Proceedings of the 1999 IEEE
International Symposium on Assembly and Task Planning
Porto, Portugal - July 1999

CONSTRAINT PROGRAMMING APPROACH TO SOLVE FACILITY LAYOUT DESIGN PROBLEMS

José António Tavares[1]
jtavares@dei.isep.ipp.pt

Carlos Ramos[1]
csr@dei.isep.ipp.pt

José Neves[2]
jneves@di.uminho.pt

[1]*Instituto Superior de Engenharia do Porto*
Rua de S° Tomé - 4200 Porto – Portugal
Phone +351 2 8340500, Fax +351 2 8321159

[2]*Dept. de Informática - Universidade do Minho*
Campus de Gualtar - 4709 Braga Codex – Portugal
Phone +351 53 604466, Fax: +351 53 604471

Abstract

In this paper, we propose a methodology to solve the facility layout design problem by using constraint logic programming (CLP), which has proved to be a technology that gives good results when applied to a combinatorial problem optimisation. Methods to solve facility layout problems have to deal with a large set of factors, namely sales and production estimations, manufacturing process compatibilities, delivery dates, quality, spatial requirements, economics, management, human resources and environment. These factors make the facility layout design an extremely complex problem to solve. This complexity motivated the development of a methodology to solve the facility layout design problem, which uses CLP.

Facilities Layout Design

One of the most complex problems in the industry is the Facility Layout Design Problem. In a more general definition, the facilities layout design is the planning of the location of machines, employees, workstations, clients service areas, warehouses and the material and people flow pattern around, the movement inside, at the input and at the output of the productive installations.

In a factory, the layout is a fundamental issue. From it, the good equipment and human resources use has a great influence on the real output, whatever is the theoretical installed capacity of facility. In a manufacturing facility it is necessary to plan what is the operation sequences, the available equipment for each operation type and the flow of the materials and people between them. The warehouses location, how they are supplied from outside, the areas and how the distribution transportation are loaded are also task of the planning process. Aspects derived from layout, like work conditions (noise levels, temperature and air quality), have to be considered. The definition and the dynamic management of the manufacturing layout is a fundamental task of the operations manager, since the manufacturing process efficiency depends from it, and also from the company capacity to use the available material and human resources.

The major objective, when designing the manufacturing layout is to design a physical arrangement that most economically achieves the required output quantity and quality [1]. Achieving the required output involves the improvement of process and work flow; the proper allocation of space and resources; the easier access to supplies and materials; the plant efficiency increase; maximise the use of space; the safety improvement and obtaining cost savings. The required output has quantitative and qualitative factors associated with it, which can be difficult to model and analyse, and so, decisions related with the design of the facilities layout must take into account some of the following aspects:

- Which production units[1] must exist in the plant?
- How much space is available and which are the production unit capacity needs?

[1] Refers to machines, cells or departments.

- What is the geometric configuration of each production unit?
- Where will be placed each production unit?

To answer these questions, the layout designer needs, at least, to have access to data like the facility layout objective, the product or services demand (frequently estimated values), processing requirements, space available and the management requirements. This data can be found in the manufacturing department, but also at marketing department, product design department and from the management policy. In order to select the best or at least a good facility layout, it is necessary to consider various performance aspects, such as capital investment, material handling, flexibility and work environment.

Layout Problem Characterisation

This section describes the model used to solve the facilities layout design problem using constraint logic programming (CLP) [2, 3, 4, 5]. In general the model used follows the multi-row model [1, 6, 7], where the production units can be located anywhere in the facility plant, thus, the production units arrangement does not have to be along straight rows. The model deals only with single-floor facilities.

To solve layout design problem using CLP it was chosen the finite domain scheme for the problem variables. The logic programming language *Prolog* is used to describe the data required to generate the facility layout.

Facility Plant

To locate all the production units it is necessary to know the available space and the facility shape. In the presented model, the facility plant is modelled by doing an approximation with rectangle surrounding its shape. All the production resources are located inside of that rectangle. As it will be shown later, locations inside the rectangle that do not belong to the facilities plant can be represented by forbidden areas, therefore the production units will be located in the real facility plant. The surrounding rectangle is represented using the following extension of the predicate *plantSize*, where the first argument w is width and the second l is length:

plantSize(w, l).

Production Units

The production units, located in the facility plant, are classified according to the performed tasks, named as production unit types. This feature allows the identification of the production units by the type. As in the facility plant, a surrounding rectangular shape is used to approximate the actual production unit shape. Figure 1 shows the rectangular model for the production units, where it is represented the production unit width and length and a third value, referred as clearance, which represents the security empty area required around a production unit in relation to the others production unit neighbours.

