Binary programming: A decision rule for selecting optimal *vs* heuristic techniques

F. P. Wyman

Department of Management Science and Organizational Behavior, College of Business Administration, The Pennsylvania State University, 120 Boucke Building, University Park, Pennsylvania 16802, USA

Management scientists have studied heuristic programming for some time, but usually with some degree of apology due to an inability to prove the optimality of their heuristic results. This paper illustrates how heuristic integer programming may be seriously studied inductively, with the purpose of developing a systematic body of propositions concerning computational properties of integer programming techniques. A decision rule is formulated for choosing between optimal and heuristic techniques on the basis of a more nearly global criterion for mathematical programming: cost of computation plus the value of the objective function.

An optimising 0-1 code is compared to a recently-developed heuristic 0-1 technique with respect to objective function value and computer time. The comparison is performed by varying three factors of problem structure: number of constraints, number of variables and constraint severity on a total of 180 problems. Regression models are postulated and fitted to the data with promising results. Breakeven charts are developed so that a user with a given problem size, cost of computation, and objective function may rapidly determine the preferability of heuristic-versus-optimal technique. The results indicate that the heuristic solutions are practically identical to the optimal solutions while requiring about one-tenth of the computational effort. The results also support the notion that a heuristic will eventually dominate any optimising technique for a sufficiently difficult problem.

(Received July 1971)

Management scientists have studied heuristic programming for some time, but usually with a degree of apology due to an inability to prove the optimality of their heuristic results. It is beginning to appear that the difficulty of generating knowledge deductively via theorem, lemma, and proof may in certain cases be practically insurmountable. A relevant example is the bewildering behaviour of integer programming codes in solving varying problem structures. This paper attempts to demonstrate that heuristic programming and problem structures may be seriously studied inductively with the purpose of developing a systematic body of propositions regarding the computational properties of integer programming techniques*.

In the December, 1968 issue of *Management Science*, Senju and Toyoda (1968) presented a heuristic technique for solving a special class of 0–1 integer programming problems of the form:

$$\text{Max } \mathbf{Cx} \tag{1}$$

$$\text{subject to } \mathbf{Ax} \leq \mathbf{b} \tag{2}$$

where $b_i \geq 0$, for each constraint **i**;

$a_{ij} \geq 0$, for each constraint **i**, and each variable **j**;

and $x_{ij} = (0, 1)$, for each constraint **i**, and each variable **j**

In particular, their method is designed for any type of problem in which an optimal mix of jobs, projects, investments, or products is desired subject to a great many 'less-than-or-equal-to' constraints on scarce resources such as facilities, labour, available funds, inventory, etc. Although the method yields only approximately optimal results, the authors argue that errors in collecting data may make the degree of non-optimality inconsequential in the final analysis. Furthermore, this problem occurs frequently enough in a wide variety of industrial settings to make it an important sub-class of 0–1 problems.

The Senju–Toyoda heuristic requires only the computation of one inner product per variable plus some ranking operations. The computation is so delightfully simple and reliable with respect to computer time, that one might be tempted to use the technique exclusively despite its approximate nature. It is especially appealing since the computation time for most integer programming codes have frequently been observed to exhibit highly erratic computational behaviour. Nevertheless, a heuristic technique by definition lacks proof of optimality and may, in many cases, be no better or even worse than the heuristics evolved by a human decision-maker. It is surprising that no attempt to estimate the degree of suboptimality of the Senju–Toyoda heuristic has yet been reported. There are two important criteria of such an evaluation: the percent deviation of the objective function from optimality and the percent of computation time saved by the heuristic. Furthermore, it would be useful to know how these two criteria of performance are affected by problem size and by level of constraint severity.

Research objectives

The objective of this paper is twofold: to report the type of analysis and comparison suggested above and to illustrate how heuristic and optimal techniques may be compared to determine which is preferable in a particular situation.

