ELSEVIER

Computers and Chemical Engineering 24 (2000) 487–493

Computers & Chemical Engineering

www.elsevier.com/locate/compchemeng

Customized solvers for the operational planning and scheduling of utility systems

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

[a] *Department of Process Integration, University of Manchester, Institute of Science and Technology, P.O. Box 88, Manchester M60 1QD, UK*
[b] *Department of Computer Science, University of Veszprém, Egyetem u.10, Veszprém H-8200, Hungary*

Abstract

The paper explains a paradigm for the integration of engineering knowledge with the search strategy of a Branch and Bound algorithm. The optimization is fairly generic and addresses industrial applications comprising power-generating units. The solution concerns the allocation of the units over time and considers expected variations in the heat load and power. The engineering knowledge exploits the Hardware Composites, a conceptual tool for the operation of utility systems. The knowledge is capitalized at three different levels: (i) to exclude redundant combinations of decision variables, (ii) to prioritize the branching of the algorithm, and (iii) to prune the binary tree. Using a non-commercial LP, an MILP solver is designed and compared with highly-valued, state-of-the-art commercial solvers. The comparisons are particularly impressive in that the customized development outperforms the sophisticated packages and accommodates accelerations of at least two orders of magnitude.

Keywords: Operational planning; Utility networks; Branch and bound; Solver design; Hardware composites

1. Introduction

The impact of optimization and mathematical programming becomes evident through a variety of applications in design and operations. The latter typically use general platforms and environments (GAMS, g-PROMS) that employ interfaces with general-purpose solvers. The environments make the applications easy to set up and require little effort from the engineer. The solvers typically constitute contributions from the Operations Research community. At their best, they epitomize general theoretical, computational and numerical knowledge in relevance to the different classes of problems they consider (LP's, MILP's, NLP's, MINLP's). The paper explains significant improvements in integrating knowledge in relevance with the type of application. Still within a generic framework, engineering knowledge is used to prioritize and coordinate the optimization search or simplify the optimization effort.

Previous efforts explain the importance of taking advantage of knowledge at the modeling stage. Raman and Grossmann (1992) reported improvements in solving MINLP problems with a combined use of logic and heuristics. They illustrated their ideas in process synthesis problems (1993), employed inference logic for the branching of the decision variables, and studied the use of logical disjunctions as mixed-integer constraints (1994). Friedler, Tarjan, Huang and Fan (1992) illustrated the potential for dramatic reductions in the solution space. Friedler, Varga and Fan (1995) later introduced a decision mapping approach that is particularly efficient for process synthesis applications. Heever and Grossmann (1999) addressed the solution of MINLP's for multiperiod problems. They employed a disjunctive OA and a disjunctive bilevel decomposition.

The optimization technology is a natural extension of the simulation technology now widely accepted in industry. It might be instructive to recall, however, that simulation earned acceptance and credit with customized algorithms (c.f. the inside-out algorithm for distillation). In a similar vein, optimization solvers are able to incorporate problem information. In the case of MILP's, for example, it's well known that the efficiency of the B&B relies on the selection criteria of the branch-

* Corresponding author. Tel.: +44-161-2004384; fax: +44-161-2367439.

0098-1354/00/$ - see front matter
PII: S0098-1354(00)00505-6

ing variables (Parker & Rardin, 1988; Floudas, 1995). In the absence of specific knowledge, the use of general heuristics cannot guarantee performance tantamount to the difficulty or the actual size of the problem. The paper explains the development of a customized B&B to incorporate conceptual knowledge and reports significant reductions in the computational effort involved. The knowledge is used from the Hardware Composites (Mavromatis & Kokossis, 1998a,b; Strouvalis Mavromatis & Kokossis, 1998) originally proposed as a graphical tool for the analysis and optimization of turbine networks.

2. Problem description and challenges

The problem considers scheduling applications of turbines and boilers with an emphasis on the maintenance of these units. The objective is to identify the optimal sequence to shut down turbines and boilers for inspection and maintenance. Such sequences contribute to the reliability, availability and profitability of the plant. Switching-off units imposes penalties to the objective function. As less efficient units are loaded to compensate for the ones maintained, the optimization has to consider demand variations over time, differences in the units' efficiencies and feasibility aspects. The formulations yield MILP problems with a large number of variables.

