Developments in linear and integer programming

K Darby-Dowman[1]* and JM Wilson[2]

[1]*Department of Mathematical Sciences, Brunel University, Uxbridge, Middlesex UB8 3PH, UK; and*
[2]*Business School, Loughborough University, Loughborough, Leicestershire LE11 3TU, UK*

In this review we describe recent developments in linear and integer (linear) programming. For over 50 years Operational Research practitioners have made use of linear optimisation models to aid decision making and over this period the size of problems that can be solved has increased dramatically, the time required to solve problems has decreased substantially and the flexibility of modelling and solving systems has increased steadily. Large models are no longer confined to large computers, and the flexibility of optimisation systems embedded in other decision support tools has made on-line decision making using linear programming a reality (and using integer programming a possibility). The review focuses on recent developments in algorithms, software and applications and investigates some connections between linear optimisation and other technologies.

Journal of the Operational Research Society (2002) **53,** 1065–1071. doi:10.1057/palgrave.jors.2601435

Keywords: integer programming; linear programming

Introduction

Since the 1950s the ability to solve sizeable linear programming models using the Simplex algorithm of Dantzig[1] has stimulated developments in research on linear programming (LP) and integer programming (IP). As a result, LP and IP models are increasingly used within decision making systems developed by Operational Research practitioners. Efficient software with good model development capability has made the use of LP much more straightforward for practitioners. In parallel with the developments in LP, IP has moved forward rapidly since the pioneering work of Land and Doig.[2] IP models are generally harder to solve than LP models of the same size, but this has not prevented practitioners from developing large sized IP models. From small beginnings, the rapid solving of large models is now commonplace. A survey of recent issues of the journal Interfaces, published by INFORMS, shows widespread application of LP and IP in many different industries. Applications include supply chain management in the motor industry,[3] production scheduling in the brewing industry,[4] aircraft and crew scheduling,[5] asset and liability management,[6] energy management in the utilities sector[7] and network design in the telecommunications sector.[8]

In this review we will discuss developments in LP and IP in two separate main sections, although it should be noted that the two topics influence each other. In a further section, extensions of LP and IP into mixed environments will be considered. Later, software (modelling and solving systems) will be discussed and finally applications—in some senses the real OR perspective—will be considered with emphasis on recent or novel applications, especially those made possible given recent developments in solving technology. It should be noted that in a review such as this, for reasons of brevity, only selected developments and applications will be considered and the choice made reflects, in part, the interests of the reviewers rather than a desire for completeness. For details of further coverage we refer the reader to previous reviews and surveys in Bixby,[9] Bixby *et al*[10] and Todd.[11] The emphasis of this review will be LP and IP but we will briefly discuss constraint programming (CP) and stochastic programming.

Developments in linear programming

Since the developments by Dantzig mentioned in the previous section, there has been considerable impetus given to methods of solving LP problems by the breakthrough of Karmarkar[12] with the development of the interior point algorithm. Bixby[9] gives an interesting account of developments in solving LPs with particular reference to progress during the last decade. He emphatically demonstrates the substantial achievement by stating 'three orders of magnitude in machine speed and three orders of magnitude in solving power: A model that might have taken a year to solve 10 years ago can now solve in less than 30 seconds'. The result is that modern solvers can now routinely solve problems that were, even recently, intractable from a practical perspective. In this section we will look at developments in the Simplex algorithm, interior point methods, criss-cross methods, and algorithmic choice.

*Correspondence: K Darby-Dowman, Department of Mathematical Sciences, Brunel University, Uxbridge, Middlesex UB8 3PH, UK.
E-mail: kenneth.darby-dowman@brunel.ac.uk

Developments in the Simplex Method

Although developed in the 1950s, the Simplex Method has continued to receive the attention of researchers. There have been drives to minimise the effort at each step of the algorithm and to minimise the number of repetitions of each step. For particular problem instances, steepest edge Simplex Methods have been successful. Such methods, discussed in Goldfarb and Reid[13] and subsequently by Forrest and Goldfarb,[14] consider not just the marginal unit effect of introducing a particular variable into the basis but also look at the total effect and choose a variable which will allow more progress to optimality to be made. Reviews of developments appear in Maros and Mitra[15] and Bixby.[9]