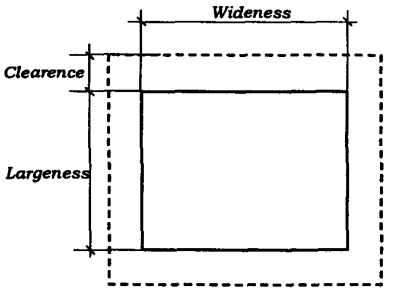

Figure 1: The production units geometric shape approximation.

To describe the each production unit type the following predicate extension is used:

productionUnit (m_i, q_i, a_i, w_i, W_i, l_i, L_i, c_i).

where

m_i	production unit type name;
q_i	number of production units of type m_i;
a_i	production unit area;
w_i, W_i	the smaller and larger width value;
l_i, L_i	the smaller and larger length value;
c_i	clearance.

This representation of the production unit imposes fixed area for the production units, but width and length values may change inside predefined limits. It is obvious that, since the area is fixed, if the width increases then the length has to decrease. This approach is used to deal with two situations. One situation occurs when a production unit is a department or a cell containing several machines requiring a given area, but with flexible shape. The other occurs when the production unit has fixed area and shape, and it must be decided what will be the orientation.

Products Demand

The product demand is not constant in time, and most of time is for off-line planning purposes. In such cases an estimated demand volume for a period of time should be used. In the layout problem representation the term part is used to group a final product or components used later to manufacture a final product, so the part demand should be used instead of final product demand. This approach means that, all the products manufactured in the facility plant have to be decomposed in their parts, each one with their demand volume. The demand for each part is represent by the following predicate extension:

partDemand(p_i, d_i, t_i).

where

p_i	part name;
d_i	demand;
t_i	time period per unit time for demand.

Production Tasks

All parts to be manufactured have to complete several processing operations in a predefined sequence. Each processing operation, which is referred as *task*, is done in one production unit able to perform it. One task may be

done in any production unit of the same type, since several production units of the same type may be available. Also, different tasks can share the same production unit, but not simultaneously. The selected instance of the production unit type to perform the task occurs later in the scheduling process, normally done during the facility plant life cycle. Usually, the part being manufactured has to visit one production unit at time if the decomposition of the products result in simple non-decomposable parts. In the presented model it is not necessary to decompose completely the parts. In this way the tasks sequences of all the parts are described by diagrams like the one shown in the Figure 2.

The task diagrams description is done by firstly identifying all the tasks themselves and then the actual task sequence. To a task itself it must be identified the part being processed and production unit type required. Additionally, the produced part quantity and time required to complete the task have to be known The task description uses the extension of the predicate

task(Task, Part, ProductionUnit, Batch, Time).

where

Task	task name;
Part	part name;
ProductionUnit	the production unit type name;
Batch	part quantity processed by the task;
Time	time needed to complete the task.

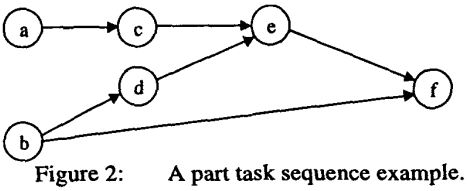

Figure 2: A part task sequence example.

After describing all the tasks it is necessary to describe the task sequences. To describe those, it is necessary to know each task immediate predecessors. For example, the *e* task shown in Figure 2 has the *c* and *d* tasks as predecessor tasks. The material flow between pairs of production units is implicitly defined by describing the task sequences. In most situations, it must be available the output quantity of all predecessor tasks required to complete a task. This situation occurs, for example, when a task is an assembly operation that requires parts produced at preceding tasks. An associated cost value, representing the effort of getting the material of the preceding tasks can be included, like for example, the cost of the material handling or transportation equipment. To identify the predecessors of a task is used the extension of the predicate

predecessor(Task,[(Task$_1$,Out$_1$, Cost$_1$),..., (Task$_i$,Out$_i$, Cost$_i$) , ..., (Task$_n$,Out$_n$, Cost$_n$)]).

CLP Solving of the Layout Problem

This section describes the approach used to solve the layout problem using CLP(*FD*) scheme. The problem variables and constraints are identified and the optimisation procedure is described.

