European Symposium on Computer Aided Process Engineering - 11
R. Gani and S.B. Jørgensen (Editors)

An Accelerated Branch-and-Bound Algorithm for Assignment Problems of Utility Systems

Alexandros M. Strouvalis[a], Istvan Heckl[b], Ferenc Friedler[b] and Antonis C. Kokossis[c*]

[a] Department of Process Integration, UMIST, P.O. Box 88, Manchester M60 1QD, UK.
[b] Department of Computer Science, University of Veszprém, Egyetem u.10, Veszprém H-8200, Hungary.
[c] Department of Chemical and Process Engineering, School of Engineering in the Environment, University of Surrey, Guildford, GU2 7XH, UK.

The paper presents a methodology for integrating logic and engineering knowledge within a Branch-and-Bound algorithm with purpose to accelerate convergence. The development addresses assignment problems of utility networks with emphasis on the optimal allocation of units over periods for maintenance. The solver exploits the special structure of the problem to (i) exclude redundant combination of variables, (ii) prioritise the branching of nodes, (iii) provide bounds of nodes and (v) prune inferior parts of the binary tree. Extraction of knowledge and analysis of operations is supported by the graphical environment of the Hardware Composites. Comparisons with commercial MILP solvers demonstrate the merits of customising the solution search engine to the particular solution space.

1. INTRODUCTION

The impact of Mathematical Programming and Optimisation proves significant through a variety of applications in design and operations. The Operations Research community contributed considerable part of the available optimisation tools. At their best, they epitomise general theoretical, computational and numerical knowledge in relevance to the different classes of problems considered. The result is application of general-purpose solvers designed to address formulations ranging from financial problems to chemical process design. Despite numerous efforts the proposed interfaces with solvers exhibit inferior performances as they are not capable of capturing the intricacies of the particular application. In the absence of specific knowledge, the use of general heuristics devotes a large computational effort to redundant searches and formulations that artificially expand the solution space.

The importance of including logic in the modelling stage was highlighted by Raman and Grossmann (1992) who employed a combination of heuristics and logic to solve MINLP problems. The same authors later (1993) used inference logic to branch on decision variables and (1994) implemented logical disjunctions as mixed-integer constraints. Solution of MILP's is mainly addressed through application of the Branch-and-Bound algorithm. The algorithmic efficiency relies on the selection criteria for candidate problems, bounding, pruning and branching (Geoffrion *et al.* (1972)). As Forrest *et al.* (1974) mentioned, important B&B

* Corresponding author.

functions are promptly determined if the user has adequate knowledge of the physical problem. The Hardware Composites (Mavromatis and Kokossis (1998), Strouvalis *et al.* (1998)) are employed to reveal information and insights of utility networks that would otherwise be impractical or expensive to acquire by algorithmic approaches. The Hardware Composites not only assist in customising and tuning solvers but also analyse solution space properties and their computational impact.

2. PROBLEM DESCRIPTION AND MODEL ANALYSIS

The problem considers the maintenance scheduling of turbines and boilers (assignment of tasks to periods). Objective is identification of the optimal sequence to shut-down units for inspection and maintenance with minimum disruption of the utility operation. Switching-off units imposes penalties to objective function as less efficient units or options (*i.e.* power purchase) are employed to compensate for the ones maintained. Optimisation has to consider demand variations over time, differences in efficiencies of units and feasibility aspects. The formulations yield MILP problems with significant number of variables. Even for moderate networks the problem can become highly combinatorial and expensive to solve.

This class of scheduling problems exhibits the block angular structure with periods coupled to each other due to binary variables assigned for the ON/OFF status of units. Binary variables are present in maintenance and individual period constraints. The special model structure is exploited to set up the B&B algorithm. Especially the customised bounding and pruning criteria capitalise on decoupled combinations of maintenance scenarios while investing computational effort on options associated with linking constraints.

