Modeling Manufacturing Cell Design Problems: CP vs. MH

Broderick Crawford[1,3], Ricardo Soto[1,2], Gustavo Zuñiga[1], Eric Monfroy[4], and Fernando Paredes[5]

[1] Pontificia Universidad Católica de Valparaíso, Chile
[2] Universidad Autónoma de Chile, Chile
[3] Universidad Finis Terrae, Chile
[4] CNRS, LINA, Université de Nantes, France
[5] Escuela de Ingeniería Industrial Universidad Diego Portales, Chile
{ricardo.soto,broderick.crawford}@ucv.cl,
gustavo.zuniga.m@mail.pucv.cl,
eric.monfroy@univ-nantes.fr,
fernando.paredes@udp.cl

Abstract. The manufacturing cell design problem aims at organizing a manufacturing plant in cells that contain machines processing parts from a same family for a given product. The purpose is to minimize the flow of parts among cells so as to increase productivity while reducing costs. This paper focuses on comparing metaheuristics and constraint programming –from a modeling standpoint– when used to solve this problem.

Keywords: Constraint Programming, Metaheuristics, Manufacturing Cell Design.

1 Introduction

The manufacturing cell design problem (MCDP) is a classic optimization problem from group technology. It aims at organizing a manufacturing plant in cells that contain machines that process similar parts for a given product. The purpose is to minimize the flow of parts among cells so as to increase productivity while reducing costs. During the last years, this problem has been tackled with different techniques which can be organized into two main groups: complete search and approximate methods. Complete search methods explore the whole set of potential solutions in order to reach a solution. These methods may be unsuitable once the problem size increases, since it is not possible to explore the complete combinatorial space in a limited period of time. In this context, approximate methods are more appropriate since they attempt to examine only promising regions of the search space. In the first group we can find approaches devoted to the MCDP based on the classic linear programming [12], goal programming [13,14], constraint programming [20], and Boolean satisfiability [19]. The second group is mainly composed of metaheuristics (MH), such as tabu search [9], simulated annealing [22], particle swarm optimization [6], genetic algorithms [21], and some hybridization of them [8,11].

C. Stephanidis (Ed.): HCII 2014 Posters, Part I, CCIS 434, pp. 498–502, 2014.

This paper focuses on comparing metaheuristics and constraint programming –from a modeling standpoint– when used to solve this problem. This paper is organized as follows: We firstly present the mathematical model representing the MCDP, we provide then the discussion about the contrasts of metaheuristics with respect to constraint programming when modeling this problem. Finally, we give the conclusions and future work.

2 Modeling the MCDP

The optimization model for the MCDP is stated as follows. Let:

- M, be the number of machines,
- P, the number of parts,
- C, the number of cells,
- i, the index of machines ($i = 1, \ldots, M$),
- j, the index of parts ($j = 1, \ldots, P$),
- k, the index of cells ($k = 1, \ldots, C$),
- $A = [a_{ij}]$ the $M \times P$ binary machine-part incidence matrix,
- M_{max}, the maximum number of machines per cell.

  The goal of the objective function is to minimize the number of times that a given part must be processed by a machine that does not belong to the cell that the part has been assigned to. Let:

$$y_{ik} = \begin{cases} 1 \text{ if machine } i \in \text{ cell } k; \\ 0 \text{ otherwise;} \end{cases}$$

$$z_{jk} = \begin{cases} 1 \text{ if part } j \in \text{ family } k; \\ 0 \text{ otherwise;} \end{cases}$$

The mathematical model is as follows:

$$\texttt{minimize} \quad \sum_{k=1}^{C} \sum_{i=1}^{M} \sum_{j=1}^{P} a_{ij} z_{jk} (1 - y_{ik})$$

Subject to

$$\sum_{k=1}^{C} y_{ik} = 1 \quad \forall_i, \qquad \sum_{k=1}^{C} z_{jk} = 1 \quad \forall_j, \qquad \sum_{i=1}^{M} y_{ik} \leq M_{max} \quad \forall_k,$$

3 Discussion

We have modeled and solved this problem via CP, Boolean satisfiability (see details in [19]) and different metaheuristics (Tabu Search [7], Artificial Fish Swarm [10], Cuckoo search [23], and Electromagnetism [24]). Several computer science students have participated in these projects, and after this experience we can highlight two important points.

