An Entropy-based Solver for Multidimensional Nonlinear Knapsack Problems

An Entropy-based Solver for Multidimensional Nonlinear Knapsack Problems

Yuji Nakagawa[a], Ross J.W. James[b], César Rego[c*], Chanaka Edirisinghe[d]

a Faculty of Informatics, Kansai University, Ryozenjicho, Takatsuki-shi, 569–1095 Japan. nakagawa@res.kutc.kansai-u.ac.jp

b Department of Management, University of Canterbury, Private Bag 4800, Christchurch, New Zealand. ross.james@canterbury.ac.nz

c School of Business Administration, University of Mississippi, University, MS 38677, USA. crego@bus.olemiss.edu

d College of Business Administration, University of Tennessee, Knoxville, TN 37996, USA. chanaka@utk.edu

Latest Revision: June, 2009

Abstract: This paper develops an implicit enumeration method for solving difficult linear and nonlinear multidimensional knapsack problems where branching is accomplished based on the sub-problem complexity. Using the concept of entropy in information theory, we develop a (sub)problem-difficulty metric that is used to devise decision rules for problem partitioning within implicit enumeration. We demonstrate the effectiveness of this concept by specializing the scheme for the improved surrogate constraint (ISC) method for solving discrete nonlinear knapsack problems. The proposed entropy-based partitioning and enumeration approach enables the resulting hybrid ISC scheme to solve problems of larger size exactly and more efficiently. The effectiveness of the method is demonstrated using computational experiments on a diverse set of test problems that are generally recognized to be difficult problem instances, including both nonconvex and convex objective functions.

Keywords: Multi-dimensional nonlinear knapsack, multi-constraint separable discrete optimization, combinatorial optimization, problem difficulty estimation, entropy

* Corresponding author

1. Introduction

This paper proposes a new approach for solving separable discrete mathematical programs, focusing in particular on nonlinear knapsack problems. The approach is novel in that it uses the information theoretic concept of entropy to estimate the time required for solving a potential sub-problem, and this information is used to both partition the problem and determine how the sub-problem will be solved. Information theory has previously been used to establish the complexity of a sorting algorithm (Takaoka, 1998); however, to our knowledge, this is the first attempt to use information theory for estimating the difficulty of a complex combinatorial optimization problem within an optimization solver.

Virtually all modern software packages for solving mixed integer linear programming problems (MILPs) use a variant of the branch-and-cut approach. Despite vast improvements in its implementation over the past two decades and recent quantum leaps in computing power, many MILPs arising in practice remain difficult to solve by branch-and-cut. The difficulty stems mainly from limitations in memory and processing power (Ralph, 2006). The entropy-based method presented here, however, succeeded in reducing the memory required by the Improved Surrogate Constraints (ISC) method (Nakagawa 2003) enabling large problems to be solved very efficiently.

The proposed approach builds upon the ISC algorithm and is herein termed the Improved Surrogate Constraint method with Problem Partitioning (ISCpp). ISCpp grew out of the authors' earlier work on surrogate constraint algorithms for nonlinear knapsack problems (Nakagawa and Iwasaki, 1999 and Nakagawa, 2003). The method partitions the problem into easier sub-problems using a problem difficulty measure based on the concept of entropy. We show that our specific entropy metric is strongly correlated to the expected CPU time required to solve a sub-problem, thus providing an effective measure of solution difficulty.

The remainder of the paper is organized as follows. In Section 2 we provide a useful background for understanding the context and relevance of the study. Section 3 provides the foundation for the framework underlying the proposed algorithm and Section 4 presents the problem formulation together with the improved surrogate constraint method (ISC) used to solve the sub-problems. Section 5 introduces the basic methodology aimed at reducing space and time complexities to solve large MNK problems. The proposed entropy-based approach for problem partitioning is introduced in Section 6, followed by the ISC version that incorporates the ISCpp partitioning technique in Section 7. Section 8 presents the methodology to determine an appropriate probability function used to estimate problem difficulty. Computational experiments on several different kinds of knapsack problems are reported in Section 9, along with comparisons between the performance of ISCpp and that of other solvers. Finally, Section 10 summarizes the results and makes some concluding remarks.

2. Background

The Multi-dimensional Nonlinear Knapsack (MNK) problem subsumes all other multi-dimensional knapsack problems as a special case, and therefore has a wide range of applications but is also the most difficult type of knapsack problem to solve.

Many papers have discussed various instances and features of the Nonlinear Knapsack Problem. Cooper (1981) presents a survey which classifies and discusses algorithms for solving nonlinear pure integer programming problems. Marsten and Morin (1978) combine dynamic programming and branch-and-bound techniques to produce a hybrid algorithm for solving the problem with multiple constraints. The Multiple-choice Knapsack Problem (MCKP), presented by Nauss (1978), is a linearization of the single-constraint nonlinear knapsack problem. Sinha and Zoltner

(1979), Armstrong, Kung, Sinha and Zoltner (1983), and Dyer, Kayal and Walker (1984] all present different algorithms for solving the multiple-choice knapsack problem. Ibaraki and Katoh (1988) present a detailed discussion about the Resource Allocation Problem, which is the minimization of a nonlinear function over one constraint and bounded integer variables. The separable version of the Resource Allocation Problem is a special case of the nonlinear knapsack problem. Bretthauer and Shetty (1995) develop a branch-and-bound algorithm to solve separable problems involving nonlinear resource allocation. Hochbaum (1995) discussed the computational complexity of convex quadratic knapsack problems subject to a single linear constraint.

In recent years, much progress has been made in developing exact methods for special cases of the 0-1 (linear) knapsack problems and in developing near-optimal (heuristic) methods for other knapsack problems. Martello, Pisinger and Toth (1999) present a combination of two new algorithms, which is shown to outperform all previous methods, for solving a single-constraint 0-1 knapsack problem exactly. Bertsimas and Demir (2002) present an approximate dynamic programming approach for the multidimensional knapsack problem. Martello and Toth (2003) present an exact algorithm for the two-constraint 0-1 knapsack problem. Many different kinds of heuristic approaches have been proposed in the literature across a broad range of academic journals (for example see Coit and Smith (1996a, 1996b), Ng and Sancho (2001), Hsieh (2002) and Liang and Smith (2004)).

The knapsack problem has many applications over domains that include R&D (Petersen, 1967), capacity planning in computer networks (Gerla and Kleinrock, 1977), stratified sampling (Hughes and Rao, 1979), sales resource allocation (Zoltners and Sinha, 1980), catalog planning (Armstrong, Sinha and Zoltners, 1982), production planning (Ziegler, 1982), capital budgeting (Mathur, Salkin and Morito, 1983), capacity

planning in manufacturing networks (Bitran and Tirupati, 1989a, 1989b), layout problem in the fashion industry (Degraeve and Vandebroek, 1998), allocating funds to highway safety improvements (Melachrinoudis and Kozanidis, 2002), multicomponent and multiproduct periodic-review assemble-to-order system (Akcay and Xu 2004), pricing and allocation of unique one-time digital products (Bapna, Goes, and Gupta, 2005) and the production-distribution system design (Elhedhli and Goffin 2005). The Index fund Optimization problem is modeled as a nonlinear knapsack problem in Nakagawa, Isada and James (2005), as are the Index-plus-alpha fund optimization problems under the restriction of nonlinear execution cost (Nakagawa, James, Rego, Glover 2009). Since MNK class of problems subsumes all of the above knapsack problems, an effective solution technique MNK would be useful in many different applications.

In the context of global optimization, the MNK problem is a non-smooth, discontinuous, nonlinear, non-convex constrained model with integer variables. When the objective function and all of the constraints (except for integer restrictions) are differentiable nonlinear functions of the decision variables, the problem is called an integer nonlinear (smooth) optimization program. A number of solvers exist for determining global optimal solutions of smooth nonlinear convex or non-convex problems, see e.g. Bonmin (Bonami and Lee 2006, Bonami et al., 2005), Baron (Sahinidis and Tawarmalani, 2005, Tawarmalani and Sahinidis, 2004), and the commercial software Premium Solver Platform (2005). When the objective or any of the constraints are non-differentiable, the resulting problem is a non-smooth optimization model, which is also the most difficult class of optimization problems to solve.

The approach proposed in this paper is a new method designed to find the exact optimal solution to these difficult non-smooth optimization problems. Our new technique not only affords the ability to solve non-smooth problems, which were

unable to be solved previously, but also improves on prior approaches for solving large smooth problems.

3. The Surrogate Constraint Framework and Entropy

The surrogate constraint method that provides a primary component of our approach was first introduced to the field of mathematical programming by Glover (1965) for solving 0-1 integer programs. In its most common form, the approach is applied to solving multi-constrained optimization problems by solving a series of single-constraint (knapsack) problems. These problems arise by replacing the original problem constraints with a single surrogate constraint, generated by a weighted combination of the original constraints. The single constraint nonlinear knapsack problem can be solved efficiently by using the modular approach of Nakagawa and Iwasaki (1999). Optimal weights, called surrogate multipliers, can be calculated for the constraints using the algorithm proposed by Dyer (1980) or the cut-off polyhedron (COP) method proposed by Nakagawa et al. (1981, 1984).