3. SOLVER CUSTOMISATION

The customisation spans the main stages of a B&B algorithm. The incorporation of knowledge introduces:

1. Assignment of priorities for the selection of candidate subproblems and branching of variables.
2. Bounding with customised use of the LP solver.
3. Customised tests for minimising the enumerated nodes (enhanced pruning).

The B&B solver is implemented in C++ with application of LINX – a simplex-based routine collection – (Fabian 1992) as LP solver.

3.1. Assignment of Priorities

As units are switched-off, the imposed to objective function penalties vary with the unit efficiency, the efficiency of the units available to replace them and the demands of the particular period. Each period is affected to a different extent and preferences/priorities are strong functions of the layout of demands. High preferences relate to minor alterations in the operation of the utility network. The *period prioritisation* is rigorously defined through calculation of penalties associated with the shut-down of single units in available periods. The basic priority lists are then defined by ordered sets $PL(u) = (P_i, P_j,\ldots,P_n)$ for unit u, with period P_i assigned to higher priority than $P_j,\ldots, P_n$ (u switch-off in P_i contributes less increase to objective function compared to $P_j,\ldots, P_n$).

Priorities are established among units as well. Their definition is based on a hierarchical analysis reflecting the relative importance of turbines and boilers. The visualisation of solution space by Hardware Composites offers qualitative understanding of unit efficiencies and capacity limits in view of specific periods and entire sets of demands. The *unit prioritisation* is included in ordered set PU = (u_a, u_b,...,u_k), with unit u_a assigned with higher priority than u_b,..., u_k (u_a is more important to operation of the system than u_b,...,u_k).

The assignment of priorities (period and unit) is referred to as the *preprocessing stage* and requires the solution of a number of LP's. The resources spent during preprocessing represent the computational cost of prioritising and revealing the structure of the solution space. Based on sets PL(u) and PU the selection of candidate subproblems and branching of nodes is arranged and customised.

3.2. Calculation of Lower Bounds

The B&B solver performs bounding by capitalising on information acquired during preprocessing. Instead of calling the LP solver to estimate bounds at every branched node, a more refined policy of solving LP's is adopted. Enumerated nodes are classified as *dependent* and *independent.* A node is dependent if it involves the shut-down of more than one unit in a period or if this node is infeasible. Otherwise the node is termed independent. Independent nodes relate to decoupled maintenance combinations (units switched-off in different periods). These nodes qualify for having their lower bounds defined by already available and computationally inexpensive information. On the contrary, nodes of coupled operations (dependent) necessitate separate bounding. In that manner the LP solver utilisation results to reduced resources spent on calculation of bounds.

3.3. Enhanced Pruning

Enhanced pruning uses the properties of dependent and independent nodes. For independent combinations tests are made using the priority lists. The tests select the combinations to enumerate and exclude a further enumeration of nodes (as having a guaranteed lower potential). Units associated with identical maintenance periods or infeasibility at a visited node are the ones justified for relaxation of the priority sequence by examining the next period(s) in preference. When a feasible independent terminal node has been reached or an independent node has been pruned then nodes involving lower periods in priority are pruned without enumeration. It is pointed out that enhanced pruning is rigorous and does not compromise on the optimality of the solution.

4. ILLUSTRATION EXAMPLE

The utility system of Fig. 1(a) includes 9 units (steam turbines T_1-T_5, gas turbine GT and boilers B_1-B_3) while operating horizon consists of 24 equal-length periods. Each period relates to a pair of constant demands in power and process steam as shown in Fig. 1(b). Preventive maintenance imposes the shut-down of all units for one period. The optimal scheduling is expected to meet all demands and maintenance needs in the most efficient way. The problem is modelled as MILP with integer variables assigned for the status of units per period. The complete model properties are represented in Table 1.

Table 1: Model size for illustration example.