With respect to the latter objective, we would like to be able to predict which technique would be the more rational to use for a given problem. Such a determination could be made in terms of the actual dollar value of the objective function relative to the value of computer time. At this point, the following notation will be useful:

- Z_u: Optimal value of the unconstrained objective function
- Z_o: Optimal value of the constrained objective function
- Z_h: Heuristic value of the constrained objective function
- T_o: Time in seconds of computation for the optimal technique
- T_h: Time in seconds of computation for the heuristic technique
- K: Cost per second of computation time
- PCT_z: Heuristic Z as a percent of the optimal Z
- PCT_t: Heuristic T as a percent of the optimal T

The condition under which the marginal value of the optimising technique exceeds its marginal cost can now be stated simply as

$$(Z_o - Z_h) > (T_o - T_h)\,K \tag{3}$$

*The author is grateful to a referee for pointing out that the heuristic and optimising techniques are not necessarily mutually exclusive alternatives. In branch and bound methods for large practical problems unattractive branches may be discarded prior to complete fathoming. As the referee states: 'This is, therefore, a cross between a heuristic and an optimising technique and one can vary the thoroughness of the search with the importance and difficulty of the problem'.

Downloaded from https://academic.oup.com/mnsc/article/16/2/155/2658 by guest on 23 August 2022

It might be feasible to predict Z_o and Z_h based on three factors of problem structure: A number of constraints; B number of variables; and C degree of 'slackness' in the constraint system. Slackness is taken to mean the average ratio of the b_i value to the sum of the a_{ij} over the ith constraint. The smaller the b_i, the more stringent the constraint hence forcing more variables to equal zero. Thus, the following questions were posed to assist the accomplishment of the research objectives

1. How can Z_o be estimated as a function of A, B, and C?
2. How can Z_h be estimated as a function of A, B, and C?
3. How can T_o be estimated as a function of A, B, and C?
4. How can T_h be estimated as a function of A, B, and C?

For a specific problem, one can readily determine K, the problem dimensions, and the degree of slackness. An analyst could use the estimating equations implied above to determine the preferability of an optimal versus heuristic technique.

A related objective of this study was to illustrate how a set of charts can be developed to depict the optimal and heuristic regions for various combinations of Z_u and K. Graphically, the tradeoff relation would appear something like **Fig. 1**, with the lower right region denoting combinations of Z_u and K in which the expected payoff of the heuristic scheme dominates the optimal approach. For a given Z_u, the heuristic would become more and more attractive as K, the cost of computer time, increases. But for a given K, the optimising approach could become more attractive as Z_u increases.

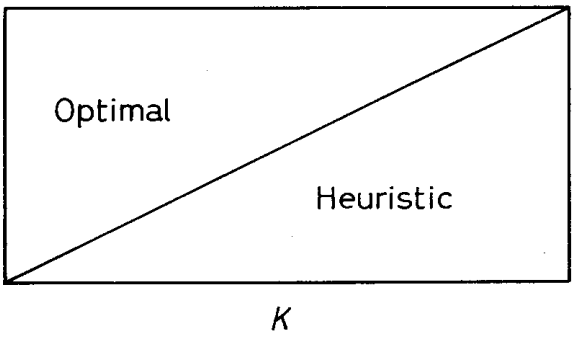

Fig. 1. Cost per second of computer time

Methodology

A wide variety of optimising 0–1 codes are available, several having been reported in a recent study of computational effectiveness (Trauth and Woolsey, 1969). Although several such codes might have been used in the present study, it was the objective of this paper to illustrate a general methodology rather than to report an exhaustive survey. Thus, the well-known code, RIP30C, written by Geoffrion and Nelson (1968), was selected due to its efficiency and availability. This code adds the features of surrogate constraints (Glover, 1965) and an imbedded linear program (Geoffrion, 1968) to accelerate the Balasian implicit enumeration method (Balas, 1965). The Senju–Toyoda method was written in FORTRAN IV using fewer than 50 statements.

Due to the variability in performance of integer programming codes, a restricted problem was intentionally selected to reduce the general variations, which might obscure the main and interacting effects of the postulated factors.