Conventional approaches propound the formulation of efficient models to solve with standard commercial solvers. Even for moderate networks, however, the problem assumes prohibitive dimensions. Furthermore, due to minor differences in nearby solutions, the branch and bound tree can be flat and difficult to fathom. Given the efficiencies and the capacities of the units, conceptual information can be used to coordinate the branching, support the bounding and reduce the search space.

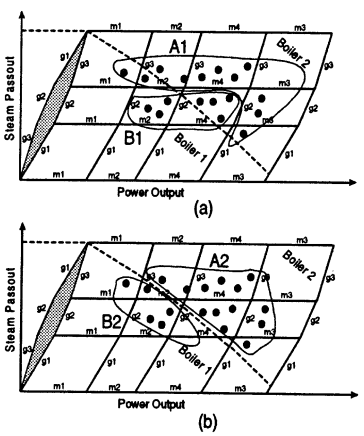

Fig. 1.

3. Solver customization

The customization spans the three main stages of the branch and bound algorithm: the selection of the relaxed variables (candidate selection), the development of the bounds (bounding) and the coordination of the termination tests (pruning). The incorporation of knowledge introduces customized hierarchies for the selection of binary variables and customized tests for minimizing the enumerated nodes. The contributions are explained at two levels:

1. The reduction of the solution space.
2. The tuning and customization of the solver.

The reduction of the solution space is accomplished by screening out infeasible and inferior combinations. This first level makes simple and straightforward use of the engineering information. The second level constitutes a more refined analysis of the knowledge. The basic idea is to replace general-purpose heuristics with a set of priorities to coordinate the branching (candidate selection) and enhance the pruning. Especially the latter significantly decreases the number of LP's to solve.

3.1. Space reduction

The envelope of the Hardware Composites (Mavromatis and Kokossis 1998a,b) encloses all sets of demands and determines the possible maintenance scenarios. The relative position of the demands and the units determines favorable, unfavorable and impossible combinations. Fig. 1 a, b illustrates the Hardware Composites for a network of four steam turbines and two boilers. The steam and power demands are plotted over the considered time horizon; the graphs yield the optimal system provided all units are available for the operation. For the set of demands A1, shown in Fig. 1a, no turbine can be shut down; the high demands in power and steam dictate the participation of all units. Similarly in the set A2 of Fig. 1b no boiler can be switched-off as both are needed to accommodate the high inlet steam turbine flowrates. Contrary to A1 and A2, sets B1 and B2 are candidate periods for maintenance.

3.2. Branching criteria

As units are switched off, penalties in the objective function generally change with the efficiency of the units, with the efficiency of the units available to replace them and with the layout of demands that are serviced at 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 stronger the link a turbine has with a period the less favorable the maintenance of the unit appears in that period.

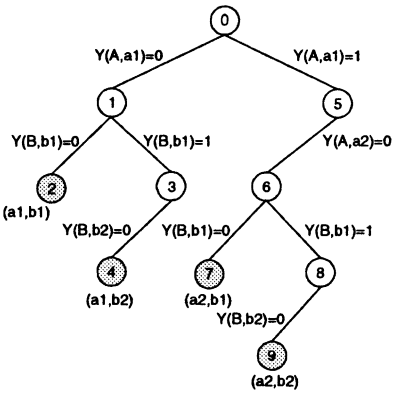

Fig. 2.

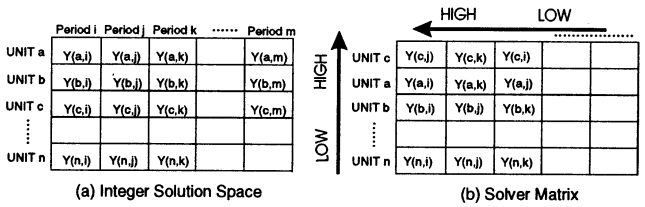

Fig. 3.