Continuous Variables	Binary Variables	Constraints	Non-zero Elements
457	216	754	2257

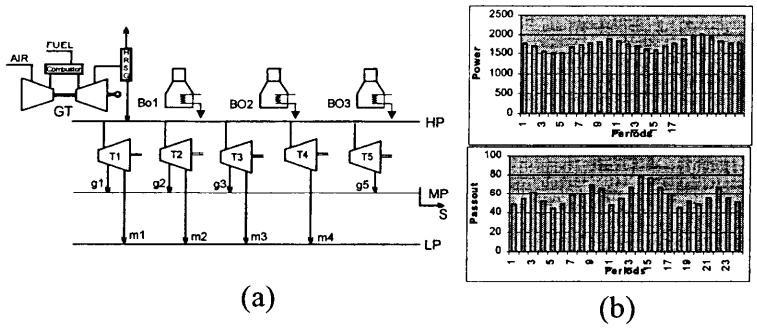

Fig. 1: (a) Utility network and (b) sets of demands in power and heat.

The solution of the problem is addressed in two steps: I) preprocessing of the solution space and II) application of the customised B&B solver.

STEP I:

The solution space is analysed to reveal the feasible and prioritised options for the B&B solver to navigate. Preprocessing is applied in conceptual and computational terms. The location of demands on the solution space (Fig. 3) in relevance to hardware limits identifies infeasible scenarios which are excluded from optimisation. Separate LP's are solved to calculate penalties associated to the shut-down of individual units in feasible periods. Penalties in increasing sequence define priority lists PL(u) for each unit u. The penalties are furthermore used in conjunction to conceptual analysis for defining a qualitative importance of units to operations. In that manner the unit prioritisation list PU is also determined. Ordered sets PL(u) and PU formulate the *solver matrix* (Fig. 2) representing the prioritised solution space. The first column includes all units subject to maintenance arranged according to PU (upper elements – GT, T_2, T_1, T_3,... – relate to higher priority units). Each element-unit of the first column associates to the corresponding period priority list PL(u) defining the rows of the matrix.

STEP II:

The customised B&B searches the solution space capitalising on the structure of the solver matrix. The branching of the binary tree initiates from the first elements of the upper rows and proceeds to deeper options only if specific properties hold. The enhanced pruning effectively disregards inferior parts of the tree from enumeration. The result is acceleration of the B&B algorithm compared to the solution of the same model by OSL implemented in GAMS (Table 2).

GT	5	3	15	14	16	13	6	2	7						
T_2	15	14	4	3	5	16	13	6	2	7	8	17	12	9	1
T_1	5	6	4	2	3	16	13	1	7	24	12	8	17		
T_3	4	5	3	15	6	2	16	13	14	7	12	8	1		
T_5	5	18	6	11	4	1	24	2	12	7	3	23	8		
B_1	5	4	6	1	11	18	24	2	12	3	23	7	20		
B_2	1	6	4	5	11	18	24	2	12	3	23	7	20		
B_3	19	20	10	9	22	21	14	17	23	8	18	15			
T_4	10	21	22	9	17	23	18	14	8	11	24	12			

Fig. 2: The solver matrix reflects the B&B customisation.

Table 2: Computational results for illustration example.

	Customised B&B	OSL(GAMS)
Nodes:	188	50,402
Iterations:		150,000 (interrupted)
Solved LP's:	76	50,404
CPU(sec) - 333 MHz:	13.2	1,339
Objective-($/oper.horizon):	1,021,133.4	1,022,899.6
(Relaxed Objective)	(1,004,690)	(1,005,439.7)

Preprocessing Stage: 217 LP's – 26.2 CPU(sec)

OSL solver invested significant computational effort in searching the solution space. Even at the iteration limit of 150,000 optimality had not been reached due to the suboptimal flat profile the solver was trapped in. Alternatively, the customised B&B performed better in all aspects and identified the optimal schedule after solving (217 + 76) LP's during preprocessing and Branching-and-Bounding respectively. Optimal maintenance is performed according to vector: (GT, T_2, T_1, T_3, T_5, B_1, B_2, B_3, T_4) = (5, 15, 6, 4, 18, 4, 1, 19,10).