Surrogate constraint methods, while yielding stronger relaxations than Lagrangean methods, can encounter a duality gap, which means the optimal solution to a surrogate constraint relaxation may fail to produce an optimal solution to the original problem by virtue of failing to satisfy some of the original problem constraints. To overcome this difficulty, Nakagawa (2003) proposed an improved surrogate constraint method (ISC) for the solution of nonlinear separable discrete optimization problems, often permitting exceedingly large problems to be solved exactly.

Notably, some large-scale problems are more easily solved by the ISC than some small-scale problems, indicating that the problem size, measured in terms of the number of variables or constraints is not the only factor that makes these problems difficult. The percent gap closure (PGC) value of Karwan, Rardin and Sarin (1987) for

measuring problem complexity suggests that problems with a large surrogate gap (i.e. a small PGC value) are more difficult to solve. Although this hypothesis often proves true, there are many instances where on the contrary problems having a small PGC value are easily solved by the ISC method.

The ISCpp method proposed in this paper goes beyond the ISC method by using an information theoretic entropy metric to assess the difficulty of the multidimensional 0-1 knapsack problem and then generalizing this to non 0-1 (multi-valued) integer knapsack problems. Each variable of the problem is binary, taking a value of 0 or 1 according to whether the associated item is chosen for inclusion in the knapsack. When two items have very similar properties, both items may have the same probability of belonging in an optimal solution. On the other hand, if the properties of one item are quite different from another item, then one may be somewhat more likely than the other to be included in an optimal solution. A difficult problem is typically one that includes many variables that have similar properties and hence that are difficult to differentiate as possible members of an optimal solution.

The ISC uses a technique similar to that of branch-and-bound with breadth-first-search, and as a result generally consumes significant computer memory to solve a difficult problem instance. To overcome memory limitations, we propose a Problem Partition technique that utilizes a measure of problem difficulty in order to partition the problem, with each sub-problem being solved using the ISC method[1].

4. Problem Formulation and Fundamental Relationships

4.1 Surrogate constraints relaxation for MNK

Multi-dimensional nonlinear knapsack (MNK) problems can be stated as:

[1] A trial version of the software implementing the pure ISC method, with a restriction on the problem size, can be obtained from http://www.res.kutc.kansai-u.ac.jp/~nakagawa/orlib/code/hope.

$$\text{P}: \max \quad f(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i)$$

$$\text{s.t.} \quad g_j(\mathbf{x}) = \sum_{i=1}^{n} g_{ji}(x_i) \le b_j \quad (j \in M)$$

$$x_i \in K_i \quad (i \in N)$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is the decision variable vector, $M = \{1, 2, \ldots, m\}$ is a set of constraint indices, $N = \{1, 2, \ldots, n\}$ is the set of variable indices and $K_i = \{0, 1, 2, \ldots, k_i\}$ identifies the alternative item set for each variable x_i. Without loss of generality, we assume that:

$$f_i(x_i) \ge 0 \qquad \text{for} \quad x_i \in K_i, \ i \in N,$$
$$g_{ji}(x_i) \ge 0 \qquad \text{for} \quad x_i \in K_i, \ i \in N, \ j \in M.$$

When $\mathbf{x}$ in problem P consists of binary variables and the functions $f_i(x_i)$ and $g_{ji}(x_i)$ are monotone non-decreasing the problem is a 0-1 (linear) knapsack problem. If $\mathbf{x}$ is binary and one of the functions $f_i(x_i)$ and $g_{ji}(x_i)$ is non-decreasing and the other is non-increasing, P can still be rewritten as a 0-1 linear programming problem by using negative coefficients and modifying the appropriate constraint's right-hand-side. Note that any separable discrete optimization problem can be transformed into problem P.

Surrogate constraint relaxation provides an effective approach to compute an upper bound f^{UB} on the optimal value $f(\mathbf{x})$ and is the corner stone to our method for estimating the difficulty of the MNK.

The surrogate problem $\text{P}^{\text{S}}(\mathbf{u})$, constituting a relaxation of the original problem P, can be written as

$$\text{P}^{\text{S}}(\mathbf{u}): \ \max \quad f(\mathbf{x})$$

$$\text{s.t.} \quad \sum_{j \in M} u_j g_j(\mathbf{x}) \le \sum_{j \in M} u_j b_j$$

$$x_i \in K_i \quad (i \in N)$$

where $\mathbf{u} = (u_1, u_2, \ldots, u_m)$ is a vector of non-negative weights associated with the original problem constraints.

The surrogate dual problem is defined by

$$\mathrm{P}^{\mathrm{SD}}: \min\left\{v^{\mathrm{Opt}}[\mathrm{P}^{\mathrm{S}}(\mathbf{u})] \ : \mathbf{u} \in \mathbf{U}\right\}$$

where $v^{\mathrm{Opt}}[\bullet]$ is the optimal objective function value of problem $\bullet$,

$$\mathbf{U} = \left\{\mathbf{u} \in R^m : \sum_{j=1}^{m} u_j = 1, \ \mathbf{u} \geq \mathbf{0}\right\}.$$

If $\mathbf{u}^* = (u_1^*, u_2^*, \ldots, u_m^*)$ denotes the optimal surrogate multiplier, then the upper bound, f^{UB} of P is calculated by solving the surrogate problem $\mathrm{P}^{\mathrm{S}}(\mathbf{u}^*)$. An optimal multiplier $\mathbf{u}^*$ for the surrogate dual P^{SD} can be obtained by using the algorithms proposed by Dyer (1980), Nakagawa and Miyazaki (1981), or Nakagawa, Hikita and Kamada (1984).

When the optimal solution of the surrogate problem $\mathrm{P}^{\mathrm{S}}(\mathbf{u}^*)$ is infeasible for P, it is said that there exists a surrogate duality gap. In this case, the value $f(\mathbf{x}^{\mathrm{SD}})$ is an upper bound on the optimal objective function value of P.

4.2 The improved surrogate constraints (ISC) method

Nakagawa (2003) proposed an improved surrogate constraints method (ISC) to close the surrogate duality gap by considering the following "target problem":

$$\begin{aligned}
&\mathrm{P}^{\mathrm{T}}(f^{\mathrm{T}}, \ \mathbf{u}^*): \text{ Enumerate all solutions } \mathbf{x} \text{ hitting} \\
&\text{the target: } f(\mathbf{x}) \geq f^{\mathrm{T}} \\
&\text{s.t. } \sum_{j \in M} u_j^* g_j(\mathbf{x}) \leq \sum_{j \in M} u_j^* b_j \\
&\quad x_i \in K_i \quad (i \in N)
\end{aligned}$$

where $\mathbf{u}^*$ is an optimal multiplier vector of a surrogate dual problem PSD associated with the original multi-constrained problem P. The solutions hitting the target are called target solutions. The problem $\mathrm{P}^{\mathrm{T}}(f^{\mathrm{T}}, \ \mathbf{u}^*)$ is used to generate all target solutions,

which are the feasible solutions that have an objective value greater or equal to a given target value f^T.

Property 1: If $f^{\mathrm{T}} \le v^{\mathrm{Opt}}[\mathrm{P}]$, then all exact optimal solutions of P are target solutions to the target problem $\mathrm{P}^{\mathrm{T}}(f^{\mathrm{T}},\ \mathbf{u}^*)$.

As a result of this property, if $f^{\mathrm{T}} \le v^{\mathrm{Opt}}[\mathrm{P}]$ and if we can enumerate every target solution of problem $\mathrm{P}^{\mathrm{T}}(f^{\mathrm{T}}, \mathbf{u}^*)$ then we have all the exact optimal solutions of P. If in addition we have a heuristic method that can produce near-optimal solutions to the original problem P, then we can use the objective value of this near-optimal solution as the target value f^{T} of $\mathrm{P}^{\mathrm{T}}(f^{\mathrm{T}}, \mathbf{u}^*)$. On the other hand, when such a near-optimal solution is not available, the target values f^T can be chosen from an interval $f(\mathbf{x}^{\mathrm{Near}}) \le f^T \le f(\mathbf{x}^{\mathrm{SD}})$, where $\mathbf{x}^{\mathrm{Near}}$ is an estimate of the optimal solution of P. Near optimal values can be obtained by a heuristic approach.

5. Basic Methodology

5.1 The modular approach

The target problems can be solved using a modular approach (MA) described in Nakagawa (1990) and Nakagawa and Iwasaki (1999). The MA consists of two fundamental components: (1) a fathoming test aimed at reducing the decision space of the variables of the current problem; and (2) an aggregation process that combines two variables into one new variable to reduce the number of variables in the current problem. The aggregation is repeated until no further reduction is possible in either step (1) or (2). The process terminates when the number of variables in the problem

becomes smaller than a given number. The problem is then solved using the enumeration method outlined in James and Nakagawa (2005).