The idea can be formalized as follows. Let MP(U) be the set of feasible maintenance periods P for a unit U: MP(U) = {P_i, …, P_j, …, P_k}. Associated with each period is a lower bound that corresponds to the (lowest) operating cost (objective function) that enables the full list of units. The bounds can be calculated by solving a separate LP for each period. Let the set of lower bounds, LB = {lb}, over all periods n of the time horizon be: LB = {lb_1, lb_2, …, lb_i, …, lb_j, …,lb_k, …, lb_n}. For an idle unit in the period k a penalty is assigned calculated by shutting down the unit and evaluating the new objective value, $\mathrm{OB}(U)_k$. The penalty is given by: $\mathrm{PT}(U)_k = (\mathrm{OB}(U)_k - \mathrm{lb}_k)$. Let the set of penalties PE(U) be:

$$\mathrm{PE}(U) = \{(\mathrm{OB}(U)_i - \mathrm{lb}_i),\ \ldots,\ (\mathrm{OB}(U)_j - \mathrm{lb}_j) \times,\ \ldots,\ (\mathrm{OB}(U)_k - \mathrm{lb}_k)\}$$

Sorted in ascending order the set defines the basic priority list PL(U):

$$\mathrm{PL}(U) = \{(\mathrm{OB}(U)_j - \mathrm{lb}_j),\ \ldots,\ (\mathrm{OB}(U)_i - \mathrm{lb}_i) \times,\ \ldots,\ (\mathrm{OB}(U)_k - \mathrm{lb}_k)\}$$

Priorities are established per period for each unit (period prioritization) as well as per unit (unit prioritization). Priority conflicts are sorted by merits of capacity and/or efficiency. Let the set of prioritized units PU be:

$$\mathrm{PU} = \{\mathrm{Unit}_i,\ \mathrm{Unit}_j,\ \ldots,\ \mathrm{Unit}_k,\ \ldots\}$$

with Unit i having higher priority than j and Unit j higher priority than k. The prioritization determines a preprocessing stage for the decision variables $Y(U,P)$ as they are assigned to every unit U and period P. It defines the sequence of branching variables and follows a depth-first approach with backtracking.

Binary variables of high priorities are branched first. Let us illustrate the terminology and the point with two Units A and B and two periods. Let PU = {Unit A, Unit B} and the ordered sets PL(Unit A) = {a_1, a_2}, and PL(Unit B) = {b_1, b_2} to describe the priorities. Fig. 2 represents the branch and bound tree. The number of nodes reflects the branching sequence. The first node corresponds to the unit with the highest priority switched-off (Unit A) in the period of the highest priority (period a_1). The second node is that of the next unit in priority (Unit B) switched-off in the period of its own highest priority (period b_1). The branch and bound search proceeds to the enumeration of the remaining combinations with the same logic. There are four maintenance scenarios (terminal nodes): (Unit A, Unit B) = (a_1, b_1), then (a_1, b_2), (a_2, b_1) and finally (a_2, b_2).

Let us represent the integer solution space by a matrix with rows that correspond to units and columns that relate to periods (Fig. 3a). The prioritization of periods and units respectively defines the sequences over the columns and rows. The matrix of Fig. 3b can be employed by the branch and bound solver to guide the search.

3.3. Enhanced pruning

The enhanced pruning uses properties of dependent and independent periods. Suppose Unit A and Unit B feature the priority lists PL(A) = {a_1, a_2, …, a_i, …, a_n} and PL(B) = {b_1, b_2, …, b_j, …, b_m}. A scheduling combination (Unit A, Unit B) = (a_i, b_j) is dependent if:

1. $a_i = b_j$ or
2. the combination is infeasible.

Otherwise the combination is independent. For independent combinations tests are made using the lists. The tests select the combinations to enumerate and exclude a further enumeration of nodes (as having a guaranteed lower potential). Consequently, nodes that follow independent nodes are pruned. For units with two or more dependent periods the tests can not be applied and conventional branching is required.

The concept is illustrated with the illustration example of Fig. 4. Following the previous discussion, the first enumeration is (Unit A, Unit B) = (a_1, b_1). Let us assume:

Case (I): (a_1, b_1) is independent (feasible and $a_1 \neq b_1$). No other combination is more advantageous: options lower in the list [(a_1, b_2), (a_2, b_1), (a_2, b_2)] are dominated by (a_1, b_1) and can be safely pruned.

Case (II): (a_1, b_1) is dependent ($a_1 = b_1$ and/or infeasible).

Consequently, the search has to explore further nodes. The ones on a_2 and b_2 include the (presumably feasible) independent (a_1, b_2) and (a_2, b_1); the one on (a_2, b_2) can alternatively be rejected.

