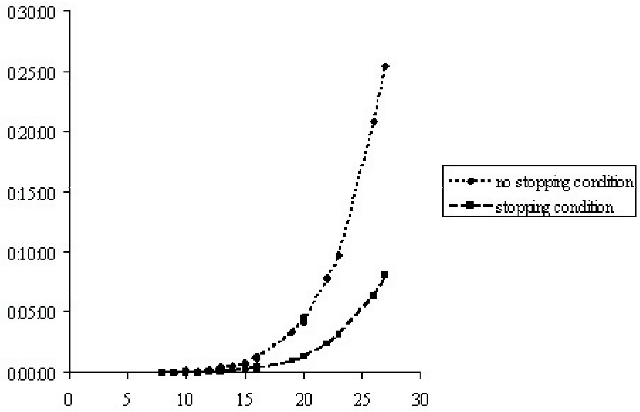


**Figure 3.** Difference in computing time due to the stopping condition

A diversification rule has been developed with the aim of searching in other areas of the solution space when it is difficult to overcome huge local optima. This rule is implemented by generating 1000 random solutions when the best solution has not improved after a certain number of iterations. The heuristic evaluates all of them and selects the best one as the current solution. The stopping condition and the diversification rule can be implemented together only if the number of iterations without improving the best solution found is smaller than the stopping condition. If the new area does not improve the best solution, new searches in other areas can be carried out.

With the aim of obtaining the best number of iterations without improving to implement the rule, some experiments have been carried out. Experiments have been applied with different numbers of iterations without improving: 10, 5, 1, and 0.5 % of the maximum number of iterations. As a comparative measure, the sum of deviation from the optimum of the previous batch of instances has been used. As computing time are similar, results show up that the best is to use 1%. Thus, the optimum number of iterations without improving the current solution before applying the diversification rule is $0.01(n^3+n^2)$.

The implementation of this rule improves previous solutions in most of the cases, as it is shown in Table 3. Due to the random condition of the diversification rule, instances have been solved five times to obtain the results.

Table 3 has shown the successive improvements gained when applying the heuristic approach to solve the RMS problem, from the implementation of the specific heuristic to the implementation of the heuristic approach based on tabu search with diversification rule. In the last case, the optimal solution is obtained in 31 of the 35 instances, with an instance average deviation from the optimum of 0.26% within an admissible computing time.

## 5. Conclusion

This paper has presented a heuristic approach to select and sequence product families in RMS. This procedure is based on a variant of the nearest neighbour heuristic and a tabu search algorithm with both intensification and diversification rules together with a stopping condition. The intensification rule has been developed in a neighbourhood created with inserting movements.