MA can use three fathoming tests based respectively on feasibility, bounding and dominance. Each of these tests can be applied to the current problem for reducing the size of the corresponding variable decision space. To solve a target problem exactly, the feasibility test and the bounding test are used for fathoming. Dominance tests are not used for a target problem with a single surrogate constraint as their use could eliminate a branch that contains the exact optimal solution. For the aggregation process the algorithm implements the following policies according to the type of the problem (Nakagawa and Iwasaki, 1999):

Homogeneous problems (with the same number of items) – Aggregate the next two variables in order of the variable index.

Heterogeneous problems (with different numbers of items) – Aggregate one variable having the minimum number of items, the other the maximum number of items.

In order to calculate the upper bound of P, the following Integer Dominance and the Decreasing Gain Ratio (DGR) dominance in the optimal surrogate problem $P^S(\mathbf{u}^*)$ are used, where $\mathbf{u}^*$ is the optimal multiplier vector:

Integer dominance – A partial solution $x_i=k$ ($k \in K_i$) is dominated and has been fathomed, if there exists $k' \in K_i$ such that

$$f_i(k') \geq f_i(k) \text{ and } g_i^*(k') \leq g_i^*(k),$$

where

$$g_i^*(k) = \sum_{j \in M} u_j^* g_{ji}(k).$$

DGR dominance – A partial solution $x_i = k$ ($k \in K_i$) is dominated, if there exists $k', k'' \in K_i$ such that

$$\frac{f_i(k) - f_i(k')}{g_i^*(k) - g_i^*(k')} \leq \frac{f_i(k'') - f_i(k)}{g_i^*(k'') - g_i^*(k)},$$

$$f_i(k') \le f_i(k) \le f_i(k''),$$
$$g_i^*(k') \le g_i^*(k) \le g_i^*(k'').$$

The DGR dominance for multiple-choice knapsack problems is equivalent to LP dominance as defined in Sinha and Zoltners (1979). After applying Integer Dominance and DGR Dominance to the problem $P^S(\mathbf{u}^*)$, we obtain a Single-Constraint Nonlinear Knapsack problem of DGR type, as follows:

$$\begin{aligned} P^B(b^B): \quad \max \quad & f^B(\mathbf{x}) = \sum_{i=1}^{n^A} f_i^B(x_i) \\ s.t. \quad & g^B(\mathbf{x}) = \sum_{i=1}^{n^A} g_i^B(x_i) \le b^B, \\ & x_i \in \{1, 2, \ldots, k_i^B\} \qquad \text{for} \quad i = 1, \ldots, n^A, \end{aligned}$$

where

$$\begin{aligned} & f_i^B(k) < f_i^B(k+1) \quad \text{for} \quad k \in \{1, 2, \ldots, k_i^B - 1\}, \ i = 1, \ldots, n^A, \\ & g_i^B(k) < g_i^B(k+1) \quad \text{for} \quad k \in \{1, 2, \ldots, k_i^B - 1\}, \ i = 1, \ldots, n^A, \\ & w_i(k) > w_i(k+1) \quad \text{for} \quad k \in \{2, 3, \ldots, k_i^B - 1\}, \ i = 1, \ldots, n^A, \\ & w_i(k) = \frac{f_i^B(k) - f_i^B(k-1)}{g_i^B(k) - g_i^B(k-1)}. \end{aligned}$$

The well known greedy algorithm of Fox (1966) generates a greedy solution $\mathbf{x}^G$ to the problem $[P^B(b^B)]$ such that:

$$\min_{i=1,2,\ldots,n^A} \{w_i(x_i^G)\} > w_{i*}(x_{i*}^G + 1),$$
$$0 \le b^B - \sum_{i=1}^{n} g_i^B(x_i^G) < g_{i*}^B(x_{i*}^G + 1),$$

where $i*$ is defined by

$$w_{i*}(x_{i*}^G + 1) = \max_{i=1,2,\ldots,n^A} \left\{ w_i(x_i^G + 1) \right\}.$$

The greedy solution obtained may update the incumbent solution. The upper bound of $[P^B(b^B)]$ can be obtained by using the greedy solution $\mathbf{x}^G$.

$$f^R = \sum_{i=1}^{n^A} f_i^B(x_i^G) + w_{i*}(x_{i*}^G + 1)(b^B - \sum_{i=1}^{n^A} g_i^B(x_i^G))$$

Sinha and Zoltner (1979) give a stricter upper bound. Identify p satisfying

$$g_{i^*}^{\mathrm{A}}(p) \le h \le g_{i^*}^{\mathrm{A}}(p+1),$$

where $h = g_{i^*}^{\mathrm{B}}(x_{i^*}^{\mathrm{G}}) + b^{\mathrm{B}} - \sum_{i=1}^{n^{\mathrm{A}}} g_i^{\mathrm{B}}(x_i^{\mathrm{B}})$,

and determine i^{Prv} and i^{Nxt} such that

$$w_{i^{\mathrm{Prv}}}(x_{i^{\mathrm{Prv}}}^{\mathrm{G}}) = \max_{\substack{i=1,2,\ldots,n^{\mathrm{A}} \\ i \ne i^*}} \left\{ w_i(x_i^{\mathrm{G}}) \right\}$$

and

$$w_{i^{\mathrm{Nxt}}}(x_{i^{\mathrm{Nxt}}}^{\mathrm{G}} + 1) = \max_{\substack{i=1,2,\ldots,n^{\mathrm{A}} \\ i \ne i^*}} \left\{ w_i(x_i^{\mathrm{G}} + 1) \right\}.$$

The Sinha-Zoltoner upper bound is

$$f^{\mathrm{UB}} = \max\left\{ U^1,\ U^2 \right\}$$

where

$$U^1 = f^{\mathrm{R}} - (w_{i^*}(x_{i^*}^{\mathrm{G}} + 1) - w_{i^{\mathrm{Nxt}}}(x_{i^{\mathrm{Nxt}}}^{\mathrm{G}} + 1))(h - g_{i^*}^{\mathrm{A}}(p)),$$

$$U^2 = f^{\mathrm{R}} - (w_{i^{\mathrm{Prv}}}(x_{i^{\mathrm{Prv}}}^{\mathrm{G}}) - w_{i^*}(x_{i^*}^{\mathrm{G}} + 1))(g_{i^*}^{\mathrm{A}}(p+1) - h).$$

The Sinha-Zoltner bound is used in our algorithm for calculating the entropy difficulty.

5.2 Prior results and motivation

The efficiency of the ISC method has been demonstrated in solving multi-constraint nonlinear knapsack problems. In this setting, problems with 3 constraints, 1000 variables, and 20 alternative items for each variable and problems with 8 constraints, 500 variables, and 50 alternative items could be solved to optimality in a reasonable amount of time. Noticeably, these instances are exceedingly large in comparison to other similar types of problems in the literature. For example, the classical testbed for reliability design problem, formulated as a nonlinear separable discrete optimization problem, consists of 33 instances containing 2 constraints, 14 variables and 216 or 1296 items for each variable. Exact optimal solutions for these

instances were unknown for many years. As exact methods were unable to solve these problems, heuristic approaches have often been the methodology of choice to seek high quality (but not necessarily the optimal) solutions for these instances. The latest research on heuristic methods for redundancy allocation problems uses an Ant Colony Optimization approach by Liang and Smith (2004) that succeeded in finding optimal solutions for 24 of these problems. By contrast, the ISC method yields exact optimal solutions for all 33 problem instances within one second for the entire problem set (Ohnishi et al. 2007).

6. ISC with Problem Partition and Entropy: ISCpp

The partitioning technique that underlies the ISCpp method operates as follows:

1) The entropy value of the variables determines which two variables are to be combined in the aggregation process of the ISC method;
2) The sum of the variable entropy is used to determine if the problem needs to be partitioned further before being solved with the ISC method.

To apply our hypothesis that the difficulty of solving a problem depends on how difficult it is to determine which variables are hard to optimize in the problem, we define the variable difficulty of x_i as the entropy or uncertainty that the value $k \in K_i$ of a variable i will be part of the optimal solution. The entropy can be measured by the probability $p_i(k)$ ($k \in K_i$) of an item appearing in the optimal solution. The probability $p_i(k)$ is related to the associated standardized difference of the upper bounds, $\delta_i(k)$, which is defined as:

$$\delta_i(k) = \frac{f^{\mathrm{UB}} - v^{\mathrm{UB}}[\mathrm{P} : x_i = k]}{f^{\mathrm{UB}} - f^{\mathrm{LB}}} \qquad (i \in N, k \in K_i)$$

where $v^{UB}[P:\bullet]$ is an upper bound of P with a restriction $\bullet$, f^{UB} is an upper bound of problem P, and f^{LB} is a lower bound of P, which is obtained by a near-optimal method. Note that we have $0 \le \delta_i(k) \le 1$, since if $v^{UB}[P: x_i = k] < f^{LB}$, then the branch where $x_i = k$ has been fathomed.