Variables

The general principle of generating a layout consists in a simplified form, in the location of different production units, represented by rectangles, inside of the facility plant floor, also represented by a bigger rectangle. Therefore, a solution to the problem is the co-ordinate assignment for all production units. So, in order to solve the problem it must be assigned to each production unit two variables, X_i and Y_i, representing the production unit co-ordinates. At the end, those variables will have assigned values representing the cartesian co-ordinates of the selected location. In addition, considering the presented model, it is also necessary to decide what is the width of the production units (once the width is known, the length is also known). The finite domain range for those four variables required for each production unit are initialised as follows:

$$W_i \in wl_i .. wh_i, \quad L_i \in ll_i .. lh_i$$
$$X_i \in 0 .. W\text{–}mindomain(W_i)\text{-}1, \quad Y_i \in 0 .. L\text{–}mindomain(L_i)\text{-}1$$

where

W_i and L_i	is the domain width and length of production unit *i*;
wl_i and ll_i	width and length minimum domain value of production unit *i*;
wh_i and lh_i	width and length maximum domain value of production unit *i*;
X_i and Y_i	represents the domain location of production unit *i*;
W and L	the facility plant width and length.

The presented domain range of the co-ordinates variables X_i and Y_i is not sufficient to prevent part of production unit area to be located outside of the facility plant area. This is because W_i and L_i (width and length) are domain variables, thus their values are not known initially. Therefore, to prevent this, the following constraints are imposed:

$$X_i + W_i \leq W, \qquad Y_i + L_i \leq L$$

As referred before, it assumed that all the production units must have a known fixed area. Since the width and length can have a range of possible values then the following constraint must be added in order to maintain a constant area.

$$W_i \times L_i = A_i$$

Note that, since the domain of those variables has discrete values (an interval of integer values), only a few possible shapes are valid.

In those cases, where the width and length are known and it must be decided the production unit orientation in the facility plant, the user set the area equal or less than 0. The system then sets the domain of width and length as follows:

$$W_i \in [wl_i, ll_i], \qquad L_i \in [wl_i, ll_i]$$

If the wl_i and ll_i values are not equal then the system add the following constraint to ensure that the production unit won't have a square shape in generated solutions:

$$W_i \neq L_i$$

Constraints

In the facility layout problem, the production units have to be located obeying to some constraints. Some of them are stated by the system in order to get a correct layout, others have to be stated by the user in to satisfy specific problem requirements like technological, environmental, strategic and others. Additionally, constraints can be stated in order to get better solutions in less time. The system supports some geometric constraints that are supposed to cover a wide range of layout problem cases.

No Overlap One constraint that should always be present is the "no overlap" constraint, which imposes that all production units, will be located in the facility plant without invading the area occupied by the others. The system always imposes this constraint to all formed production units pairs.

Neighbourhood In some situations it is desirable to locate two production units close to each other, like for example, when there is a large volume of material flow between them. Expressing this desire earlier by stating a constraint, it is possible to reduce the search space significantly. This constraint is available to the user (layout designer), but have to be used with some care, having the risk of getting a problem without solution, like for example, a production unit having to be neighbour to a large number of other production units.

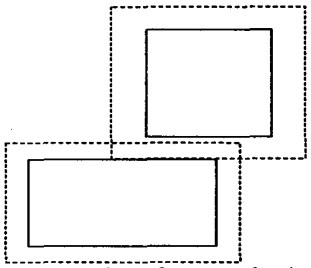

Figure 3: Location of two production units taking into account their clearance.

Adjacency The adjacency constraint is a specialised version of the neighbourhood constraint. Like the neighbourhood, it imposes that two production units should be neighbours. In addition, this constraint places the production units in order that the middle point of them is at the minimum distance. Figure 3 shows a placed solution that is acceptable to the neighbourhood constraint, but it is not to the adjacency constraint. An acceptable solution for both constraints is shown in the Figure 4.

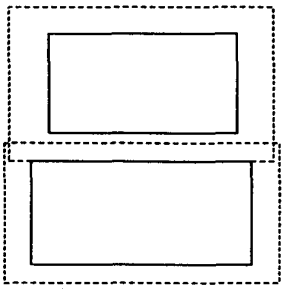

Figure 4: One placement resulted from the adjacency constraint use.

Distance There are situations in which some production units have to be located at some distance from each other. One possible situation occurs when some production units have to operate in a temperature-controlled environment not compatible with others located in the neighbourhood. To prevent situations like this, a distance constraint was made available to the user. This distance constraint applies to pairs of production units. The parameters are the middle co-ordinates of the two production units, the constraint relational operator and an integer value for the distance. The available relational operators are $=$, $<$, $>$, $\neq$, $\leq$ and $\geq$. The model used to measure distances was the rectilinear distance model. This distance constraint can be used, also, to locate one production unit at some distance of given point.