Developments in interior point algorithms

In Karmarkar[12] there was indication of two developments, namely that a workable (as opposed to pathological) polynomial algorithm for LP was possible and that algorithms differing completely from the Simplex algorithm had practical value. Many new algorithms followed, making use of the ideas of barrier or potential functions and the computation of an approximate analytic centre of a polytope specified by inequalities rather than the evaluation of the LP objective function at vertices of the feasible region. Recent developments include the work of Tardos,[16] Mizuno *et al*,[17] Vavasis and Ye,[18] Ye *et al*,[19] and Tütüncü.[20] Many aspects of interior point algorithms are described by Ye[21] in which he considers the issues surrounding the implementation of interior point algorithms. There is also a large chapter by Roos and Vial[22] in Beasley[23] on general developments in interior point algorithms. Commercially available LP solvers such as CPLEX and XPRESS-MP include versions of such algorithms as alternatives to the more usual Simplex algorithm.

Criss-cross methods

Following work by Zionts[24] and Bland[25] researchers have developed new methods for selecting pivots in LP solution algorithms. Ultimately this has led to methods which solve LP problems without requiring feasibility of the basis. The so-called criss-cross method has attracted some attention. A finite criss-cross algorithm, combining aspects of the work of Zionts and Bland has been developed independently by Chang,[26] Terlaky[27] and Wang.[28] Because feasibility of the basis is not required, a criss-cross method can be regarded as different from Simplex type methods. A survey on pivot algorithms in general can be found in Terlaky and Zhang.[29] The criss-cross method selects a pivot element from a row and column without resorting to any type of ratio test. Instead criteria such as smallest-subscript, first-in-last-out/last-out-first-in, or most-often-selected-variable are used. The ideas used in criss-cross methods have been inspired by work on matroids and show promise.

Choice of algorithms

It is still not ultimately clear which version of the Simplex Method should be used to solve a particular LP problem or whether an interior point approach would do better (see Gondzio and Terlaky[30] for some views on this issue). Fortunately software (see the later section, developments in software) allows this kind of choice to be made and experimented upon. It is rare that solving an LP problem is a one-off exercise. Because models will usually be solved repeatedly, when data change over time or scenarios have to be evaluated, it will be worth experimenting on sample problems using several different algorithms. LP tends to be a fairly robust technique—if a model remains the same size, but some data values change then the time taken to solve a problem remains fairly constant. LP algorithms are capable of solving very large problems. Bixby[9] observes that interior point or Primal-Dual Log Barrier algorithms 'have emerged as overall the most powerful single algorithm for solving LPs'. However, a major advantage of the Simplex Method occurs when solving an LP from an advanced starting point. Thus, for example, the Simplex Method has a significant role as the LP solver during the branch and bound phase of IP, after the initial LP relaxation has been solved. However, the position is not totally conclusive and the Simplex Method is still capable of outperforming interior point methods on many real life models of all sizes.

Developments in IP

Three common approaches to IP (assuming the problem is one of maximisation and is bounded) are as follows.

Branch and bound

Branch and bound (B&B) adopts a tree search in which the tree development process is characterised by two operations that perform *branching* and *bounding* of the solution space. The root node, P_0 of the tree represents the entire state space $S = S_0$ while subsequent nodes (sub-problems) P_j represent successively smaller partitions S_j of S. The set of all feasible solutions is represented by the set of feasible solutions of the sub-problems associated with the uninvestigated or dangling nodes of the tree.

Branching takes place by selecting a variable, x, with a fractional value ($>k$ but $< (k + 1)$) and eliminating the solution space between the adjacent integer values (k, $k + 1$). Thus two new sub-problems (nodes) are created—one (P1) with the additional constraint $x \leqslant k$ and the other (P2) with $x \geqslant k + 1$.

At a chosen node of the tree, the linear programming relaxation (LPR) of the IP, in which the integrality requirement is dropped, is solved. If there is no feasible solution to the LPR at a node, then the node is terminated. Otherwise, if the solution of the LPR of a sub-problem is integer feasible and its objective function value is greater than the previous lower bound, then the objective function value of this sub-problem is set as a new lower bound for the optimal objective function value of the problem.

After each branching process, those sub-problems with an objective function value smaller than the value of the best integer feasible solution found so far are excluded from further branching. The branching continues until the best integer feasible solution is proven to be optimal.