However in order to convert $\delta_i(k)$ into the probability of the variable i being in the final solution $(p_i(k))$ we need to develop a distribution that represents the likelihood that if a variable has a given standardized difference how likely it is to be in the final solution. In order to develop this distribution we examine a series of $\delta_i(k)$ values for similar problems with known optimal solutions. The $\delta_i(k)$ values are then grouped into sets of decreasingly more "difficult" variables and the probability function $p_i(k)$ is created to approximate this distribution. This process is demonstrated in Section 8.

We conjecture that the difficulty of a variable is correlated with its level of uncertainty and may be estimated using the concept of entropy from information theory and statistical mechanics (Shannon, 1948). Under this assumption, the entropy of a variable i taking the value $k \in K_i$ as part of the optimal solution be calculated as:

$$h_i = -\sum_{k \in K_i} p_i(k) \log_2 p_i(k).$$

Variable entropy is maximized when the item probability $p_i(k)$ is near $1/k_i$ and minimized when every $p_i(k)$ is near 0.0 or 1.0. It is difficult to determine an optimal value of a variable when the variable entropy is high. Conversely, it can be said that an optimal value of the item is easily determined when the variable entropy is small. It is difficult to solve problems that contain too many variables with high entropy. By contrast, problems containing a large number of variables with low levels of variable entropy should be easy to solve. Hence, the sum of the entropy values over all variables provides a meaningful metric to estimate the problem difficulty, that is:

$$H = \sum_{i \in N} h_i .$$

7. The ISCpp Algorithm

Our proposed ISCpp algorithm modifies the modular approach used in the original ISC method by introducing the variable entropy measure as the criterion for selecting the two variables to be combined into one. A small entropy value generally signals that the variable's optimal item can be determined easily. We use an aggregation policy of choosing one variable that has the minimum entropy and one variable that has maximum entropy. After aggregating a sufficient number of variables, easy variables disappear from the problem and difficult variables with high entropy remain.

When choosing a bounding test for fathoming, there is a trade-off between the computational time of obtaining a better upper bound and the saving of computational time through the use of a better upper bound. At this stage, an enumeration technique with a relatively light loaded bounding test (i.e. the bounds are computed quickly and less accurately) is helpful for solving the problems having only difficult variables, since high loaded bounding tests are not accurate enough to calculate bounds for problems containing just difficult variables. An enumeration technique (James and Nakagawa, 2005) has been developed for this purpose. The method employs a so-called "move table technique" to determine the next variable with a better objective value that satisfies the constraint requirement for each variable and constraint. The present solver uses the ISC method with this enumeration technique.

The ISCpp algorithm can be sketched as follows. The method operates on a candidate list, L, of subproblems, initialized with the original problem P. The method iterates by selecting one subproblem P^S from the list L, applying the fathoming test to P^S and calculating the entropy of the problem in order to determine its difficulty. If the problem difficulty falls below a given threshold σ, the problem P^S is judged easy

enough to be solved by the ISC method. Otherwise, a variable with maximum difficulty is selected from the variables in P^S. The subproblem P^S is then partitioned into several problems by fixing the variables selected to be assigned to specific items and the associated new subproblems are added to the candidate list L. This procedure is repeated until the list L becomes empty. The ISCpp selects the variable with the highest entropy to be assigned to specific items. This policy is like the branch selection process in a LP-based branch-and-bound used in today's MIP solvers (Achterberg, Koch and Martin, 2005). However the entropy measure is based on the upper bounds of the problem as opposed to using a score based on the change in the objective function of the LP relaxations of the child subproblems and the LP relaxation of the parent sub-problem. In the branch-and-bound the variable having the best score is usually selected for the next branch.

The fathoming test comprises two independent tests for narrowing the decision space (i.e. item space) of the subproblem P^S:

(1) Feasibility test (FT) – If the subproblem P^S does not include any feasible solutions, then the problem has been fathomed.

(2) Bounding test (BT) – If an upper bound of the subproblem P^S is less than the objective function value of the current solution, then the problem has been fathomed.

The general procedure for the ISCpp is described in Figure 3, making use of the following functions:

- ProblemExtraction (L, PE): returns one problem, P^S, out of the problem candidate list L based on a specified policy PE.
- FathomingTest (P^S): applies the FT and BT fathoming tests to the problem P^S and returns the reduced problem P^S.

- ProblemDifficultyEstimate (P^S, PDE): returns an estimate of the problem difficulty, δ, of the problem P^S according to the criterion (or metric) specified by PDE.
- FixedVariableSelection (P^S, FVS): selects one variable x_i from the subproblem P^S based on the policy defined by FVS, and returns the variable index, i.
- ISC (P^S): execute the specialized surrogate constraint method of Nakagawa (2003) and return an exact optimal solution, $\mathbf{x}$ to the problem P^S.

```
Procedure ISC_PP-Method (L, x^Exact, f^Exact, σ)
   While (L is not empty) do
      P^S ← ProblemExtraction (L, PE)
      P^S ← FathomingTest (P^S)
      δ ← ProblemDifficultyEstimate (P^S, PDE)
      If (δ < σ)
         x ← ISC (P^S)
         If (f(x) > f^Exact)
            f^Exact ← f(x);
            x^Exact ← x;
         Endif
      Else
         i ← FixedVariableSelection (P^S, FVS)
         Partition P^S into several subproblems by setting the value of
            variable x_i to each of the allowed values;
         Add the new subproblems obtained to the candidate list L;
      Endif
   Endwhile
End.
```

Figure 3. The General procedure for the ISCpp method

8 Creating the Item Probability Function

The development of the item probability function to estimate problem difficulty is critical to the effectiveness of the partitioning method and so to the efficiency of our ISCpp algorithm. To assess accuracy of the proposed estimation technique for the

problem difficulty, a representative set of 30 classical multi-dimensional 0-1 knapsack benchmark problems presented in Chu and Beasley (1998)[2] was analyzed. This testbed is characterized by problems generated using correlated random numbers, yielding instances that are significantly more difficult to solve than problems generated by independent random numbers. There are 30 problem instances for each size problem. The tightness of the constraints has been changed for every 10 problems so that when the constraint becomes loose, an optimal solution is likely to contain more items. A constraint tightness of 0.25, 0.5, and 0.75 is used to generate problems 00~09, 10~19, and 20~29, respectively. The problems with 5 constraints and 250 binary variables are used to create a function that estimates the variable uncertainty.

Table 1 shows the proportion of error (a measure of uncertainty) that result when the $\delta_i(k)$ ratios, as defined in Section 3, for $k \in \{0,1\}$, are grouped into variables of similar difficulty. To illustrate the process, consider the first proportion of 0.5469 for the problems numbers 00~09 and a $\delta_i(k)$ of 0.00~0.01. There were 64 variables whose values fell into this category. For 35 of these 64 variables the smallest upper bound was generated when the variable is set to its known optimal value, i.e. for these variables $\upsilon^{\mathrm{UB}}\left[\mathrm{P} : x_i \neq k^*\right] \geq \upsilon^{\mathrm{UB}}\left[\mathrm{P} : x_i = k^*\right]$. For the remaining 29 variables the known optimal value produced a larger upper bound i.e. for these variables $\upsilon^{\mathrm{UB}}\left[\mathrm{P} : x_i \neq k^*\right] \leq \upsilon^{\mathrm{UB}}\left[\mathrm{P} : x_i = k^*\right]$. The proportion comes from the fraction 35/64=0.5469. In each column a total of 2500 variables are classified from the 10 sample problem instances.

[2] These test instances are currently available at http://people.brunel.ac.uk/~mastjjb/jeb/info.html

	Proportion of variables in the range $\delta_i(k)$ where the largest upper bound is generated with a non-optimal item			Approximated error function
	Problem Number (Size 5*250)			
$\delta_i(k)$	00~09	10~19	20~29	$\phi_i(k)$
0.00~	0.5469	0.5517	0.4098	0.4595
0.01~	0.2800	0.2937	0.3085	0.2146
0.10~	0.1161	0.1008	0.1221	0.0921
0.20~	0.0421	0.0382	0.0490	0.0396
0.30~	0.0265	0.0167	0.0187	0.0170
0.40~	0.0074	0.0000	0.0000	0.0073
0.50~	0.0000	0.0000	0.0000	0.0031
0.60~	0.0000	0.0000	0.0000	0.0013
0.70~	0.0000	0.0000	0.0000	0.0006
0.80~	0.0000	0.0000	0.0000	0.0002
0.9~1.0	0.0000	0.0000	0.0000	0.0001

Table 1. The relationship between the optimal probability and the associated upper bound ratio

The upper bound ratio has a strong correlation with the probability of the value (either 0 or 1) that each variable can take. The following approximate equation of the error is used instead of the actual data as shown in the right hand column of Table 1:

$$\phi_i(k) = 2^{-12.2*\delta_i(k)-1.0}.$$

It should be noted that $\delta_i(k) = 0.0$ produces $\phi_i(k) = 0.5$.