**Table 3.** Solution of instances applying the diversification rule

| Reference | Optimal method † | Specific heuristic (SH) | Tabu search (TS) | TS with div. rule (TSDR) | % deviation (TSDR) from optimum | CPU time (TSDR) |
|---|---|---|---|---|---|---|
| SHA95A | 85 | 87 | 85† | 85† | 0.00 | 0:00:01 |
| SHA95B | 88 | 88† | 88† | 88† | 0.00 | 0:00:02 |
| COA88 | 89 | 89† | 89† | 89† | 0.00 | 0:00:02 |
| AKT96 | 126 | 129 | 128 | 126† | 0.00 | 0:00:06 |
| CHE96C | 128 | 138 | 128† | 128† | 0.00 | 0:00:02 |
| SHA95C | 158 | 167 | 159 | 159 | 0.63 | 0:00:02 |
| SEI89 | 143 | 144 | 143† | 143† | 0.00 | 0:00:04 |
| MCC72A | 79 | 79† | 79† | 79† | 0.00 | 0:00:05 |
| VAK90 | 166 | 166† | 166† | 166† | 0.00 | 0:00:06 |
| CRA96 | 150 | 152 | 150† | 150† | 0.00 | 0:00:08 |
| ASK87 | 159 | 163 | 159† | 159† | 0.00 | 0:00:13 |
| CHA82 | 98 | 102 | 98† | 98† | 0.00 | 0:00:16 |
| SHA95D | 78 | 82 | 78† | 78† | 0.00 | 0:00:16 |
| CHE95 | 136 | 148 | 136† | 136† | 0.00 | 0:00:22 |
| BOC91A | 340 | 367 | 350 | 340† | 0.00 | 0:00:28 |
| BOC91B | 242 | 250 | 244 | 242† | 0.00 | 0:00:28 |
| BOC91C | 211 | 224 | 212 | 211† | 0.00 | 0:00:28 |
| BOC91D | 302 | 315 | 302† | 302† | 0.00 | 0:00:24 |
| BOC91E | 240 | 240† | 240† | 240† | 0.00 | 0:00:29 |
| BOC91F | 252 | 260 | 252† | 252† | 0.00 | 0:00:26 |
| BOC91G | 286 | 297 | 286† | 286† | 0.00 | 0:00:24 |
| BOC91H | 294 | 301 | 301 | 294† | 0.00 | 0:00:35 |
| BOC91I | 241 | 275 | 253 | 241† | 0.00 | 0:00:38 |
| BOC91J | 222 | 255 | 232 | 231 | 4.05 | 0:00:32 |
| SRI90 | 243 | 255 | 243† | 243† | 0.00 | 0:00:26 |
| KIN80 | 353 | 368 | 362 | 353† | 0.00 | 0:00:29 |
| ASK91 | 118 | 122 | 120 | 120 | 1.69 | 0:01:08 |
| VEN90A | 305 | 322 | 305† | 305† | 0.00 | 0:01:27 |
| CHE96A | 341 | 392 | 364 | 341† | 0.00 | 0:02:08 |
| CHE96B | 278 | 325 | 299 | 278† | 0.00 | 0:01:42 |
| NG96 | 248 | 292 | 290 | 248† | 0.00 | 0:02:20 |
| VEN90B | 126 | 134 | 128 | 126† | 0.00 | 0:02:23 |
| KUM86 | 364 | 412 | 369 | 364† | 0.00 | 0:05:44 |
| ADI97 | 497 | 529 | 516 | 497† | 0.00 | 0:10:18 |
| MCC72B | 541 | 596 | 559 | 555 | 2.95 | 0:10:09 |

The implemented specific heuristic based on the nearest neighbour offers good solutions very quickly (in less than one second). Solutions to the problem are significantly improved when the specific heuristic is used with tabu search, within a reasonable computing time. These results have been improved applying a diversification rule to the tabu search procedure.

Therefore, it can be stated that the implementation of the heuristic approach to solve the RMS problem drives to obtain satisfactory results within a reasonable computing time, overcoming the limitations imposed by the optimal method which was limited to offer results when there was more than 25 products to group. This research has pointed out that meta-heuristic methods are useful to tackle difficult real problems.

## References

Adenso-Diaz, B.; Glover, F.; Ghaziri, H.; Gonzalez, J.L.; Laguna, M.; Moscato, P.; Tseng, F. (1996). Heuristic optimization and Neural Networks in Operations Management and Engineering. Paraninfo.

Gambardella, L.M.; Dorigo, M. (2000). An Ant Colony system hybridized with a new local search for the sequential ordering problem. INFORMS Journal on Computing, Vol. 12, No. 3, pp. 237-255.

Glover, F. (1989). Tabu Search – Part I. ORSA Journal on Computing, Vol.1, No.3, pp.190-206.

Helsgaun, K. (2000). An Effective Implementation of the Lin-Kernighan Travelling Salesman Problem. European Journal of Operational Research, 126, pp.106-130.

Koren, Y.; Heisel, U.; Jovane F.; Moriwaki, T.; Pritschow, G.; Ulsoy, G.; Van Brussel, H. (1999). Reconfigurable Manufacturing Systems. Annals of the CIRP, Vol. 48, pp. 1-14.

Racero, J.; Eguia, I.; Galan, R.; Canca, D. (2005). A Mathematical Model for Product Families Selection in Reconfigurable Manufacturing Systems. Proceedings of 15th International Conference on Flexible Automation and Intelligent Manufacturing, pp.810-816, Bilbao.

Rosenkrantz, D.J.; Stearns, R.E.; Lewis P.M. (1977). An Analysis of several heuristics for the travelling salesman problem. SIAM Journal on Computing, 6, pp.563-581.

Xiaobo, Z.; Jiancai, W.; Zhenbi, L. (2000). A Stochastic Model of a Reconfigurable Manufacturing System, Part 1: A Framework. International Journal of Production Research, Vol.38, No.10, pp.2273-2285.