Priority lists equally apply for larger problems (i.e. more units and periods). The branch and bound search enumerates only options that survive the pruning tests on the nodes. It should be pointed out that pruning rigorous and does not compromise on the optimality of the solution.

4. Computational experiments

4.1. Illustration 1

The utility system (Fig. 5) includes six steam turbines — backpressure passout (turbines 1, 2, 3) and single (turbines 4, 5, 6) — and three boilers. The utility supply produces steam (S stands as the passout flowrate) and power. The maintenance imposes the switch-off on every hardware component (turbine and boiler) for one period. There are 12 periods of operation with their characteristic demands (four per period).

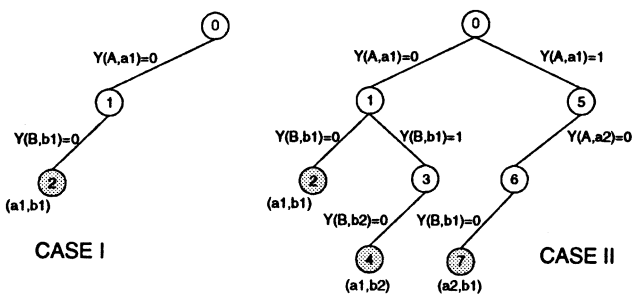

Fig. 4.

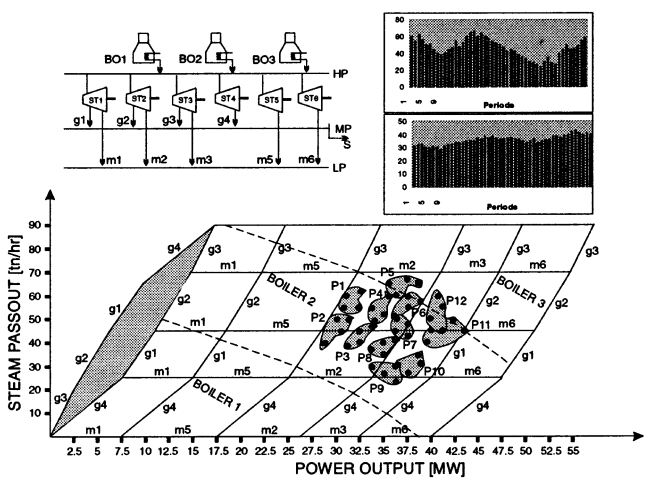

Fig. 5.

The integer variables assigned are 108 [(number of units) × (number of periods)]. The steam turbine operation is described by the inlet steam consumption (1) and that of boilers by their fuel consumption (2):

$$M_i = m_i(E_i) + (1 - m_i/g_i)(S_i) \quad (1)$$

$$F_j = C_j(\mathrm{MS}_j) \quad (2)$$

where, M_i, is the inlet steam consumption of turbine i; E_i, the power output of turbine i; S_i, the steam passout supplied by turbine i; m_i, g_i, characteristic parameters of the turbine thermodynamic efficiency; F_j, the cost of fuel consumption of boiler j, MS_j, the steam flowrate raised by boiler j and C_j, weighed cost coefficient of boiler j.

The objective is assumed to be the minimization of the cost (of fuel) over the entire time horizon.

The optimal solution space is visualized by the Hardware Composites with all units available for operation (Fig. 5). The graph consists of linear segments (m and g) representing the steam turbine expansion paths. The most efficient expansion paths are first fully loaded before less efficient are used. The boiler operation is represented by three zones (dashed lines). The most efficient boiler is first employed (Boiler 1) followed by the less efficient ones (Boilers 2 and 3). Any set of demands can be directly traced on the graph (power output versus passout demand).

The solution is addressed by the solver customization through the solver matrix formulation (prescreening and preprocessing of the solution space). The B&B algorithm is then applied to navigate it and locate the optimal solution. The development of the customized search is explained as follows.