The relationship between the optimal probability $p_i(k)$ and the upper bound ratio $\delta_i(k)$ is made via the function $\phi_i(k)$. If we define $\theta_i(k) = \dfrac{\phi_i(k)}{1-\phi_i(k)}$, the item probability, i.e. the probability that the item $k \in K_i$ is part of the optimal solution, is then assumed to be:

$$p_i(k) = \frac{\theta_i(k)}{\sum_{s \in K_i} \theta_i(s)} \quad (k \in K_i, i \in N).$$

In the case of 0-1 problems, we have $p_i(0) = 1 - \phi_i(1)$ and $p_i(1) = \phi_i(1)$ for $v^{\mathrm{UB}}[\mathrm{P} : x_i = 0] \geq v^{\mathrm{UB}}[\mathrm{P} : x_i = 1]$, since we have $f^{\mathrm{UB}} = v^{\mathrm{UB}}[\mathrm{P} : x_i = 0]$ and $\phi_i(0) = 0.5$ for most variables. Our concern is to estimate the problem difficulty for these 30 test problems with 5 constraints and 250 variables. A personal computer (Pentium IV with a 3.2GHz processor and 2GB of memory) was used for these experiments. Exact optimal

solutions are obtained with the ISCpp algorithm. The correlation between the problem difficulty entropy value and the logarithm of CPU time by ISC is shown in Figure 1. The correlation coefficient is 0.910 and therefore this corresponds comparatively well.

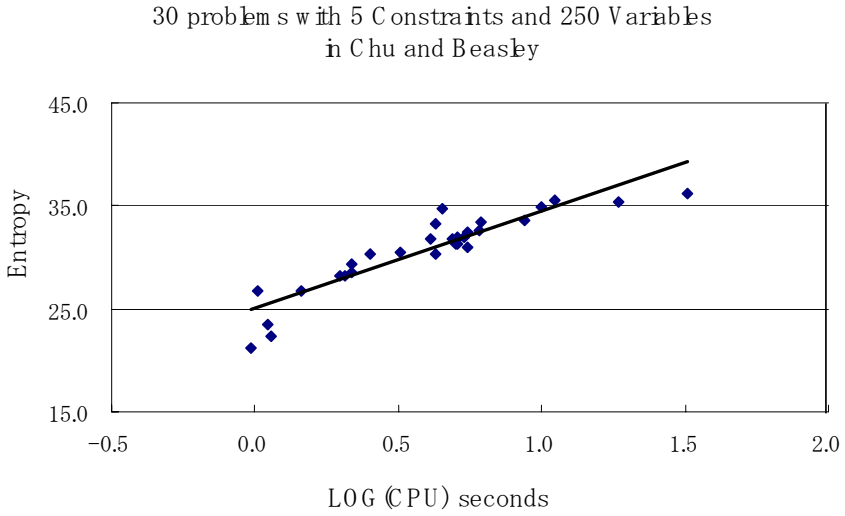

Figure 1 Computation time and difficulty entropy

As a comparison, consider the PGC (Percent Gap Closure) values of Karwan, Rardin and Sarin (1987) applied to the same test problems. The PGC value is given by

$$\frac{f^{\mathrm{UB}} - v^{\mathrm{Opt}}[\mathrm{P}^{\mathrm{SD}}]}{f^{\mathrm{UB}} - v^{\mathrm{Opt}}[\mathrm{P}]} \times 100$$

with $v^{\mathrm{Opt}}[\bullet]$ denoting the optimal objective function value of problem $\bullet$ and P^{SD} being the surrogate dual problem of the original problem P. The correlation coefficient between PGC value and the logarithm of the ISC CPU time is -0.072, hence no strong correlation can be found.

9. Computational Results

Our computational tests showed that our ISCpp method was able to solve several important classes of problems that other methods, such as CPLEX, performed very poorly and in some cases proving unable to solve even a single instance of the test problems considered. On the other hand, on those problem instances where CPLEX performed well, our method did not lag far behind in its performance.

Although the proposed ISCpp is general and accepts instances where variables may have a different number of alternative items, all test problems considered in this study are instances where all variables have the same number of items (i.e. $k_i = \kappa$ for all $i \in N$). Therefore, problem sizes may be expressed as $m \times n \times k$, without loss of generality.

We evaluated the performance of the ISCpp on 190 multidimensional nonlinear knapsack problem instances each of which are classified into one of three distinct sets, and within each set there are a number of subsets.

Set A contains three subsets of 0-1 knapsack problems each from Chu and Beasley (1998). The instances are characteristically more difficult to solve than arbitrary instances of similar size.

A1: Thirty 5 × 250 × 2 linear knapsack instances.

A2: Thirty 5 × 500 × 2 linear knapsack instances.

A3: Thirty 10 × 100 × 2 linear knapsack instances.

Set B contains non 0-1 (multi-valued) knapsack problems based on Petersen (1967), in which 11.6% of the coefficients are zero. There are twenty 5 × 50 × 11 convex quadratic knapsack problem instances.

The instances are generated by using variables restrictions of x_i = 0, 1, …, 2 or 10 instead of the original restriction x_i = 0 or 1 and the following convex quadratic objective functions is used:

$$\min \quad f(\mathbf{x}) = \sum_{i=1}^{n} c_i (x_i - 10)^2,$$

where c_i are coefficients of objective function of the Petersen problem. The right-hand-sides of the constraints in the instances are generated by setting

$b_j = floor\left(\gamma \sum_{i=1}^{n} a_{ji} k_i / 2\right)$ and 20 values of γ =0.60, 0.65, 0.70, …, 1.55.

Set C contains two subsets of non 0-1 knapsack problems based on Chu and Beasley (1998), which do not include any zero coefficients.

C1: 30 instances of a convex quadratic knapsack problem with size of 5 × 250 × 4,

C2: 30 instances of a cubic knapsack problem with size of 5 × 250 × 4,

where instances are generated by using variables restrictions of x_i = 0, 1, 2, or 3 instead of the original restriction x_i = 0 or 1, and convex quadratic and cubic objective functions are used for instances of C1 and C2, respectively. Subset C1 uses the following quadratic objective function:

$$\min \quad f(\mathbf{x}) = \sum_{i=1}^{n} c_i' (x_i - 10)^2,$$

while subset C2 uses the following cubic objective function:

$$\max \ f(\mathbf{x}) = \sum_{i=1}^{n} c_i' x_i + a_{1i}' x_i^{\,2} + a_{2i}' x_i^{\,3},$$

where c_i', a_{1i}', and a_{2i}' are the coefficients of the objective function, first constraint, and second constraint of the Chu and Beasley problem, respectively. The cubic function comes from Cooper (1980). Specifically, set A corresponds to 0-1 linear knapsack problems. Sets B and C are multi-valued smooth (differentiable) optimization problems. More details about the characteristics of the test sets will be given as the results are presented and analyzed. We first define the parameters considered in the implementation of the algorithm as well as the specification of the computational

environment used in our experiments. All computational tests were carried out on a Pentium IV, 3.2GHz personal computer with 2GB of memory.

The following parameters were considered in the definition of the various functions of the ISCpp. PE is a depth-first strategy that preferentially selects the problem containing fewer active variables. FVS selects the variable with the highest entropy value. PDE is defined by the proposed problem difficulty entropy metric. As noted, a problem is considered easy if the corresponding entropy is small. Finally, we used the same threshold values, σ, for each subset, which is used to determine whether a sub-problem should be partitioned. The value of σ can be determined roughly by using trial runs lasting for several minutes for several difficult problems in each subset. A threshold of σ = 35 is used in the cases except set B where we use σ =22. Empirically each instance seems to have its own proper value of σ. Developing more appropriate procedures for determining the value of σ remains a topic for future research.

We now undertake the computational analysis of the algorithm on the different data sets. For convenience of the analysis we present summary tables of results for each problem set and refer to the Appendix section for the full set of results. Tables in the Appendix are numbered using the name of the corresponding problem class (or subset) to ease identification.

We first discuss the problems where at least one competing method (namely CPLEX) performed well, starting with problem Set A. Problems in Set A, despite their relatively small size, are characteristically more difficult to solve than arbitrary instances of similar size due to the inherent correlation between the coefficients of the objective function and those of the constraints. For a comparative analysis, we provide results for ISCpp and CPLEX V9.0. Results for set A are summarized in Table 2. A full set of results for individual classes of problems in set A are given in Tables A1, A2, and A3 in the appendix.

					CPU Time					
Problem Class	Entropy Ave	Min	Max	Proportion Using PP	ISCpp Average	Min	Max	CPLEX Average	Min	Max
A1	30.7	21.1	36.2	0/30	5.70	0.97	32.05	168.38	6.91	857.80
A2	39.9	31.2	48.3	26/30	272.47	7.00	970.50	3783.67*	234.03	8507.64*
A3	34.9	27.3	40.1	13/30	170.30	3.98	841.24	30.86	1.44	141.89

* Over the 29 instances that were solved.