5. CONCLUSIONS

This work reports on the advantages observed from the customisation in optimisation applications. Experience has shown that general-purpose solvers fail to capture and profit from special properties of problems. Ad hoc inexpensive solvers prove superior to expensive commercial packages. Customised solution search engines with built-in intelligence and search technology perform better orders of magnitude. The capability to apply the basic B&B functions (branching, pruning) tailored to the structure of the solution space accelerates convergence and reduces computational cost.

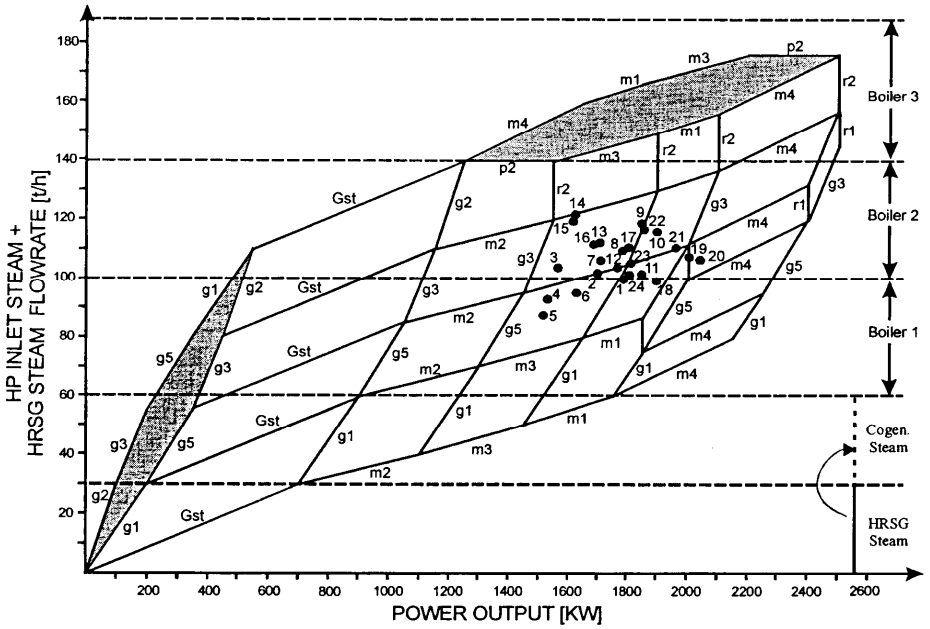

Fig. 3: The Hardware Composites represent the solution space of the utility network.

REFERENCES

1. Fabian, C.I. (1992) LINX: An interactive linear programming library.
2. Forrest, J.J.H., Hirst, J.P.H. and Tomlin, J.A. (1974) Practical solution of the large mixed integer programming problems with Umpire, Management Science, 20(5), 736.
3. Geoffrion, A.M. and Marsten, R.E. (1972) Integer programming algorithms: a framework and state-of-the-art survey, Management Science, 18(9), 465.
4. Mavromatis, S.P., and Kokossis, A.C. (1998). Hardware Composites: a new conceptual tool for the analysis and optimisation of steam turbine networks in chemical process industries – Parts I & II, Chemical Engineering Science, 53(7), 1405.
5. Raman, R. and Grossmann, I.E. (1992) Integration of logic and heuristic knowledge in MINLP optimisation for process synthesis, Computers and Chemical Engineering, 16(3), 155.
6. Raman, R. and Grossmann, I.E. (1993) Symbolic Integration of logic in Mixed-Integer Linear Programming Techniques for process synthesis, Computers and Chemical Engineering, 17(9), 909.
7. Raman, R. and Grossmann, I.E. (1994) Modelling and computational techniques for logic based integer programming, Computers and Chemical Engineering, 18(7), 563.
8. Strouvalis, A.M., Mavromatis S.P., and Kokossis, A.C. (1998). Conceptual optimisation of utility networks using hardware and comprehensive hardware composites, Computers and Chemical Engineering, 22, S175.