All problems were generated randomly with objective and constraint function coefficients being uniformly distributed between 0·0 and 40·0. Nine problem sizes were selected. The number of constraints were studied at three levels: 10, 20 and 30. The number of variables were also studied at three levels: 15, 30 and 45. In addition to problem size, the degree of constraint 'slackness' was studied at four levels: 30 per cent, 50 per cent, 70 per cent, and 90 per cent. For example, in the 30 per cent case, the b_i for *each* constraint was set equal to 30 per cent of the sum of the a_{ij}, and likewise for the other three levels of slackness. For a given problem, each constraint had an equal degree of constraint severity.

Thus a factorial research design was used with a total of 36 'cells' for the combinations of three levels of factor A, three of B, and four levels of factor C. Five problems were generated and solved by both methods for each cell for a total of 180 problems. The cell size of five was chosen to provide errors of $\alpha = 0{\cdot}01$, and $\beta = 0{\cdot}01$, for not detecting a true difference of 10 per cent in each factor. Sample variance was estimated on the basis of a pilot study. Although the implication for the use of analysis of variance was clear, objectives of the study involved estimation of parameters of somewhat obvious relationships. However, analysis of variance was used simply to evaluate the adequacy of the initially conceived regression models (which are discussed in the following section). Once the data were gathered, analysis of variance was used for each relationship to assess:

(*a*) relative magnitude of main versus interaction effects;
(*b*) significance of non-linear trend in the main effects;
(*c*) significance of non-linear components of the interaction effects.

The regression models were modified where both the pattern of cell means as well as theoretical considerations clearly indicated the validity of doing so. Such modifications involved the addition of interaction terms including AB, AC, BC or BC^{-1}. Finally, the stepwise regressions were performed to develop estimating equations for the components of the decision rule (3). Trade-off charts for optimal-versus-heuristic were then developed for the four extreme 'corners' of the research design.

For purposes of operationalising the concepts of the decision rule, the incremental cost of using the optimising code was defined simply as the cost of additional CPU time. We assume that the two approaches cost approximately the same amount in all other respects such as cost of programming and implementation, use of peripheral equipment, systems time, program storage, etc.

A word should be said about the measurement of computation times. Since the computer system used existed in a multi-processing environment*, interrupts from other jobs caused the usual time functions to be highly unreliable. Furthermore, the typical functions return values accurate only to the nearest integer second. A special subroutine was written† to interrogate the computer's accounting clock, accurate to one/9,600th of a second. No input or output was executed between pre and post timings, since they generate interrupts. Apparently, concurrent interrogations of the clock by other users can have a slight effect on updating of elapsed time. Repeated timings of a simple set of instructions revealed a range of variation in CPU times equal to about three percent (3%) of the mean.

Expected shapes of curves

First let us consider the formulation of T_o. Since the Balasian approach requires either explicit or implicit enumeration of all 2^n solutions, T_o was expected to exhibit an exponential relation with factor B: the number of variables. In every explicit enumeration, each constraint must be evaluated. Since the number of explicit enumerations grows exponentially with factor B, it would appear logical that T_o would also be exponentially related to AB, the *size* of the constraint system. Since the Balasian technique begins with the infeasible solution vector that would yield the optimal unconstrained solution, it

*The PSU Computation Centre is equipped with an IBM 360/67 with several hundred remote batch and remote typewriter entry points.
†The author would like to acknowledge the helpful assistance of John R. Harris in overcoming this problem.

Downloaded from https://academic.oup.com/comjnl/article/16/2/135/372658 by guest on 20 August 2022

requires the least time to solve problems with the least degree of infeasibility. Hence, T_o would be inversely related to the factor C, the degree of constraint slackness. In fact, for equal amounts of increase in constraint severity, one should find that the number of solutions to be enumerated grows exponentially. Hence, we suggest that T_o would be *negative exponentially* related to factor C. The model for T_o is then

$$T_o = \exp(\alpha_o + \alpha_1 A + \alpha_2 B - \alpha_3 C + \alpha_4 AB) \quad (4)$$