Branch and cut

For branch and cut (B&C), see for instance Padberg and Rinaldi,[31] let S_1 be the set of feasible (not necessarily integer) solutions to P1 and S_2 for P2. At each stage in the development of the above tree, using the LPR of P1 or P2 or their descendant nodes, we may adjoin a constraint(s), termed a cut(s),

$$\sum_j a_j x_j \leqslant b \qquad (1)$$

to augment the problem defined at a node such that S_3 (set of solutions to the LP relaxation of IP with (1)) (or P1 with (1) or P2 with (1)) has the property

$$S_3 \subseteq S$$

and desirably $S_3 \subset S$ but no *integer* solutions present in S are absent from S_3.

Thus B&C can be seen as contributing cuts at the root node only or at both the root node and its descendants. Clearly when B&C is applied at a descendant node the cut(s) adjoined at a descendant node must not exclude integer solutions at that node or its descendants but may exclude integer solutions valid at the predecessor nodes. Lucena and Beasley[32] describe a B&C algorithm for the minimum spanning tree problem.

Branch and price

With a Branch and Price (B&P) approach, an auxiliary problem is solved to identify columns to be added to the LPR of the IP. This relaxation is then optimised and further sets of columns are identified and considered successively. Thus B&P operates in the style of column generation. Both B&P and B&C can be combined with B&B (and with each other) to provide a comprehensive framework for solving.

Mixed environments

In addition to standard IP there have been developments to link LP with discrete decision making. This has led to a number of developments to be described in this section.

Mixed logical/linear programming

Mixed logical/linear programming (MLLP) introduced by Hooker and Osorio[33] considers the problem of optimising a linear function subject to constraints that are specified as logical conditions. A formulation is:

$$\begin{aligned} &\min \quad cx \\ &\text{subject to} \quad p_j(y,\ h) \rightarrow (A^j x \geqslant a^j),\ j \in J \mid q_i(y,h),\ i \in I \end{aligned}$$

The logical part consists of formulae $p_j(y, h)$ and $q_i(y, h)$ involving atomic propositions $y = (y_1, \ldots, y_n)$ that can be either true or false. A typical formula might be 'y_1 or y_2 (or both) must be true'.

In the formulation the constraint set has a logical part on the right-hand side of the vertical bar and a continuous part (on the left-hand side). The continuous part involves linear inequalities based on A which are controlled by implication ($\rightarrow$) from p. There may also be some variables (h) that can take several discrete values.

An MLLP can be solved in a manner analgous to B&C algorithms used in IP. However, scope exists for moving beyond the use of the linear relaxation to the problem, which is often a poor guide to the solution of the discrete part of the problem. This becomes important for problems involving fixed charges (eg set up costs, warehousing costs). Other approaches to MLLP have been developed by McAloon and Tretkoff[34] who supply easy to use software suitable for smaller problems.

Hybrid integer and constraint programming

The Operational Research community has traditionally modelled discrete optimisation problems as integer programs and used LP-based technology to solve these problems. Constraint Programming (CP) emerged from the Artificial Intelligence community and addressed similar problems. Darby-Dowman and Little[35] examined the performance of each approach on a number of different combinatorial optimisation problems and reported on problem characteristics that may lead to better performance of one technology over the other. CP differs from LP and IP in that it may be an LP- or IP-type model without an objective function, where the emphasis is on satisfying constraints, for example scheduling subject to constraints. CP also permits constraints to be specified in forms more general than linear inequalities, for example by using logical expressions. CP also allows discrete variables within models. CP problems are solved by algorithms that make use of logical inference to develop the search space. The algorithms may also incorporate elements of LP-based and B&B-based techniques.

During the last decade there has been considerable interest in harnessing the strengths of both approaches and developing some form of hybrid approach. CP has powerful inferencing capabilities through constraint propagation whereas IP reduces the search space by repeated solving of LP relaxations.

A major issue in developing a cooperative methodology is the communication between different models of the same problem during the solving process. Ottosson,[36] in his PhD thesis, suggests that a modelling framework be adopted which is designed specifically for a hybrid solver. He proposes the use of mixed constraints that enable inferencing and branching to be carried out. Inferencing within LP comes via the use of cutting planes and, for CP, is achieved by domain reductions. Branching takes place with CP but makes use of information produced by solving LP relaxations.