Position To force a production unit to be located inside or outside of some area of the facility plant three constraints were defined, which are collectively named as position constraints. These constraints are the following:

Inside imposes that a production unit should be located in one area from a list of given areas;

Outside a production unit cannot be located inside of any area from a given list.

With these constraints it is possible to reserve space areas for various different purposes like offices or warehouses by specifying a fixed location or by selecting/excluding from a list of possible locations.

Layout Cost Evaluation

There is large number of cost function that could be used to evaluate the generated layout quality. Those cost functions may use qualitative and quantitative factors. The selected cost function used, shown below, focus on quantitative factors and is based in three parameters: the product demand (which defines the flow volume), the distance between each production unit pair having material flow and the cost of moving a unit of material per unit of distance.

$$Cost = \sum_{j=1}^{n} \sum_{i=1}^{n} f_{ij} * d_{ij} * c_{ij}$$

- f_{ij} material flow volume between production unit i and j;
- d_{ij} rectilinear distance between production unit i and j;
- c_{ij} cost of moving a unit of material per unit of distance between production unit i and j.

The c_{ij} parameter is more flexible than the other two and not so obvious. It can be used to represent qualitative factors, each one associated with a weight. Also, it can represent the time spent in the transportation, or the cost associated with the operation of the handling equipment. Others quality and quantitative cost values could be enumerated and they frequently involve a combination of various types of cost value types. In practice, it is hard to determine c_{ij} and so, often it is estimated [6]. This cost function shows that the generated layout has best quality when the cost is minimum, thus the search algorithm has to minimise the value of the cost function.

Search Algorithm

The search algorithm responsible to find the best solution is a depth first branch-and-bound algorithm. The process of searching the solution with the lower cost satisfying the problem constraints can take a long time period to complete, and in most cases, for the average size problems, the time spent is not acceptable. If all the search space was explored in an acceptable time period then the selected solution would be the optimal, otherwise the best solution found in acceptable time period is taken.

The variable selection order has a great impact in a CLP search algorithm performance. The selection order used locates all the production units, one by one. The instantiation order of the variables (co-ordinates, width and length) of each selected production unit to be located is based on first-fail principle, i.e., is selected the variable with small domain. The actual instantiation of each variable starts with the middle value of the domain and then goes alternating to the minimum and the maximum values of the domain.

The actual production unit selection order is done as follows:

1. Select first all the production units having at least one distance constraint. If more than one then sort by area value (bigger first);
2. Select the adjacency constraint pairs having the bigger flow between them. Again, the one with bigger area is located first;
3. If there is no adjacency constraint pair then select the production units participating in the pair which have the bigger flow between them. The bigger area criteria is also used;
4. Of the last two located production units, select one not yet located, which has the bigger flow. If the selected production unit participate in an adjacency constraint pairs then select the other and go to step 3;
5. Go to step 2 until all production units are located.

Some Performance Results

To make some performance tests it was invented some hypothetical facilities in order to use the implemented system to generate a layout. One of these hypothetical facilities has 10 production units, with no special constraints with the exception of two that have to be located far from each other. By analysing the best layout generated it was observed that there is a trend to locate production units close to each other if they have a large material flow between them. This observation suggested that the adjacency constraint should be imposed for some pairs in order to improve the performance and the solution quality. Some experiments were done and for most of the cases it was demonstrated that in fact there was a performance and a solution quality improvement, but the adjacency pairs have to be selected with care since frequently the arbitrary selection leaves to a problem without solutions.

Since the adjacency pairs were chosen arbitrarily, then the question was how to compute the best adjacency pairs. One possible answer to this question is to compute the maximum weight matching. The concept consists in creating a graph with the production units as nodes. There is an arc linking two nodes if there is material flow between the associated production unit and the weight is the flow volume between them. This approach was already followed by [8, 9] to create slicing trees in a cutting stock problem solved, using genetic algorithms.

Same performance and quality results are reported in the Table 1. There are two rows, the first one for the layout generated without the adjacency constraints and the second with some adjacency constraint computed by following the maximum weight matching concept. The results are obtained after a timeout of 60 minutes. Figure 5 shows the best layout generated with the adjacency constraints.

Adjacency Constraint	CPU time	First solution (cost)	Best Solution Found (cost)	Time to find the best solution
No	1590 s	31535.5	30501.5	1474 s
Yes	1418 s	22978.0	22330	87 s

Table 1: Results for hypothetical facility with 10 production units.