Table 2 Results Summary for Set A Problem Instances

ISCpp yields exact optimal solutions for all 90 test instances of set A. On the other hand, CPLEX failed to solve one of these problems (A2-8) (as seen in Table A2) and, on average, required roughly an order of magnitude more computation time. The effectiveness of our approach in these cases is noteworthy because the problems considered are at the limit of those that surrogate constraint methods are supposedly capable of handling. Normally such methods are considered useful for solving instances containing a large number of variables but a relatively small number of constraints. In many practical applications surrogate constraints methods are of limited effectiveness for problems containing more than half a dozen constraints. Problems in subset A3 contain 10 constraints and 100 variables, which are considered very difficult for surrogate constraint methods. Indeed CPLEX is faster than ISCpp on these A3 set as shown in Table 2. The average computational time of the ISCpp over all 90 problems is 149.49 (or 145.54 for the 89 problems solved by CPLEX) seconds on a Pentium IV 3.2GHz while CPLEX took on average 1327.64 seconds on the same computer for the 89 problems it could solve.

For problem sets B and C, which contain more difficult problem structures that include nonlinearities as well as discrete conditions, we initially intended to compare the ISCpp method to some of the major global optimization solvers in the field namely Bonmin, Baron, and the Interval Global Solver in the Frontline Premium Solver Platform version 6.5. Additionally, for quadratic convex knapsack instances, we use CPLEX and the LP/Quadratic Solver in Frontline Premium Solver Platform.

Unfortunately, Baron fails to produces exact solutions even for small instances (5 constraints, 10 variables and 10 items). The Interval Global Solver is too slow to use for comparison. The LP/Quadratic Solver fails to produces exact solutions for some quadratic multi-valued integer knapsack instances. Therefore we only compare Bonmin and CPLEX to the ISCpp.

The Bonmin algorithm finds the global optimum when the objective and constraint functions are convex but is only a heuristic when this is not the case. CPLEX can solve convex quadratic programs and linear programs.

The computational results for set B are summarized in Table 3. The ISC without problem partitioning (non PP) could solve seven of the 20 problems. ISCpp showed its superiority by being able to solve 10 of the remaining 13 problems. The final three problems could not be solved due to running out of memory. When Bonmin was tested on a sample of four problems it proved capable of solving only one of them to completion, finding a suboptimal solution to the other three but running out of memory before finishing. CPLEX is clearly the dominant technique here solving the problems instances in less than 4 seconds. Clearly the cutting-plane strategies of CPLEX work very well for set B.

For the convex quadratic problem set, C1, six of the thirty problem instances could be solved with the ISC method without problem partitioning. The remaining problems required the use of ISCpp to be solved. Bonmin could not be used to solve these instances as it halts abnormally when used to solve any instance of this class of problem. The cutting plane strategies that were very successfully used by CPLEX to solve problem set B do not work very well for problem set C1. The CPLEX quadratic solver does not solve any instance in C1 due to it running out of memory, even though a computer with 4 GB memory was used. On the other hand, the ISCpp solved all C1

instances on a computer with just 1 GB memory. This success is due to the problem partitioning technique.

CPLEX and Bonmin cannot solve the cubic instances in problem set C2. It should be noted that Bonmin can solve non-separable convex nonlinear problems. CPLEX does not solve nonlinear instances except for certain types of convex quadratic problems. ISCpp is designed to solve separable nonlinear instances. Note that the differentiable knapsacks are quite difficult to solve when the objective or constraint function is not convex. We are unaware of any other solver that can yield exact optimal solutions for problems with non-convex functions like those solved in C2.

Problem no.	Entropy	CPU Sec.		
		ISC	Bonmin	CPLEX
B-00	11.5	0.05	7259.0	0.36
B-04	15.9	0.15	> 5141.8	1.72
B-07	17.2	0.58	> 5249.8	0.64
B-15	19.4	1.35	> 6350.3	3.38

Table 3. CPU times for the four sample set B Instances solved by all solvers

The corresponding correlation coefficients between the problem difficulty (i.e. entropy value) and common logarithm of CPU time of the ISCpp are shown in Table 4.

Problem subset and type		$m \times n \times k$	PP rate ISCpp	ISCpp correlation with entropy	Winning ratio of CPLEX over ISCpp
A1	Linear	5×250×2	0/30	0.91	0/30
A2	Linear	5×500×2	26/30	0.94	0/30
A3	Linear	10×100×2	13/30	0.94	30/30
B	VexQ[1]	5×50×11	13/20	0.95	16/20
C1	VexQ[1]	5×250×4	24/30	0.66	0/30
C2	CavC[2]	5×250×4	15/30	0.90	

Table 4 The correlation between the entropy and the logarithm of CPU time

The CPU time of the ISCpp is highly correlated with the entropy except for subset C1. The correlation between the entropy and the CPU time of the ISC without PP technique is 0.74 but the correlation between entropy and ISCpp is only 0.66. The ISC using PP seems to have an overhead in the computation which affects the correlation value.

We hypothesized that the entropy data in Table 1 is applicable to other nonlinear knapsack problems. The computational results show that this hypothesis seems reasonable.

10. Summary and Conclusion

It is well-established that problem size is not the only factor that makes a problem difficult to solve. In combinatorial optimization, where the feasible solution space is finite and discrete, it is important to find effective techniques to reduce the effect of the intrinsic combinatorial explosion. The degree by which the domain variable or solution space may be reduced is inversely related to the complexity (or difficulty) of the problem. A number of techniques have been proposed to determine appropriate metrics to estimate the problem difficulty.

Unlike other methods that relate problem difficulty to the duality gap for a representative set of instances, we propose a new method to estimate the problem difficulty based on the concept of variable uncertainty as measured by entropy. The variable uncertainty shows the degree of difficulty in determining the value a variable should take in the optimal solution. Problem difficulty is a function of the uncertainty defined over the set of all problem variables. Statistical analysis carried out on various samples of problems proves the effectiveness of the proposed technique to estimate the problem difficulty, revealing that the underlying entropy metric provides an appropriate scale by which the degree of difficulty of the problem can be measured.

Our new method is a hybrid approach that combines problem partitioning with variable aggregation methods for the solution of large scale multidimensional nonlinear knapsack problems (MNK). The combinatorial explosion is tamed by successively partitioning a problem into smaller problems induced by the variable of maximum difficulty in a (sub)problem, and iteratively aggregating variables so that the problem is reduced to a minimum number of difficult variables. An enumeration procedure is then applied to solve the problem for the restricted set of difficult variables.

Tests carried out on a variety of complex MNK problems clearly demonstrate the efficiency of our method to solve relatively large scale problems. These outcomes invite further investigation of the method's potential to solve more complex problems and of generalizations to other combinatorial optimization problems. In this paper, we treat problems involving ten or fewer constraints, leaving the consideration of problems with more constraints as a topic for future research.

References

Achterberg, T., T. Kocha, A. Martin. 2005. Branching rules revisited. Oper. Res. Lett. **33** 42-54.

Akcay, Y., S. H. Xu. 2004. Joint inventory replenishment and component allocation Optimization in an assemble-to-order system. Manage. Sci. **50** 99-116.

Armstrong R. D., D. S. Kung, P. Sinha, A. A. Zoltners. 1983. A computational study of a multiple-choice knapsack algorithm, ACM T. Math. Software, **9** 184-198.

Armstrong, R. D, P. Sinha, and A.A. Zoltners. 1982. "The Multiple-Choice Nested Knapsack Model," Manage Sci. 28 34-43

Bapna, R., P. Goes, A. Gupta. 2005. Pricing and allocation for quality-differentiated online Services. Manage. Sci. **51** 1141-1150.

Bertsimas; D., R. Demir. 2002. An approximate dynamic programming approach to multidimensional knapsack problems. Manage. Sci. **48** 550-565.

Bitrang, R., D. Tirupati. 1989a. Tradeoff curves, Targeting and balancing in manufacturing queueing networks. Oper. Res. **37** 547-564.

Bitrang, . R., D. Tirupati. 1989b. Capacity planning in manufacturing networks with discrete options. Ann. Oper. Res. **17** 119-135.

Bretthauer K. M., B. Shetty. 1995. The nonlinear resource allocation problem. Oper. Res. **43** 670-683.

Bonami, P., J. Lee. 2006. BONMIN users' manual.

Bonami, P., L.T. Biegler, A.R. Conn, G. Cornuejols, I.E. Grossmann, C.D. Laird, J. Lee, A. Lodi, F. Margot, N.Sawaya and A. Waechter. 2005. An Algorithmic Framework for Convex Mixed Integer Nonlinear Programs. IBM Research Report RC23771.

Chu P. C., J.E. Beasley. 1998. A genetic algorithm for the multidimensional knapsack problem. J. Heuristics. **6** 63-86.

Coit D. W., A.E. Smith. 1996a. Reliability optimization of series-parallel systems using a genetic algorithm. IEEE T. Reliab. **45** 254-260.

Coit, D.W., A. E. Smith. 1996b. Adaptive penalty methods for genetic optimization of constrainted Combinatorial problems. INFORMS J. Comput. **8** 173-182.