ILOG (www.ilog.com) have produced OPL Studio which provides users with a single modelling language for LP, IP and CP together with the solving technology to customise search strategies and develop alternative and hybrid approaches.

Other models

There is much connection between IP and the Constraint Satisfaction Problem (CSP). In some senses CSP could be regarded as IP (or LP, in certain cases) without an objective function. A useful review appears in Brailsford *et al.*[37]

Developments in software

The 1990s saw consolidation in the number of 'big employers' in the business of providing LP and IP optimisation systems, together with takeovers or alliances. Dominant software systems such as CPLEX (www.ilog.com) and XPRESS-MP (www.dash.co.uk), operate in an environment that is vibrant and growing but in which there are fewer large competitors than before. CPLEX is now part of the software products company ILOG which has an established presence in the CP market. ILOG recognised the need to augment and enhance their solving technology by having a powerful LP-based solver. XPRESS-MP is a product developed and marketed by Dash Optimization. Many LP end-users solve their problems using the Solver feature of the EXCEL spreadsheet if their models are small, and well-established systems such as LINDO (www.lindo.com) are still popular. However, CPLEX and XPRESS-MP are the leading systems that offer a fully comprehensive modelling and solving tool available on many different platforms and both are widely used.

Because the solving feature of the LP and IP systems has become much more routine in recent years there is much emphasis on the modelling capabilities of systems. Elsewhere in this issue, Mitra *et al*[38] provide a comprehensive review of developments in modelling systems, so we will not dwell further on this matter. Returning to the solver side, it is important to stress that systems are becoming much more flexible in offering the end-user the ability to choose algorithms or to build a personal version of a system to allow switching between algorithms, using an algorithmic tools approach. Some recent discussion of comparison between LP solvers appears in Dolan and Moré.[39]

In the remainder of this section, we will consider some non-standard systems that offer potential for the future and may produce developments that become standard commercial practice.

Condor and NEOS

The Condor project (www.cs.wisc.edu/condor) aims to solve large-scale IP test problems that have so far defeated researchers. The approach uses a set of remote computers simultaneously and splits the problem into separate sections in a form of parallelisation to make use of idle time on these computers. Researchers offer time on 'their' machines by agreement. The project has been successful in solving extremely large problems, including the Seymour problem from MIPLIB (www.caam.rice.edu/ ~ bixby/miplib/miplib.html). While the work described is by nature experimental, there is likely application for the solution of large industrial problems using an in-company network of computers. Associated with Condor, the NEOS project (www.mcs.anl.gov./metaneos and http://neos.mcs.anl.gov) provides a computational grid, or *metacomputer*, for the optimisation community. A large number of commercial and research solver systems are made available to researchers for experimental purposes. The work is described in Czyzyk *et al*[40] and Cropps and Moré.[41] This team effort should help push back the barriers of computational optimisation.

Travelling salesman problem code

The travelling salesman problem (TSP) is one of the most familiar combinatorial OR problems and is the subject of much research (see for instance Junger *et al*[42]). A recent development has been to make available to the research community a TSP research code known as CONCORDE (http://www.keck.caam.nce.edu/concorde.html). Thus a cutting-edge code is readily available from a website. This spirit of co-operation is comparatively new and generally encouraging. The authors of the code, Applegate *et al*,[43] have had success in solving a 15112-city instance of the TSP (details are to be found at http://www.math.princeton.edu/tsp/index.html). This remarkable achievement indicates the power of new developments in IP—parallel computing, use of advanced data structures, incorporation of heuristics and a B&C algorithm—in enabling massively combinatorial problems to be solved.

Applications of LP and IP

It would be impossible in this review to provide comprehensive detail on all the many applications of LP and IP that have been published over the years. Instead we will be selective and consider just two areas where there is particularly active use or potential.

Data envelopment analysis

Developed from work in economics by Farrell,[44] data envelopment analysis (DEA) was made workable in an OR context by Charnes *et al.*[45] This form of DEA allows the computation of the relative efficiencies of decision-making units (DMUs). DEA models can be built to incorporate different assumptions about returns to scale: constant (CRS), variable (VRS), non-increasing (NIRS) and non-decreasing (NDRS) (for further discussion, see Appa and Yue[46]). The use of DEA has grown to such an extent that there is a dedicated website www.deazone. At the heart of DEA is the tool of LP to solve the problems required in the analysis.