and by taking the logarithm (4), we have

$$\ln T_o = \alpha_o + a_1 A + \alpha_2 B - \alpha_3 C + \alpha_4 AB \quad (5)$$

Next, let us consider the model for T_h. The heuristic initially requires computation of one inner product per variable. Feasibility can then be obtained by setting to zero the variable with the minimum 'effective gradient' (Senju and Toyoda, 1968). Suboptimality is decreased if the gradients are recomputed following each variable deletion. Such recomputation requires $n(n+1)/2$ inner products if all variables are ultimately deleted. The number of variables deleted depends upon factor C, percent constraint slackness, such that $(1 - C^2)[n(n+1)/2]$ is a reasonable estimate of the number of inner products computed. Hence, T_h is related to the square of both factors B and C. The time of inner-product computation is also related to factor A, the number of constraints, and to AB as well. The model selected for T_h was, therefore

$$T_h^{\frac{1}{2}} = \alpha_o + \alpha_1 A + \alpha_2 B - \alpha_3 C + \alpha_4 AB \quad (6)$$

Z_u increases linearly with factor B since the very addition of variables obviously increases the unconstrained value of Z. Z_o was also expected to increase linearly over factor B. Z_o was anticipated to be directly related to constraint slackness. Z_h was anticipated to be almost identical to Z_o so that they were modeled as

$$Z_o = \alpha_o + \alpha_1 B + \alpha_2 C \quad (7)$$

$$Z_h = \alpha_o + \alpha_1 B + \alpha_2 C \quad (8)$$

where the regression coefficients would be unique for each model.

Results

General

The Senju–Toyoda heuristic was found to give surprisingly accurate results in a small fraction of the time necessary to execute the optimising code. Over all 180 problems, the heuristic solution averaged 98·6 per cent of the optimal solution while taking an average of 9·34 per cent of the time required by RIP30C. In 37 per cent of the cases, the heuristic found the optimal solution. Such performance would undoubtedly be satisfactory in a great many industrial situations. In fact, if the measurement error of the expected payoff of projects were as much as three percent (3%) of the true mean, then the difference between the payoffs of the different solutions would be statistically indistinguishable.

Neither PCT_t nor PCT_z was found to vary significantly over factor A, the number of constraints. **Tables 1** and **2** show cell means for objective function values and computation times respectively, for factors B and C. **Table 3** summarises the data in percentage terms.

In general, the efficiency of the heuristic actually *increased* as the problems became more difficult. PCT_z was not substantially influenced by the number of variables, but PCT_t decreased significantly. T_o increased much more rapidly than T_h over factor B. As C varied, PCT_z was significantly influenced. The more constraining the problem, the lower the PCT_z achieved by the heuristic.

However, the heuristic performance with respect to time *improved* as the problem became more constrained, apparently because T_o also increased more rapidly than T_h for the more

Table 1 Optimal-versus-heuristic mean values of objective function (Z_o: Z_h)

		Factor C				
	Levels	1	2	3	4	Overall
Factor B	1	111·9 109·2	182·5 179·6	234·2 233·5	270·9 269·9	199·9 193·0
	2	252·7 244·1	395·8 385·8	502·5 498·2	569·1 567·0	430·0 423·8
	3	413·8 399·1	639·0 632·6	799·7 794·0	890·7 889·4	685·8 678·8
Overall		259·5 250·8	405·7 399·3	512·1 508·5	576·9 575·4	438·5 433·5

Table 2 Optimal-versus-heuristic mean computation times in seconds (T_o: T_h)

		Factor C				
	Levels	1	2	3	4	Overall
Factor B	1	1·008 0·106	0·872 0·081	0·543 0·058	0·209 0·032	0·658 0·069
	2	31·198 0·389	13·781 0·299	3·988 0·198	0·705 0·095	12·418 0·245
	3	191·260 0·854	100·103 0·638	23·989 0·420	4·048 0·186	79·850 0·525
Overall		74·489 0·449	38·251 0·339	9·507 0·225	1·654 0·104	30·975 0·279

Table 3 Heuristic as percent of optimal: Value of objective function and computation time (PCT_z: PCT_t)