- The CP paradigm has an important advantage w.r.t metaheuristics. In CP, there is no need to design a specific procedure to solve the given problem, the user rather models the problem and a search engine finds a result. In practice, when problems are solved using metaheuristics a specific algorithm must be designed and implemented according to some pre-established patterns. After the experience gained on the aforementioned projects, students may take about 1 month in studying the metaheuristic and then another month to implement the solution. In the context of CP, students may take 1 month in studying CP but only 2 weeks in implementing the solution.

- On the other side, metaheuristics have also an important advantage w.r.t. CP. After implementation, an experimentation phase is required in order to tune the solvers to reach the optimum. This phase requires a high expertise in both contexts, metaheuristics and CP. However, due to the metaheuristics is built from the scratch by the student, he naturally domains better its implementation so it is easier to modify, tune and experiment with it. This is not the case of CP, where the student uses a toolkit which is normally a black-box that provides some kind of configuration but does not provide the full control. It also exists the possibility of exploring the source code of the toolkit, but this is really a hard task as common solvers includes several code lines, most of them hard to handle and as a consequence hard to successfully modify according to our requirements.

4 Conclusion

In this paper, we discussed the main differences between metaheuristics and constraint programming when used for solving the manufacturing cell design problem. In CP, it suffices to encode the mathematical model in the language of the solver, while when metaheuristics are used it is necessary to implement a specific algorithm to solve the problem. On the other side, an experimentation phase may be easier to face off when metaheuristics are used since the user has the full control of its implementation, which is not the case when CP is used.

A continual research direction is about facilitating the user modeling and experimentation phases, for instance to propose easy-to-use modeling languages [18], modeling techniques [1], and modeling features [17,15,2]. The resolution of this problem via CP in conjunction with autonomous search [16,4,5,3] may be also an interesting direction to follow.

Acknowledgements. Ricardo Soto is supported by Grant CONICYT/ FONDECYT/INICIACION/11130459, Broderick Crawford is supported by Grant CONICYT/FONDECYT/REGULAR/1140897, and Fernando Paredes is supported by Grant CONICYT/FONDECYT/REGULAR/1130455.