A simple CRS DEA model can be formulated as:

$$\begin{aligned} \text{maximise} \quad & \sum_{r=1}^{s} u_r Y_{rk} \Big/ \sum_{i=1}^{m} v_i X_{ik} \\ \text{subject to} \quad & \sum_{r=1}^{s} u_r Y_{rj} \Big/ \sum_{i=1}^{m} v_i X_{ij} \leqslant 1 \quad j = 1, 2, \ldots, n \\ & u_r, v_i \geqslant e \quad \text{for all } r, i \end{aligned}$$

where k is the index of the DMU under discussion, Y_{rj} the amount of the rth output from unit j, u_r the weight of the rth output, X_{ij} the amount of the ith input to unit j, and v_i is the weight of the ith input. There are n DMUs, s outputs, m inputs and e is a small constant.

In this model a typical DMU, indexed by k, endeavours to maximise its efficiency, measured as a ratio of weighted outputs to inputs, by choosing the set of weights to attach to its inputs and outputs. The model is then subjected to constraints ensuring that no other DMU will have a relative efficiency greater than 1.0 if it chose the same weights as unit k. The model must then be run for each value of k to determine the efficiency of each such unit. An efficiency value of less than 1.0 for a particular DMU indicates scope for improvement in that at least one DMU or a combination of other DMUs produces a greater weighted output for the same weighted input.

DEA has been used to analyse efficiency in such sectors as: insurance, retailing, banking, education, and transport. Several papers on the topic appear in the March 2002 (53:3) issue of this journal, devoted to Performance Management. Examples of DEA applications discussed in that issue include applications in sewerage[47] and insurance.[48]

Finance

There are significant applications of mathematical programming in finance, particularly in the area of portfolio selection. Much of this activity stems from the pioneering work of Markowitz[49] who considered the mean and variance of a portfolio's return as representations of the benefit and risk associated with an investment. He proposed a quadratic programming model in which a portfolio is selected whereby a specified expected rate of return is achieved at minimum risk.

Let $N = \{1, 2 \ldots n\}$ be a set of n assets in which one can invest; $r_i, i \in N$, be the expected return from asset i; σ_{ij}, $i, j \in N$, be the covariance between the returns from assets i and j; and R be the desired expected return from the portfolio. Consider the decision variable x_i, $i \in N$, as the proportion of the total investment allocated to asset i. The Markowitz model is given as:

$$\begin{aligned} \min \quad & \sum_{i,j \in N} \sigma_{ij} x_i x_j \\ \text{subject to} \quad & \sum_{i \in N} r_i x_i = R \\ & \sum_{i \in N} x_i = 1 \\ & x_i \geqslant 0, \quad i \in N \end{aligned}$$

A set of efficient portfolios can be obtained by parameterising the desired expected rate of return and solving the model for different values of R. Practical issues and computational aspects of the use of such models in portfolio selection is presented in Jobst *et al.*[50]

During the past 10 years, there have been many developments in the use of stochastic programming as a tool to support asset and liability management (ALM). Stochastic programming deals with uncertainty in mathematical programming models by allowing model coefficients to be defined probabilistically instead of by estimated constant values. These ALM models are multi-stage models and the uncertainty associated with future returns is modelled using a probabilistic generation of event trees to produce a large number of possible outcomes or scenarios. In practice, the stochastic programming models generated tend to lead to very large formulations with hundreds of thousands of variables and constraints. Special purpose solution algorithms that exploit the structure of the models have been developed to good effect.[51–53] Kouwenberg and Zenios[54] give a comprehensive discussion on the issues associated with the use of stochastic programming models in asset liability management.

Conclusions

It is clear that LP and IP, although established techniques with a half-century of history, are still active fields for research and important methods for modelling and solving problems. Both techniques continue to be used in traditional applications such as scheduling and allocation but are also providing a solver capability for new application areas. It seems reasonable to conclude that this buoyant field will continue to develop in the 21st century.

References

1 Dantzig GB (1951). Maximisation of a linear function of variables subject to linear inequalities. In: Koopmans TC (ed). *Activity Analysis of Production and Allocation*. Wiley: New York.