		Factor C				
	Levels	1	2	3	4	Overall
Factor B	1	97·6 13·8	98·5 12·6	99·6 14·6	99·6 18·0	98·8 14·8
	2	96·5 2·95	97·6 4·65	99·1 8·93	99·6 17·4	98·2 8·42
	3	96·5 0·71	99·0 1·67	99·3 4·26	99·8 12·5	98·6 4·75
Overall		96·9 5·83	98·3 6·31	99·4 9·26	99·7 16·0	98·6 9·34

Table 4 Scale of measurement for main factors

Factor	Scale value (level of factor) 1	2	3	4	Units	Transformation of scale*
A	10	20	30	—	Constraints	$A = A'/10$
B	15	30	45	—	Variables	$B = B'/15$
C	30%	50%	70%	90%	$(b_i/\sum a_{ij})$	$C = (C' - 10) \div 20$

*A′, B′, C′ refer to actual values of independent variables.

Downloaded from https://academic.oup.com/comjnl/article/16/2/135/372658 by guest on 20 August 2022

Table 5 Dollar tradeoffs of Z_u for choosing optimal *vs.* heuristic techniques*

CONSTRAINT SLACKNESS	COST OF COMPUTATION PER SECOND ($)	NUMBER OF 0–1 VARIABLES ($) 15	45
30%	0·01	0·87	112·60
	0·10	8·74	1,126·00
	1·00	87·40	11,260·00
90%	0·01	0·46	19·31
	0·10	4·59	193·10
	1·00	45·90	1,931·00

*If Z_u is less than the indicated amount, then Senju-Toyoda heuristic is preferred to the optimising code (RIP30C) on the basis of computation cost plus the cost of suboptimality.

severely constrained problems. By carefully inspecting Table 3, the reader will notice that PCT_t rises more rapidly over factor C for higher levels of factor B. Hence, an interaction effect of the form BC^{-1} seems to be operating which indicates that PCT_t *improves non-linearly* as problems become more complex.

The most difficult problems for both techniques are those designated by cell B_3C_1 (45 variables with 30 per cent slackness). PCT_z and PCT_t are both minimal for this cell in Table 3. Note that any movement from cell B_1C_4 (highest PCT_t and PCT_z) produces more of an absolute decrease in PCT_t than PCT_z. Referring to decision rule (3), the reader will note that if the heuristic is economically preferable to the optimiser for problems of the type B_1C_4, then the heuristic will actually become *more preferable* as the problems become *more complex*. Furthermore, if the heuristic is not preferable for a given K and Z_u, these data strongly suggest that the heuristic will eventually become economically preferable for sufficiently large and severely constrained problems.

Regression analysis

T_o. The analysis of variance of the transformed values of computation time for RIP30C ($\ln T_o$) revealed that the assumed model was quite adequate except for a significant BC interaction. Examination of the BC cell means showed that upward slope of $\ln T_o$ versus factor B grew steeper for successively lower values of factor C. Such an interaction is theoretically reasonable since a more severely constrained problem multiplies the number of operations per variable (checking for feasibility, back-tracking, etc.). Hence, the term BC^{-1} was added for the stepwise regression, yielding the following result

$$\begin{array}{lllllll} & \ln T_o = -1{\cdot}30 & +2{\cdot}02B & -1{\cdot}53C & +0{\cdot}258AB & -0{\cdot}83BC^{-1} & (9)\\ \text{S.E.:} & (0{\cdot}844) & (0{\cdot}17) & (0{\cdot}117) & (0{\cdot}035) & (0{\cdot}16) & \\ \text{F(4,175):} & 245{\cdot}05 & 132{\cdot}18 & 170{\cdot}40 & 52{\cdot}40 & 24{\cdot}29 & \\ R^2 = 0{\cdot}848 & & & & & & \end{array}$$

The $\ln(T_o)$ values ranged from approximately -2 to $+5$. The scale of measurement for the main factors is shown in **Table 4.** Plots of residuals versus the three main factors as well as the regression estimates exhibited normality, homogenity of variance, and absence of curvilinearity. Factor A, the number of constraints, did not emerge as a significant term as anticipated. Apparently, the *number* of constraints *per se* is not an influential factor (except for AB, size of constraint system). This factor is to be clearly distinguished from the 'integerising' power of a specific cutting plane or surrogate constraint, which was not investigated in this study.