The first prototype system was developed using *ECLiPSe* system [10] using the finite domain library and the tests were ran in a DEC AlphaStation.

Conclusions

The work discussed in this paper presents an approach taken to solve the facilities layout problem using CLP. The development of an application following this approach is at an early stage, so a lot of work has to be done. One first implemented prototype was tested with some small test problems. It was showed that, although the performance was not good enough for real cases yet, there is room to improvements in terms of performance and problem size. The work done also showed that, the problem of generating industrial systems configurations could be solved in an effective manner using CLP technology.

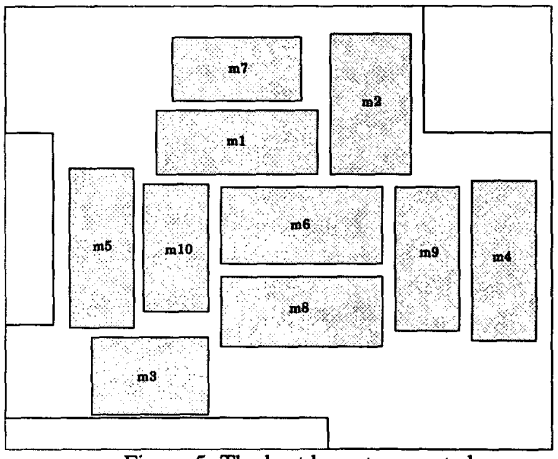

Figure 5: The best layout generated.

One area that must be treated, in order get better performance, is the improvement of the constraints described to achieve a better value propagation. It was observed that the current implementation of the presented constraints to solve the layout problem presents a performance bottleneck in the value propagation.

Another area to be treated is the reduction of the problem search space size, where heuristics can do the job. In fact, applying adjacency constraints as referred in the previous section, which can be viewed as a heuristic, could reduce the search space and provide better solutions, as it was showed. The adjacency constraint, as it was defined, showed that the location of one production unit forces by propagation the location of the adjacent production units. Also the area occupied and the formed shape of the two adjacent productions units is always the same. Those facts suggest that adjacent pairs could be treated as single production units. More, it is possible to apply this method recursively, which is identical to the work developed in [8, 9] when applying the slicing trees methodology.

Finally, the implemented branch-and-bound algorithm makes a systematic search on the search space to find optimal solution. Since the search space is too large, it may take a lot of time, which may be not practicable. Also, stopping after a timeout can give solutions far from the optimal. A different algorithm that would sample the search space is an alternative. The use of genetic algorithms seems to be a good alternative to do the work. The use of genetic algorithms to search optimal solutions in CLP application is a combination that will be explored in our future work.

References

[1] Riggs, J. L., *Production Systems: planning, analysis, and control*, Fourth Edition, John Wiley & Sons, Inc, 1987, ISBN 0-471-85888-9.

[2] Kumar, V., *Algorithms for Constraint Satisfaction Problems: A Survey*, Department of Computer Sciences, University of Minnesota, 1992.

[3] Smith, B. M., *A Tutorial on Constraint Programming*, Division of Artificial Intellingence, University of Leeds, April, 1995.

[4] Jaffar, J., Lassez, Jean-Louis, A., *Constraint logic programming.*, In Proceedings of the 14th ACM Symposium on Principles of Programming Languages, Munich, Germany, pages 111-119, ACM, January, 1987.

[5] Frühwirth, T., Herold, A., Küchenhoff, V., Provost, T., Lim, P., Monfroy E. and Wallace, M., *Constraint Logic Programming - An Informal Introduction*, European Computer-Industry Research Centre, 1993.

[6] Heragu, S., *Facilities Design*, PWS Publishing Company, 1997, ISBN 0-534-95183-X.

[7] Kusiak, A., *Intelligent Manufacturing Systems*, Prentice Hall, Englewood Cliffs, NJ, 1990, ISBN 0-13-468364-1.

[8] Fritsch, A., Vornberger, O., *Cutting Stock by Iterated Matching*, Operations Research Proceedings, Selected Papers of the Int. Conf. on OR 94, U. Derigs, A. Bachem, A. Drexl (eds), Springer Verlag, pp. 92-97, 1995.

[9] V. Schnecke, *Hybrid Genetic Algorithms for Solving Constrained Packing and Placement Problems*, PhD Dissertation, University of Osnabrück, Dept. Mathematics/Computer Science, Dec 1996.

[10] ECRC GmbH, *ECL^{i}PS^e 3.5 - Extensions User Manual*, December, 1995.