Cooper M., W. 1980. The use of dynamic programming for the solution of a class of nonlinear programming problems. Nav. Res. Logist. Q. **27** 89-95.

Cooper, M. W. 1981. Survey of methods of pure nonlinear integer programming. Manage. Sci. **27** 353-361.

Degraeve, Z., M. Vandebroek. 1998., A mixed integer programming model for solving a layout problem in the fashion industry. Manage. Sci. **44** 301-310.

Dyer M.E. 1980. Calculating surrogate constraints. Math. Program. **19** 255-278.

Dyer M. E., N. Kayal and J. Walker. 1984. A branch and bound algorithm for solving the multiple-choice knapsack problem, J. Comput. Appl. Math., **11** 231-249.

Elhedhli, S., J. L. Goffin. 2005. Efficient production-distribution system design. Manage. Sci. **51** 1151-1164.

Fox, B., 1966. Discrete optimization via marginal analysis, Manage. Sci., **13** 210-216.

Gerla M, L. Kleinrock. 1977. On the topological design of distributed computer networks. IEEE T. Commun. **25** 48-60.

Glover F. 1965. A multiphase-dual algorithm for the zero-one integer programming problem. Oper. Res. **13** 879-919.

Hochbaum D.S. 1995. A nonlinear knapsack problem. Oper. Res. Lett. **17** 103-110.

Hsieh, Y. (2002) A linear approximation for redundant reliability problems with multiple component choices. Comput. Ind. Eng. **44** 91–103.

Hughes, E., J. N. K. Rao. 1979. Some problems of optimal allocation in sample surveys involving inequality constraints. Commun. Stat. **15**, 1551-1574.

Ibaraki T. , N. Katoh. 1988. Resource allocation problems: Algorithmic Approaches. MIT Press, New York.

James, R. J. W., Y. Nakagawa. 2005. Enumeration methods for repeatedly solving multidimensional knapsack sub-problems. IEICE T. Inf. Sys., **E88-D** 2329-2340.

Karwan M.H., R.L. Rardin, S. Sarin. 1987. A new surrogate dual multiplier search procedure. Nav. Res. Log. **34** 431-450.

Liang Y-C., A. Smith. 2004. An ant colony optimization algorithm for the redundancy allocation problem (RAP). IEEE T. Reliab. **53** 417-423

Marsten R.E., T.L. Morin. 1978. A hybrid approach to discrete mathematical programming, Math. Program. **14** 21-40.

Martello, S., D. Pisinger, P. Toth. 1999. Dynamic programming and strong bounds for the 0-1 knapsack problem. Manage. Sci. **45** 414-424.

Martello, S., P. Toth. 2003. An exact algorithm for the two-constraint 0-1 knapsack problem. Oper. Res. **51** 826-835.

Mathur K., H.M. Salkin, S. Morito. 1983. A Branch and search algorithm for a class of nonlinear Knapsack problems, Oper. Res. Lett. **2** 155-160.

Melachrinoudis E., G. Kozanidis. 2002. A mixed integer knapsack model for allocating funds to highway safety improvements, Transport. Res. A-Pol. **36** 789–803.

Nakagawa Y., S. Miyazaki. 1981. Surrogate constraints algorithm for reliability optimization problems with two constraints. IEEE Trans. Reliab. **R-30** (2) 175-180.

Nakagawa Y., M. Hikita, H. Kamada. 1984. Surrogate constraints algorithm for reliability optimization problems with multiple constraints. IEEE Trans. Reliab. **R-33** (4) 301-305.

Nakagawa Y. 1990. A new method for discrete optimization problems. Electro. Comm. Jpn. **73** (3) 99–106. (Translated from Trans. IEICE. **J73-A** (3) 1990, 550-556 (in Japanese))

Nakagawa Y., A. Iwasaki. 1999. Modular approach for solving nonlinear knapsack problems. IEICE Tr. Fun. Electr. **E82-A** 1860-1864.

Nakagawa Y. 2003. An improved surrogate constraints method for separable nonlinear integer programming. J. Oper. Res. Soc. Jpn. **46** 145-163.

Nakagawa, Y., Y. Isada, R.J.W. James. 2005. Interactive improved surrogate constraints method for nonconvex portfolio optimization problems. Working Paper. Faculty of Informatics, Kansai University.

Nakagawa, Y., R.J.W. James, C Rego, F. Glover. 2009. Interactive surrogate constraints method for an plus-alpha portfolio optimization. Working Paper. Faculty of Informatics, Kansai University.

Nauss R. M. 1978. The 0-1 Knapsack Problem with Multiple Choice Constraints, Eur. J. Oper. Res. **2** 125-131.

Ohnishi, J, S. Kimura, R.J.W. James, Y. Nakagawa. 2007. Solving the redundancy allocation problem with a mix of components using the improved surrogate constraint method, IEEE Trans. Reliab. **56** 94-101.

Ng, K.Y.K., N.G.F. Sancho 2001. A hybrid dynamic programming depth-first search algorithm with an application to redundancy allocation. IIE Trans. **33** 1047–1058.

Petersen, C.C. 1967. Computational Experience with Variants of the Balas Algorithm Applied to the Selection of R&D Projects. Manage. Sci. **13** 736-750.

Ralphs, T.K. 2006. Parallel Branch and Cut, in Parallel Combinatorial Optimization, E. Talbi, ed., Wiley (Working paper version: http://www.lehigh.edu/~tkr2/cv/).

Premium Solver Platform User Guide. 2005. Frontline Systems, Inc.

Sahinidis, N. V., M. Tawarmalani 2005. BARON 7.2.5: Global Optimization of Mixed-Integer Nonlinear Programs, User's manual.

Shannon C. 1948. A mathematical theory of communication. Bell Sys. Tech. J. **27**, 379-423.

Sinha P., A. A. Zoltners. 1979. The multiple-choice knapsack problem Oper. Res. **27**, 125-131.

Tawarmalani, M., N. V. Sahinidis. 2004. Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. Math. Program., **99** 563-591.

Takaoka, T. 1998. Measure of Disorder. Proc. CATS (Computation: Australasian Theory Symposium), 77-85.

Ziegler H. 1982. Solving certain singly constrained convex optimization problems in production planning. Oper. Res. Lett. **1** 246-252.

Zoltners A. A., P. Sinha. 1980. Integer programming models for sales resource allocation. Manage. Sci. **26** 242-260

Appendix

Problem No.	f^{Exact}	Entropy	nonPP or PP	CPU time (sec.)	
				ISCpp	CPLEX V9.0
A1-00	59312	26.7	non	1.5	22.5
A1-01	61472	31.4	non	5.0	161.0
A1-02	62130	26.7	non	1.0	8.8
A1-03	59463	36.2	non	32.0	857.8
A1-04	58951	31.8	non	4.9	270.1
A1-05	60077	34.9	non	10.0	451.3
A1-06	60414	28.1	non	2.0	111.8
A1-07	61472	33.4	non	6.1	198.3
A1-08	61885	30.5	non	3.2	60.5
A1-09	58959	23.4	non	1.1	16.0
A1-10	109109	32.6	non	6.1	125.9
A1-11	109841	28.5	non	2.2	38.0
A1-12	108508	31.8	non	4.1	144.0
A1-13	109383	35.5	non	11.1	151.6
A1-14	110720	35.3	non	18.5	618.6
A1-15	110256	32.4	non	5.5	223.6
A1-16	109040	30.3	non	4.3	184.7
A1-17	109042	33.5	non	8.7	309.9
A1-18	109971	34.8	non	4.5	162.9
A1-19	107058	31.3	non	5.1	174.6
A1-20	149665	31.8	non	5.1	99.0
A1-21	155944	30.3	non	2.5	59.3
A1-22	149334	31.9	non	5.4	148.6
A1-23	152130	28.2	non	2.0	85.7
A1-24	150353	33.3	non	4.3	87.4
A1-25	150045	22.3	non	1.1	9.2
A1-26	148607	21.1	non	1.0	6.9
A1-27	149782	31.0	non	5.5	129.2
A1-28	155075	29.3	non	2.2	19.7
A1-29	154668	32.0	non	5.1	114.9
Average		30.7	0/30	5.7	168.4