1. Solution space reduction. The understanding of hardware limits and position of demands depicts infeasible scenarios: period 11 requires the operation of turbines 1, 2, 5 and 6, while periods 5, 11, and 12 the operation of all the boilers. Similarly all the remaining infeasible options are identified and removed from the solution space providing the possible maintenance periods sets MP(U) of unit U.
2. Period prioritization. Penalties are calculated for all feasible maintenance scenarios of MP(U) to quantify the deviation from the lower bound of each period. The penalties in increasing sequence define the priority lists for each unit.
3. Unit prioritization. The most important thing for operation (assigned with the highest priorities) units are situated on the bottom-left part of the Hardware Composites and the less efficient on the upper-right part. As such a priority sequence of units extracted is:

T4 > T5 > T1 > B1 > T2 > B2 > T3 > T6

Table 1

Integer variables: 108, Cont. variables: 1021	Customized B&B	GAMS (OSL)
Iterations	624	71 443
Nodes	624	22 034
CPU 366 MHz (s)	285	997
Objective value ($/operating horizon)	1 011 276	1 011 276

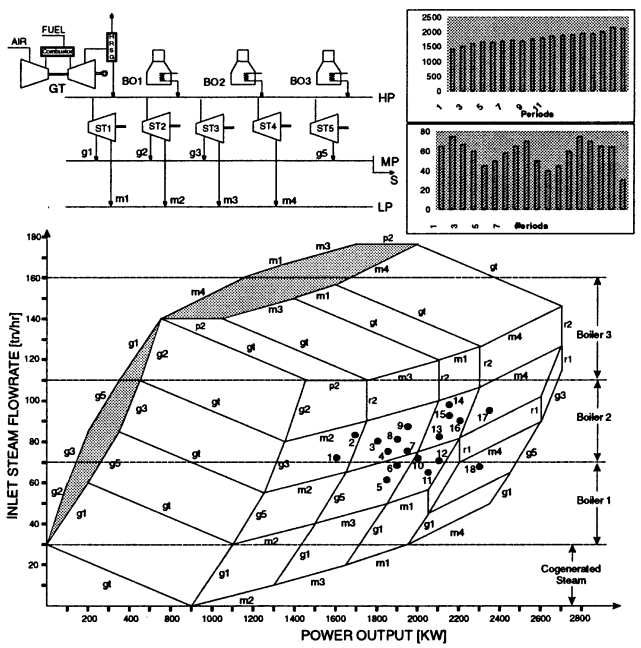

Fig. 6.

This sequence is the result of conceptual analysis supported by the Hardware Composites. Thereafter the PU set is:

PU = {T4, T5, T1, B1, T2, B2, T3, T6}

Having obtained sets PL(U) and PU, the solver matrix is formulated:

T4 3, 2, 4, 9, 1, 8, 10, 7, 6, 12, 11

T5 1, 2, 3, 4, 5, 6, 8, 7, 9

T1 2, 1, 3, 4, 8, 5, 6, 7, 9, 10, 12

B1 9, 2, 3, 8, 10, 7

T2 2, 1, 3, 4, 8, 7, 6, 9, 10, 12

B2 9, 2, 3, 8, 10, 7

B3 6, 10, 7, 4, 9, 8, 1, 3, 2

T6 12, 5, 6, 10, 7, 4, 9, 1, 8, 3, 2

T3 5, 4, 1, 8, 3, 2, 8, 7, 9, 12, 10, 11

The customized solver searches the solution space capitalizing on the structure of the above matrix. The search is initiated from the first elements of upper rows and proceeds to deeper options only if there is a justified reason. The enhanced pruning effectively disregards inferior nodes from enumeration. The result is the acceleration of the B&B compared to the solution of the same model with OSL implemented in GAMS as shown in Table 1. Two orders of magnitude less nodes are enumerated. The optimal maintenance solution vector is shown below (Table 1).

(T4, T5, T1, B1, T2, B2, B3, T6, T3)

= (3, 1, 4, 9, 2, 2, 6, 12, 5)

4.2. Illustration 2

The utility system of Fig. 6 consists of one gas and five steam turbines. The inlet steam is provided both by the three boilers and the Heat Recovery Steam Generator of the gas turbine. The utility network supplies power and process steam (MP steam passout). Maintenance scheduling imposes the shut-down of every unit for one period. The operational horizon consists of 18 periods with one characteristic set of demands per period. The problem requires the assignment of 162 integer variables. The operation of steam turbines and boilers is modeled as in illustration 1. The gas turbine operation is represented by the steam raised in the HRSG (3) and the fuel burnt in the combustion chamber (4):

$$\mathrm{HRS} = A\ (E_g) \tag{3}$$

$$F_g = C_g(\mathrm{HRS}) \tag{4}$$

where, HRS is the steam flowrate raised in the Heat Recovery Steam Generator; E_g, the power output of the gas turbine; F_g, the cost of fuel consumption of the gas turbine, A, the coefficient of the gas turbine efficiency and C_g, weighed cost coefficient of the gas turbine.