References

1. Chenouard, R., Granvilliers, L., Soto, R.: Model-Driven Constraint Programming. In: Proceedings of the 10th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (PPDP), pp. 236–246 (2008)
2. Chenouard, R., Granvilliers, L., Soto, R.: High-Level Modeling of Component-Based CSPs. In: da Rocha Costa, A.C., Vicari, R.M., Tonidandel, F. (eds.) SBIA 2010. LNCS, vol. 6404, pp. 233–242. Springer, Heidelberg (2010)
3. Crawford, B., Castro, C., Monfroy, E., Soto, R., Palma, W., Paredes, F.: Dynamic Selection of Enumeration Strategies for Solving Constraint Satisfaction Problems. Romanian Journal of Information Science and Technology 15(2), 106–128 (2012)
4. Crawford, B., Soto, R., Castro, C., Monfroy, E.: Extensible CP-based autonomous search. In: Stephanidis, C. (ed.) Posters, Part I, HCII 2011. CCIS, vol. 173, pp. 561–565. Springer, Heidelberg (2011)
5. Crawford, B., Soto, R., Montecinos, M., Castro, C., Monfroy, E.: A framework for autonomous search in the ecl^{i}ps^e solver. In: Mehrotra, K.G., Mohan, C.K., Oh, J.C., Varshney, P.K., Ali, M. (eds.) IEA/AIE 2011, Part I. LNCS, vol. 6703, pp. 79–84. Springer, Heidelberg (2011)
6. Durán, O., Rodriguez, N., Consalter, L.A.: Collaborative particle swarm optimization with a data mining technique for manufacturing cell design. Expert Systems with Applications 37(2), 1563–1567 (2010)
7. Glover, F., Laguna, M.: Tabu Search. Kluwer Academics Publishers (1997)
8. James, T., Brown, E., Keeling, K.: A hybrid grouping genetic algorithm for the cell formation problem. Computers and Operations Research 34(7), 2059–2079 (2007)
9. Lozano, S., Díaz, A., Eguía, I., Onieva, L.: A one-step tabu search algorithm for manufacturing cell design. Journal of the Operational Research Society 50(5) (1999)
10. Neshat, M., Sepidnam, G., Sargolzaei, M., Toosi, A.: Artificial fish swarm algorithm: a survey of the state-of-the-art, hybridization, combinatorial and indicative applications. In: Artificial Intelligence Review, pp. 1–33 (2012)
11. Nsakanda, A., Diaby, M., Price, W.: Hybrid genetic approach for solving large-scale capacitated cell formation problems with multiple routings. European Journal of Operational Research 171(3), 1051–1070 (2006)
12. Purcheck, G.F.K.: A linear - programming method for the combinatorial grouping of an incomplete set. Journal of Cybernetics 5, 51–58 (1975)
13. Sankaran, S.: Multiple objective decision making approach to cell formation: A goal programming model. Mathematical Computer Modeling 13, 71–77 (1990)
14. Shafer, S.N., Rogers, D.F.: A goal programming approach to cell formation problems. Journal of Operations Management 10, 28–34 (1991)
15. Soto, R.: Controlling search in constrained-object models. In: Kuri-Morales, A., Simari, G.R. (eds.) IBERAMIA 2010. LNCS, vol. 6433, pp. 582–591. Springer, Heidelberg (2010)
16. Soto, R., Crawford, B., Monfroy, E., Bustos, V.: Using autonomous search for generating good enumeration strategy blends in constraint programming. In: Murgante, B., Gervasi, O., Misra, S., Nedjah, N., Rocha, A.M.A.C., Taniar, D., Apduhan, B.O. (eds.) ICCSA 2012, Part III. LNCS, vol. 7335, pp. 607–617. Springer, Heidelberg (2012)
17. Soto, R., Crawford, B., Monfroy, E., Paredes, F.: Syntax Extensions for a Constrained-Object Language via Dynamic Parser Cooperation. Studies in Informatics and Control 21(1), 41–48 (2012)

18. Soto, R., Granvilliers, L.: The Design of COMMA: An Extensible Framework for Mapping Constrained Objects to Native Solver Models. In: Proceedings of IEEE ICTAI, pp. 243–250 (2007)
19. Soto, R., Kjellerstrand, H., Duran, O., Crawford, B., Monfroy, E., Paredes, F.: Cell formation in group technology using constraint programming and boolean satisfiability. Expert Syst. Appl. 39(13), 11423–11427 (2012)
20. Soto, R., Kjellerstrand, H., Gutiérrez, J., López, A., Crawford, B., Monfroy, E.: Solving manufacturing cell design problems using constraint programming. In: Jiang, H., Ding, W., Ali, M., Wu, X. (eds.) IEA/AIE 2012. LNCS, vol. 7345, pp. 400–406. Springer, Heidelberg (2012)
21. Venugopal, V., Narendran, T.T.: A genetic algorithm approach to the machine-component grouping problem with multiple objectives. Computers & Industrial Engineering 22(4), 469–480 (1992)
22. Wu, T., Chang, C., Chung, S.: A simulated annealing algorithm for manufacturing cell formation problems. Expert Systems with Applications 34(3), 1609–1617 (2008)
23. Yang, X.-S., Deb, S.: Cuckoo search via lévy flights. In: World Congress on Nature & Biologically Inspired Computing (NaBIC), pp. 210–214. IEEE (2009)
24. Zhang, C., Li, X., Gao, L., Wu, Q.: An improved electromagnetism-like mechanism algorithm for constrained optimization. Expert Syst. Appl. 40(14), 5621–5634 (2013)