All other terms were highly significant. The multiple coefficient of determination indicates that about 85 per cent of the variation in RIP30C computer time is explained by the above model. These results suggest that the regression equation may be used as a fairly accurate predictor of RIP30C time.

T_h. Analysis of variance indicated that interaction terms AC and BC were both highly significant. Examination of AC cell means indicated that the *increase* in T_h versus the number of constraints became progressively smaller as the problems became less severely constrained. The BC term exhibited the same pattern as it had for T_o. The upward slope of T_h grew steeper for successively lower values of factor C, suggesting the term BC^{-1}. Thus, adding the two non-linear terms AC and BC^{-1}, the following equation was derived by stepwise regression

$$\begin{array}{lllllll} & T_h^{\frac{1}{4}} = 0{\cdot}044 & +0{\cdot}036A & +0{\cdot}164B & -0{\cdot}119C & + & (10)\\ \text{S.E.:} & (0{\cdot}041) & (0{\cdot}012) & (0{\cdot}011) & (0{\cdot}008) & & \\ \text{F(6,173):} & 1064{\cdot}88 & 8{\cdot}14 & 195{\cdot}07 & 193{\cdot}54 & & \\ R^2 = 0{\cdot}974 & & & & & & \\ & & & & 0{\cdot}071AB & -0{\cdot}025AC & -0{\cdot}110BC^{-1}\\ & & & & (0{\cdot}004) & (0{\cdot}003) & (0{\cdot}008)\\ & & & & 249{\cdot}15 & 59{\cdot}91 & 188{\cdot}00 \end{array}$$

All of the initially proposed terms plus the two suggested by analysis of variance were highly significant at the $\alpha = 0{\cdot}01$ level. Practically all of the variation was explained using this model. Residuals were found to exhibit normality with satisfactory homogeneity of variance and absence of curvilinearity.

Z_o and Z_h. The postulated functions for Z_o and Z_h were practically identical. The interaction term BC was again seen to be highly significant in both cases in the analysis of variance. The slope of Z_o and Z_h versus factor C seemed to vary directly with the level of factor B. Adding the term BC and performing stepwise regression provided

$$\begin{array}{lllll} & Z_o = -47{\cdot}12 & +110{\cdot}53B & +52{\cdot}97BC & (11)\\ \text{S.E.:} & (47{\cdot}18) & (5{\cdot}64) & (1{\cdot}45) & \\ \text{F(2,177):} & 2257{\cdot}64 & 385{\cdot}08 & 1326{\cdot}24 & \\ R^2 = 0{\cdot}962 & & & & \end{array}$$

$$\begin{array}{lllll} & Z_h = -47{\cdot}22 & +104{\cdot}77B & +54{\cdot}24BC & (12)\\ \text{S.E.:} & (47{\cdot}08) & (5{\cdot}61) & (1{\cdot}45) & \\ \text{F(2,177):} & 2267{\cdot}91 & 347{\cdot}76 & 1397{\cdot}98 & \\ R^2 = 0{\cdot}962 & & & & \end{array}$$

The number of constraints (factor A) was found to be of no significance in predicting Z_o and Z_h. The main effect for factor C was eliminated by the term BC. Since the term BC had the most highly significant regression coefficient, the results suggest that the underlying interactive nature of factors B and C is indeed a salient basis for predicting Z_o and Z_h. The residuals again supported the validity of the regression analysis exhibiting normality, homogeneity, and linearity.

Z_o and Z_u. To extend the findings of this study to objective functions of any scale, a predictive model was formulated for Z_o relative to Z_u. The fit turned out quite well, and the residuals were well behaved. No significant interaction effects were detected using analysis of variance.

$$\begin{array}{lllll} & Z_o/Z_u = 0{\cdot}224 & +0{\cdot}028B & +0{\cdot}178C & (13)\\ \text{S.E.:} & (0{\cdot}047) & (0{\cdot}004) & (0{\cdot}003) & \\ \text{F(2,177):} & 1602{\cdot}33 & 41{\cdot}68 & 3162{\cdot}98 & \\ R^2 = 0{\cdot}948 & & & & \end{array}$$

It should be pointed out that this linear model has limited validity for levels of factors B and C beyond the scope of this study. The predicted value of Z_o/Z_u will eventually exceed unity which is meaningless by definition of Z_o and Z_u.