The customized search is performed as follows:

1. Solution space reduction. The integer solution space is reduced to the feasible maintenance scenarios for each unit. The screening of infeasible modes provides the sets MP(U) for each unit U.
2. Period prioritization. Following the penalty analysis, priorities are assigned for the shut down of units. Priority lists PL(U) are acquired.
3. Unit prioritization. Based on qualitative analysis, units more important to operations are given higher priorities. The defined set of prioritized units is: PU = GT, T1, B1, T2, T5, T3, B2, B3, T4} transforming the integer solution space to the solver matrix:

Table 2

Integer variables: 162, Cont. variables: 505	Customized B&B	GAMS (OSL)
Iterations	70	100 000 (interger)
Nodes	70	42396
CPU 366 MHz (s)	6	737
Objective value ($/operating horizon)	704 423	705 232

GT	1, 2, 3, 5, 4, 8, 6, 7, 9
T1	5, 6, 1, 4, 8, 3, 7, 2, 9, 10, 11, 12, 13, 15, 16, 14
B1	5, 11, 18, 6, 12, 10, 1, 4, 7, 3, 8, 13, 2, 9, 16, 15, 17, 14
T2	1, 2, 3, 4, 8, 5, 9, 6, 7, 10, 11, 14, 13, 15, 12, 16
T5	11, 5, 12, 6, 18, 10, 1, 7, 4, 8, 3, 9, 13, 2, 15, 14, 16, 17
T3	1, 2, 3, 4, 5, 6, 8, 7, 9, 10, 11, 12, 13, 15, 14, 16
B2	18, 11, 6, 5, 12, 10, 1, 4, 7, 3, 8, 13, 2, 9, 16, 15, 17, 14
B3	17, 14, 6, 15, 18, 13, 9, 12, 10, 7, 8, 2, 11, 3, 4, 6, 1, 51
T4	16, 14, 15, 13, 12, 10, 9, 11, 7, 8, 2, 3, 4, 6, 1, 5

The customized B&B searches for the optimal solution based on the above prioritized solution space. The minimum operational cost is achieved when the units are switched-off as shown in the solution vector:

(GT, T1, B1, T2, T5, T3, B2, B3, $T4$)

$=$(1, 5, 5, 2, 11, 3, 18, 17, 16)

Comparison with OSL solver reveals the reduction in computational needs achieved by the customization of the solution search (Table 2). The OSL search was interrupted at the iteration limit of 100 000 without having found the optimal node. The customized solver identifies the optimal schedule in more than three orders of magnitude less nodes (Table 2).

4.3. Impact of constraints

Four sets of constraints are introduced to the problem. The computational experiment aims at testing the solver on different solution spaces. The presence of constraints makes the node of the optimal solution vector (1, 5, 5, 2, 11, 3, 18, 17, 16) infeasible, imposing the branching of deeper nodes. Depending on the specific set of constraints the solution space is changed at different parts of the B&B tree. The question is whether the customized B&B solver can still enhance the computational efficiency under varying solution space environment. Four experiments are performed: each set of constraints is separately introduced to the model solved by both OSL and the customized solver. The results are as presented in Table 3. In all the cases OSL fails to locate the optimal solution in less than the iteration limit of 100 000. On the contrary, the B&B solver is not influenced by the presence of constraints and exhibits steady efficient performance. At least three orders of magnitude less nodes need enumeration.