2 Land AH and Doig AG (1960). An automatic method for solving discrete programming problems. *Econometrica* **28**: 497–520.

3 Hahn CK, Duplaga EA and Hartley JL (2000). Supply-chain synchronization: lessons from Hyundai Motor Company. *Interfaces* **30**(4): 32–45.

4 Katok E and Ott D (2000). Using mixed-integer programming to reduce label changes in the Coors Aluminium Can Plant. *Interfaces* **30**(2): 1–12.

5 Desrosiers J *et al* (2000). Air Transat uses ALTITUDE to manage its aircraft routing, crew pairing and work assignment. *Interfaces* **30**(2): 41–53.

6 Mulvey JM, Gould G and Morgan C (2000). An asset and liability management system for Towers Perrin—Tillinghast. *Interfaces* **30**(1): 96–114.

7 Hobbs M *et al* (2001). Rebuilding the coal model in the Energy Information Administration's national energy modelling system. *Interfaces* **31**(5): 24–42.

8 Shortle JF *et al* (2001). Optimal design of a data-offload network. *Interfaces* **31**(5): 4–12.

9 Bixby RE (2002). Solving real-world linear programs: a decade and more of progress. *Opns Res* **50**: 3–15.

10 Bixby RE *et al* (2000). MIP: theory and practice—closing the gap. In: Powell MJD and Scholtes S (eds). *System Modelling and Optimization: Methods, Theory and Applications*. Kluwer Academic Press: New York, pp 19–49.

11 Todd MJ (2002). The many facets of linear programming. *Math Program* **91**: 417–436.

12 Karmarkar NK (1984). A new polynomial-time algorithm for linear programming. *Combinatorica* **4**: 373–395.

13 Goldfarb D and Reid JK (1977). A practicable steepest-edge simplex algorithm. *Math Program* **12**: 361–371.

14 Forrest JJ and Goldfarb D (1992). Steepest-edge simplex algorithm for linear programming. *Math Program* **57**: 341–374.

15 Maros I and Mitra G (1996). Simplex algorithms. In: Beasley JE (ed). *Advances in Linear and Integer Programming*. Oxford University Press: Oxford, pp 1–46.

16 Tardos E (1986). A strongly polynomial algorithm to solve combinatorial linear problems. *Opns Res* **34**: 250–256.

17 Mizuno S, Todd MJ and Ye Y (1993). On adaptive-step primal dual algorithms for linear programming. *Math Opns Res* **18**: 964–981.

18 Vavasis S and Ye Y (1996). A primal-dual accelerated interior point method whose running time depends only on A. *Math Program* **74**: 79–120.

19 Ye Y, Todd MJ and Mizuno S (1994). An O $(\sqrt{n}L)$ iteration homogeneous and self-dual linear programming algorithm. *Math Opns Res* **19**: 53–67.

20 Tütüncü RH (1999). An infeasible-interior-point potential reduction algorithm for linear programming. *Math Program* **86**: 313–334.

21 Ye Y (1997). *Interior Point Algorithms: Theory and Analysis*, Wiley-Interscience: New York.

22 Roos C and Vial JP (1996). Interior point methods. In: Beasley JE (ed).. *Advances in Linear and Integer Programming*. Oxford University Press: Oxford, pp 47–102.

23 Beasley JE (ed) (1996). *Advances in Linear and Integer Programming*, Oxford University Press: Oxford.

24 Zionts S (1969). The criss-cross method for solving linear programming problems. *Mngt Sci* **15**: 426–445.

25 Bland RG (1977). New finite pivoting rules for the simplex method. *Math Opns Res* **2**: 103–107.

26 Chang YY (1979). Least index resolution of degeneracy in linear complementary problems. *Technical Report 79-14*, Department of Operations Research, Stanford University.

27 Terlaky T (1985). A convergent criss-cross method. *Math Oper Stat, Ser Optim* **16**: 683–690.

28 Wang Z (1997). A conformal elimination-free algorithm for oriented matroid programming. *Chin Ann Math* **8**: 120–125.

29 Terlaky T and Zhang S (1993). Pivot rules for linear programming: a survey on recent theoretical developments. *Ann Opns Res* **46**: 202–233.