Decision-rules

To implement the above findings, the following steps may be followed to determine the preferability of the optimising-versus-heuristic techniques studied for any particular case.

Downloaded from https://academic.oup.com/comjnl/article/16/2/135/372658 by guest on 20 August 2022

1. Scale the actual value of the problem dimensions using the transformations in **Table 4.**
2. Compute estimates of Z_o, Z_h, T_o, T_h, and Z_o/Z_u regardless of the actual scale of Z_u.
3. Compute PCT_z from (2).
4. Let $\Delta z = (1 - PCT_2)\ (Z_o/Z_u)\ Z_u$, where Z_u is the unconstrained value of the objective function of the actual problem.
5. Let $\Delta t = T - T_h$
6. If $\Delta z \geq (\Delta t)K$, use the optimal procedure else use the heuristic (K is cost-per-second of CPU time).

An approximate method might be to use the entries in **Table 5** which is based on the four corner-entries of Table 3. Z' gives the slope for the break even function which emanates from the intersection of the Z_u and K axes

$$Z' = K(\Delta t)/[(1 - PCT_2)(Z_o/Z_u)] \qquad (14)$$

Example:
Consider a 45 variable problem with 30 per cent constraint slackness, with K = 15¢/second and Z_u = \$450. By Table 5, the low value of the unconstrained objective function relative to the cost of computation would indicate the preferability of the heuristic method.
For example, consider the above problem except that Z_u = \$5,000. Also by Table 5 it is clear now that the optimising method is indicated, due to the increased economic value of the decision variables.

For a given problem size and constraint system, optimisation will eventually become preferable for a sufficiently large increase in the objective function coefficients. But for a given distribution of objective function coefficients, the heuristic becomes preferable to the optimiser for a sufficiently difficult problem.

Summary and conclusions

The purpose of this paper was to suggest and illustrate a methodology for the systematic study of performance of integer programming techniques. The vehicle for such an effort has been the comparison of two binary programming codes, on a specific yet important problem. The comparison was performed by varying three factors of problem structure in an experimental fashion. Experimental control was exercised by generating problems randomly, drawing both c_j and a_{ij} coefficients from a horizontal distribution. Such control was exercised to establish a definite benchmark for future comparisons with other problem structures, and other computational algorithms.

The results indicate that the methodology (sample size, factor levels, and control measures) provided a highly sensitive experiment. On the basis of this study, it may be fairly concluded that a benchmark has now been made. Support for this conclusion includes the following considerations:

1. The heuristic results were practically identical to the optimal solutions while requiring about one-tenth the computational effort. Hence, the study was successful in that the example permitted clear illustration of the instances in which a heuristic dominates the optimising technique, and should, therefore, be advocated by the operations researcher as the 'economically' rational choice.
2. The sensitivity of the experiment yielded significant predictive tools for estimating the parameters of the decision rule. These tools add reliability to the general result above (1), and formally express an intuitive understanding of the behaviour of the two codes. The relationships expressed will surely vary for different methods of problem generation, different problem structures, and other integer algorithms. Nevertheless, these relationships can serve as a baseline for comparison with predictive tools derived in future studies.
3. It does not appear that constraint severity (slackness) has been reported as a salient factor of computational behaviour. Since this aspect of problem structure was highly significant in each regression equation, it might be useful for future studies to reflect the relative significance of constraint severity (in addition to problem size) when reporting computational experience.

Limitations and extensions

Since the problem studied was highly specific, several limitations, cautions and extensions for further research should be noted. First, the criteria for choosing a computer code include several qualitative factors, which were not included in this study. The inclusion of these factors should be the long-range goal of those interested in decision rules for selecting computer codes. In the interim, they must be considered as distinct but parallel criteria by the decision maker.