Table A1 Computation time (sec.) of the ISCpp and CPLEX 9.0 for subset A1

Problem No.	f^{Exact}	Entropy	nonPP or PP	CPU time (sec.)	
				ISCpp	CPLEX
A2-00	120148	44.5	PP	574.8	8507.6
A2-01	117879	37.1	PP	61.7	1086.6
A2-02	121131	41.7	PP	317.0	4993.7
A2-03	120804	40.3	PP	245.4	5244.8
A2-04	122319	43.4	PP	255.4	2039.1
A2-05	122024	43.5	PP	401.5	3965.8
A2-06	119127	44.9	PP	695.0	7608.7
A2-07	120568	38.4	PP	120.8	3178.6
A2-08	121586	44.9	PP	615.8	-
A2-09	120717	41.9	PP	435.2	6412.3
A2-10	218428	42.5	PP	464.5	4268.1
A2-11	221202	38.8	PP	52.5	2692.0
A2-12	217542	44.8	PP	856.1	7492.9
A2-13	223560	43.7	PP	713.2	7988.9
A2-14	218966	39.2	PP	54.5	646.7
A2-15	220530	39.3	PP	191.9	5922.1
A2-16	219989	39.1	PP	176.9	2004.6
A2-17	218215	36.3	PP	44.4	3031.4
A2-18	216976	40.6	PP	185.5	4367.0
A2-19	219719	48.3	PP	970.5	4228.9
A2-20	295828	35.4	non	12.4	234.0
A2-21	308086	38.2	PP	107.3	3409.8
A2-22	299796	37.8	PP	35.9	528.2
A2-23	306480	35.5	non	23.3	5223.5
A2-24	300342	39.9	PP	150.7	1845.8
A2-25	302571	37.6	PP	36.8	1876.4
A2-26	301339	32.8	non	13.6	818.9
A2-27	306454	31.2	non	7.0	550.9
A2-28	302828	36.6	PP	55.5	1873.3
A2-29	299910	39.6	PP	298.9	7685.7
Average			26/30	272.5	3783.7

Table A2 Computation time (sec.) of the ISCpp and CPLEX 9.0 for subset A2

Problem No.	f^{Exact}	Entropy	non PP or PP	CPU time (sec.)	
				ISCpp	CPLEX V9.0
A3-00	23064	39.2	PP	184.9	103.0
A3-01	22801	39.2	PP	389.0	69.3
A3-02	22131	35.3	non	43.6	15.6
A3-03	22772	38.2	PP	565.4	141.9
A3-04	22751	34.1	non	53.7	7.5
A3-05	22777	38.9	PP	820.2	99.4
A3-06	21875	34.8	non	38.3	19.1
A3-07	22635	32.6	non	27.8	8.2
A3-08	22511	32.7	non	15.6	12.2
A3-09	22702	37.1	PP	253.9	23.3
A3-10	41395	39.5	PP	133.7	41.4
A3-11	42344	33.0	non	22.5	16.2
A3-12	42401	37.0	PP	103.2	23.4
A3-13	45624	36.4	PP	177.9	49.4
A3-14	41884	35.3	non	75.3	16.4
A3-15	42995	39.5	PP	233.0	35.8
A3-16	43574	40.1	PP	841.2	48.5
A3-17	42970	36.8	PP	176.4	19.1
A3-18	42212	38.8	PP	187.0	27.0
A3-19	41207	39.9	PP	392.6	52.1
A3-20	57375	27.3	non	4.0	1.4
A3-21	58978	33.7	non	68.5	20.4
A3-22	58391	33.5	non	43.0	24.2
A3-23	61966	28.5	non	4.6	2.7
A3-24	60803	31.5	non	5.6	4.4
A3-25	61437	31.3	non	9.4	6.5
A3-26	56377	35.5	non	211.5	23.9
A3-27	59391	27.4	non	4.0	2.5
A3-28	60205	32.9	non	16.9	5.6
A3-29	60633	28.1	non	6.0	5.7
Average		34.9	13/30	170.3	30.9

Table A3 Computation time (sec.) of the ISCpp and CPLEX 9.0 for subset A3

Problem No.	f^{Exact}	Entropy	nonPP or PP	CPU time (sec.)		
				ISCpp	Bonmin	CPLEX
B-00	102723	11.5	non	0.03	7259.0	0.4
B-01	128040	59.5	PP	-	-	1.1
B-02	153492	22.7	non	13.7	-	1.8
B-03	181954	26.4	PP	29.1	-	1.9
B-04	212669	15.9	non	0.1	> 5141.8	1.7
B-05	248250	63.8	PP	-	-	4.0
B-06	283298	25.3	PP	21.2	-	1.1
B-07	320495	17.2	non	0.4	> 5249.8	0.6
B-08	361978	21.1	non	8.0	-	0.7
B-09	406805	40.0	PP	196.9	-	1.0
B-10	452572	41.8	PP	3893.6	-	0.7
B-11	500180	28.7	PP	61.9	-	0.9
B-12	552201	28.3	PP	33.5	-	2.0
B-13	606288	33.9	PP	853.9	-	0.9
B-14	661569	44.3	PP	5784.1	-	2.3
B-15	718854	19.4	non	1.0	> 6350.3	3.4
B-16	780545	20.4	non	10.9	-	0.5
B-17	847078	62.7	PP	-	-	2.0
B-18	912881	36.9	PP	413.2	-	0.9
B-19	981939	28.7	PP	21.6	-	0.7
Average		32.4	13/20	667.2		1.4

Table B Computation time (sec.) of the ISCpp, Bonmin and CPLEX 9.0 for set B

Problem No.	Objective value f^{Exact}	Entropy	nonPP or PP	CPU time (sec.)	
				ISCpp	CPLEX Quadratic
C1-00	16905983	40.6	PP	33818.6	> 2387.25
C1-01	17661527	29.9	non	22.4	> 2447.75
C1-02	17826900	38.9	PP	38036.7	>2451.74
C1-03	17640880	38.6	PP	6989.1	>2370.43
C1-04	17213361	41.1	PP	11952.3	>2477.05
C1-05	17472685	45.8	PP	271482.0	>2894.87
C1-06	17785584	35.5	PP	14732.8	>2447.80
C1-07	18018498	41.0	PP	33339.7	>2486.93
C1-08	17809662	32.7	non	186.3	>2668.93
C1-09	17292443	35.2	PP	3584.3	>2312.29
C1-10	16602967	43.7	PP	4613.2	>2237.97
C1-11	16350115	39.1	PP	1779.1	>2207.89
C1-12	16280824	38.0	PP	310.1	>1969.29
C1-13	16406519	46.4	PP	64816.2	>2133.15
C1-14	16938577	49.9	PP	8299.6	>1937.81
C1-15	16657783	35.6	non	103.7	>2387.54
C1-16	16345554	43.2	PP	3108.6	>1984.31
C1-17	16543132	44.8	PP	22291.1	>1940.24
C1-18	16667744	37.2	non	143.1	>1903.12
C1-19	16118851	37.2	non	230.5	>2244.81
C1-20	15402773	51.5	PP	10688.0	>1926.91
C1-21	15922625	45.8	PP	3389.6	>1983.50
C1-22	15399844	39.5	PP	310.9	>1927.60
C1-23	15591875	44.6	PP	2111.4	>1920.39
C1-24	15357276	47.8	PP	6925.7	>1944.02
C1-25	15282034	40.7	PP	667.5	>1898.07
C1-26	15231092	33.2	non	27.3	>1911.12
C1-27	15205396	41.3	PP	1308.8	>1849.41
C1-28	15841810	45.8	PP	5110.2	>1903.38
C1-29	15793696	43.5	PP	1814.0	>1839.61
Average		40.9	24/30	876.1	

Table C1 Computation time (sec.) of the ISCpp and CPLEX 9.0 for subset C1

Problem No.	f^{Exact} ISCpp	Entropy	nonPP or PP	CPU time (sec.) ISCpp
C2-00	425472	35.3	non	24.28
C2-01	445419	42.6	non	178.7
C2-02	436044	39.6	non	490.6
C2-03	433893	45.1	non	470.9
C2-04	431991	43.8	non	729.7
C2-05	428604	36.5	non	51.9
C2-06	443166	42.9	non	1319.5
C2-07	446634	39.5	non	287.9
C2-08	445779	39.3	non	121.7
C2-09	444030	39.4	non	137.1
C2-10	878154	42.7	non	1242.0
C2-11	887025	43.7	PP	2628.0
C2-12	829734	32.1	non	21.5
C2-13	877134	52.0	PP	7073.4
C2-14	891336	48.5	PP	6208.1
C2-15	875385	44.2	non	1149.9
C2-16	875364	42.6	non	2041.2
C2-17	857100	49.2	PP	2778.0
C2-18	866436	48.1	PP	4460.1
C2-19	842871	46.5	PP	2314.9
C2-20	1307934	48.0	PP	5679.2
C2-21	1285269	53.9	PP	9423.3
C2-22	1354146	53.5	PP	3057.5
C2-23	1316532	56.6	PP	8773.4
C2-24	1272156	47.9	PP	2845.4
C2-25	1279689	44.9	non	2123.2
C2-26	1267677	54.7	PP	9566.4
C2-27	1248228	53.3	PP	4734.9
C2-28	1310874	48.0	PP	2833.3
C2-29	1386237	47.3	PP	776.0
Average		45.4	15/30	2784.7

Table C2 Computation time (sec.) of the ISCpp for subset C2

Acknowledgments

This research was supported in part by the “Academic Frontier” Project (2003) and the Grant-in-Aid for Scientific Research (C) 14580397 and [B] 19300003 of the Ministry of Science, Education and Culture of Japan. Part of this research was conducted when Cesar Rego was visiting MIT Sloan School of Management.