5. Discussion

Combinatorial complexity, variation of demands, hardware feasibility limits are factors influencing the solution search efficiency. The proposed B&B is designed to take maximum advantage of the problem's structure. At least two orders of magnitude less nodes have been enumerated before reaching optimality compared to search approaches performed by OSL. Significant CPU time reductions are also reported. Especially for the CPU comparisons it is important to point out that the customized solver is equipped with an aca-

Table 3

	Nodes	CPU on 366 MHz PC (s)	Objective value ($/operating horizon)	Solution vector
Set 1				
Customized B&B	96	8	704, 502.7	(1, 6, 5, 2, 11, 3, 18, 17, 16)
OSL (GAMS)	36647	750	708, 004.6	
SET 2				
Customized B&B	58	5	704, 645.3	(1, 5, 11, 3, 11, 2, 18, 17, 16)
OSL (GAMS)	35253	787	710, 321.7	
SET 3				
Customized B&B	35	3	704, 691.6	(1, 6, 5, 3, 11, 2, 18, 17, 16)
OSL (GAMS)	37377	918	706, 284.5	
SET 4				
Customized B&B	48	4	704, 649.5	(1, 6, 11, 2, 5, 3, 18, 14, 16)
OSL (GAMS)	39614	790	706, 617.2	

demic LP solver. Although it is not as efficient as commercial LP solvers, the overall B&B solver is more efficient than commercial MILP solvers. The difference is justified by the special built-in interface between solver and problem.

The extra pruning stage is drastic allowing for significant parts of the solution space to be deleted and the optimal node is effectively identified (Table 3). The customized solver is capable of profiting from special properties while general-purpose algorithms can not. The Hardware Composites provide the appropriate background for the conceptual analysis of computational aspects and space characteristics disregarded under full solution approaches.

The objective function of operational problems is usually dominated by flat profiles. A pool of similar solutions with minor difference in objective value exists impeding the solver efficiency. Conventional solvers are trapped in local optima. This was the case in illustration 2 where OSL failed to reach optimality in reasonable times. The B&B solver navigates the space in an arranged manner avoiding the impact of such complexities.

6. Conclusions

The paper outlines the development of customized solvers and reports on the advantages observed from the customization in optimization applications. Adhoc inexpensive solvers are shown to be superior to expensive commercial packages. Customized solution search engines with built-in intelligence and search technology perform better orders of magnitude. Their capability to apply branching and pruning tailored to the structure and properties of the solution space rather than using general-purpose heuristics proves particularly effective. The integration of the logic is more essential at the level of the solver and much less at the modeling stage. The customization is applied for the maintenance scheduling of utility networks. Similar MILP solvers are possible to design for a much wider range of applications.

References

Floudas, C. (1995). Nonlinear and mixed-integer optimization, fundamentals and applications. Oxford: Oxford University Press.

Friedler, F., Varga, J. B., & Fan, L. T. (1995). Decision-mapping: a tool for consistent and complete decisions in process synthesis. *Chemical Engineering Science*, *50*(11), 1755–1768.

Friedler, F., Tarjan, K., Huang, Y. W., & Fan, L. T. (1992). Graph-theoretic approach to process synthesis: axioms and theorems. *Chemical Engineering Science*, *47*(8), 1973–1988.

Heever, S. A., & Grossmann, I. E. (1999). Disjunctive multiperiod optimization methods for design and planning of chemical process systems. *Computers & Chemical Engineering*, *23*, 1075–1095.

Mavromatis, S. P., & Kokossis, A. C. (1998a). Hardware composites: a new conceptual tool for the analysis and optimisation of steam turbine networks in chemical process industries — Part I: principles and construction procedure. *Chemical Engineering Science*, *53*(7), 1405–1434.

Mavromatis, S. P., & Kokossis, A. C. (1998b). Hardware composites: a new conceptual tool for the analysis and optimisation of steam turbine networks in chemical process industries — Part II: application to operation and design. *Chemical Engineering Science*, *53*(7), 1435–1461.

Parker, G. & Rardin R. (1988). *Discrete optimization*. New York: Academic Press.

Raman, R., & Grossmann, I. E. (1992). Integration of logic and heuristic knowledge in MINLP optimisation for process synthesis. *Computers & Chemical Engineering*, *16*(3), 155–171.

Strouvalis, A. M., Mavromatis, S. P., & Kokossis, A. C. (1998). Conceptual optimisation of utility networks using hardware and comprehensive hardware composites. *Computers & Chemial Engineering*, *22*(22), S175–S182.

Further Reading

Raman, R., & Grossman, I.E. (1993). Symbolic integration of logic in mixed-integer linear programming techniques for process synthesis. *Computers & Chemical Engineering*, *17*(9), 90–927.

Raman, R., & Grossman, I.E. (1994). Modelling and computational techniques for logic based integer programming. *Computers & Chemical Engineering*, *18*(7), 563–578.