30 Gondzio J and Terlaky T (1996). A computational view of interior point methods. In: Beasley JE (ed). *Advances in Linear and Integer Programming*. Oxford University Press: Oxford, pp 103–144.

31 Padberg M and Rinaldi G (1987). Optimization of a 532 city symmetric travelling salesman problem by branch and cut. *Opns Res Lett* **6**: 1–7.

32 Lucena A and Beasley JE (1996). Branch and cut algorithms. In: Beasley JE (ed). *Advances in Linear and Integer Programming*. Oxford University Press: Oxford, pp 187–221.

33 Hooker JN and Osorio MA (1999). Mixed logical-linear programming. *Discrete Appl Math* **96–97**: 395–442.

34 McAlloon K and Tretkoff C (1996). *Optimization and Computational Logic*, Wiley: New York.

35 Darby-Dowman K and Little J (1998). Properties of some combinatorial optimization problems and their effect on the performance of integer programming and constraint logic programming. *INFORMS J Comput* **10**: 276–286.

36 Ottosson G (2000). *Integration of Constraint Programming and Integer Programming for Combinatorial Optimization*. Uppsala Thesis in Computing Science 33.

37 Brailsford CS, Potts CN and Smith BM (1999). Constraint satisfaction problems: algorithms and applications. *Eur J Opl Res* **119**: 557–581.

38 Dominguez-Ballesteros B, Mitra G, Lucas CA and Koutsoukis N-S (2002). Modelling and solving environments for mathematical programming (MP): a status review and new directions. *J Opl Res Soc* **53**: 1072–1092.

39 Dolan ED and Moré JJ (2002). Benchmarking optimization software with performance profiles. *Math Program* **91**: 201–213.

40 Czyzyk J, Mesnier MP and Moré J (1998). The NEOS Server. *IEEE J Comput Sci Engng* **5**: 68–75.

41 Cropps W and Moré J (1997). Optimization environments and the NEOS Server. In: Buhmann MD and Iserles A (eds). *Approximation Theory and Optimization*. Cambridge University Press: Cambridge, pp 167–182.

42 Junger M, Reinelt G and Rinaldi G (1997). The travelling salesman problem. In: Dell Amico M, Maffioli F and Martello S (eds). *Annotated Bibliographies in Combinatorial Optimization*. Wiley: Chichester, pp 199–222.

43 Applegate D, Bixby R, Chvatal V and Cook W (1998). On the solution of travelling salesman problems. *Doc Math J Deutschen Mathematiker-Verein* **ICMIII**: 645–656.

44 Farrell MJ (1957). The measurement of productive efficiency. *J R Stat Soc A* **120**: 253–282.

45 Charnes A, Cooper WW and Rhodes E (1978). Measuring the efficiency of decision making units. *Eur J Opl Res* **2**: 429–444.

46 Appa G and Yue M (1999). On setting scale efficient targets in DEA. *J Opl Res Soc* **50**: 60–69.

47 Thanassoulis E (2002). Comparative performance measurement in regulation: the case of English and Welsh sewerage services. *J Opl Res Soc* **53**: 292–302.

48 Meimand M, Cavona RY and Laking R (2002). Using DEA survival analysis for measuring performance of branches in New Zealand's Accident Compensation Corporation. *J Opl Res Soc* **53**: 303–313.

49 Markowitz HM (1952). Portfolio selection. *J Finance* **7**: 77–91.

50 Jobst NJ, Horniman MD, Lucas CA and Mitra G (2001). Computational aspects of alternative portfolio selection models in the presence of discrete asset choice constraints. *Quant Finance* **1**: 489–501.

51 Kouwenberg R (2001). Scenario generation and stochastic programming models for asset liability management. *Eur J Opl Res* **134**: 279–292.

52 Mulvey JM and Vladimirou H (1992). Stochastic network programming for financial planning problems. *Mngt Sci* **38**: 1643–1664.

53 Nielson SS and Zenios SA (1996). A stochastic programming model for funding single premium deferred annuities. *Math Program* **75**: 177–200.

54 Kouwenberg R and Zenios SA (2002). Stochastic programming for asset and liability management. In: Zenios SA and Ziemba WT (eds). *Handbook of Asset and Liability Management*. Elsevier Science: Amsterdam.

Received January 2002;
accepted April 2002 after one revision