Secondly, the high values obtained for R^2, the multiple coefficient of determination, should not intimidate prospective researchers who wish to improve upon the regression model. Since the phenomena are purely inanimate and deterministic we should be able to explain nearly all of the observed variation. Furthermore, alternative models that seem intuitively reasonable, might yield comparable reliability. In such a case, accuracy of estimates beyond the experimental treatment levels (i.e. power of generalisation) should be the criterion of selecting the more valid tool.

Third, the particular problem studied might be better understood by the inclusion of additional factors. This is especially true for T_o, since the path of enumeration depends on several poorly understood attributes of problem structure. Some of the following factors could be varied in future studies to obtain a better fit for T_o:

1. Matrix density: the proportion of non-zero entries in the constraint system (see Trauth and Woolsey, 1969).
2. Distribution (mean, dispersion, skewness, and kurtosis) of constraint severity.
3. Distribution of the c_j coefficients.
4. Distribution of the a_{ij} coefficients.

The suggested factors in addition to problem dimensions would require a total of 15 factors, each at least three levels to assess quadratic trends. Obviously the experiment would have to be executed as a series of sub-experiments.

The final and perhaps most stringent limitation on the results is the fact that the Senju–Toyoda heuristic is not a general integer or even a general 0–1 heuristic. To extend the relevance of this study, it would be useful if the Senju–Toyoda principle could be modified somehow for the general 0–1 problem. If this could be done, the heuristic could be imbedded in the optimising algorithm to provide a bound near the optimal, thus greatly accelerating implicit enumeration. It would also be useful if this technique could be modified somehow to be a heuristic for the general integer programming problem.

This paper is the first in a series that will hopefully carry out some of the extensions suggested above.

References

Balas, E. (1965). An Additive Algorithm for Solving Linear Programs with 0-1 Variables, *Operations Research*, Vol. 13, No. 4, pp. 517-549.
Geoffrion, A. M. (1968). An Improved Implicit Enumeration Approach for Integer Programming, The Rand Corporation, RM-5644-PR.
Geoffrion, A. M. and Nelson, A. B. (1968). Users Instructions for (0, 1) Integer Linear Programming Code RIP30C, Rand Memorandum RM-5627-PR.

Downloaded from https://academic.oup.com/comjnl/article/16/2/135/372858 by guest on 20 August 2022

GLOVER, F. (1965). A Multiphase-Dual Algorithm for the Zero-One Integer Programming Problem, *Operations Research*, Vol. 13, No. 6, pp. 879-919.
SENJU, S. and TOYODA, Y. (1968). An Approach to Linear Programming with 0-1 Variables, *Management Science*, Vol. 15, No. 4, pp. B-196 to B-208.
TRAUTH, C. A. and WOOLSEY, R. E. (1969). Integer Linear Programming: A Study in Computational Effectiveness, *Management Science*, Vol. 15, No. 9, pp. 481-494.

Eric Mutch Memorial Prize

In earlier notices concerning the above prize it was stated that the result would be announced in the May 1973 issue of the *Journal*. In the event, the early copy date of this issue has not allowed time for the adjudicators to come to a decision. Hence, the result will now be published in August.

Errata

In the Volume 15, Number 4 issue of this *Journal* there were a number of errors.

On page 307, in the letter of D. R. Gayler, line 4, the name of Mr. Walwyn was spelled incorrectly as Walwan.

On page 308, first footnote, Mr. S. W. Taylor was inadvertently referred to as Mr. J. W. Taylor.

In the paper 'Numerical solution of the differential equations of a synchronous generator' by I. R. Smith and L. A. Snider there were two errors:

(*a*) in equation 4 of page 351 the final terms in the expressions common to the third row and the fourth and fifth columns should read $i'_D M_f^* \cos \theta_b$;

(*b*) in Fig. 5(*b*) of page 354 the computed results with the lower of the two experimental recordings should be ● rather than ○.

Downloaded from https://academic.oup.com/comjnl/article/16/2/135/372658 by guest on 20 August 2022