0-1Knapsack Problems with Random Budgets

by

Shubhabrata Das &
Diptesh Ghosh*

February 2001

*Faculty of Economic Sciences
University of Groningen
The Netherlands

Please address all correspondence to:

Dr. Shubhabrata Das
Assistant Professor
Indian Institute of Management
Bannerghatta Road
Bangalore – 560 076
Fax: (080) 6584050
E-mail: shubho@iimb.ernet.in

0-1 Knapsack Problems with Random Budgets

Shubhabrata Das

QM&IS Area, Indian Institute of Management, Bangalore, India

Diptesh Ghosh

Faculty of Economic Sciences, University of Groningen, The Netherlands

Abstract

Given a set of elements, each having a profit and cost associated with it, and a budget, the 0-1 knapsack problem finds a subset of the elements with maximum possible combined profit subject to the combined cost not exceeding the budget. In this paper we study a stochastic version of the problem in which the budget is random. We propose two different formulations of this problem, based on different ways of handling infeasibilities, and propose exact and heuristic algorithms to solve the problems represented by these formulations. We also present the results from some computational experiments.

Keywords: static stochastic knapsack problem, random budget, infeasibility, branch and bound.

Given a budget B and a set of elements $\mathbf{E} = (e_1, \ldots, e_n)$ with a profit vector $\mathbf{P} = (p_1, \ldots, p_n) \in \Re_+^n$ and a cost vector $\mathbf{C} = (c_1, \ldots, c_n) \in \Re_+^n$, the *0-1 knapsack problem* (refer Martello and Toth [10] for a detailed introduction), denoted here by KNAP$(\mathbf{P}, \mathbf{C}, B)$, is the problem of finding a subset $S \subseteq \mathbf{E}$ which maximizes $\sum_{e_j \in S} p_j$ while satisfying $\sum_{e_j \in S} c_j \leq B$. If we identify a binary vector $\mathbf{x} = (x_1, \ldots, x_n)'$ with a set $S \subseteq \mathbf{E}$ by the relation $x_i = 1 \iff e_i \in S$, we can define KNAP$(\mathbf{P}, \mathbf{C}, B)$ as

$$\text{KNAP}(\mathbf{P}, \mathbf{C}, B) = \arg\max \{\mathbf{P}\mathbf{x} : \mathbf{C}\mathbf{x} \leq B, \mathbf{x} \in \{0,1\}^n\}. \tag{1}$$

The vector $\mathbf{x} \in \{0,1\}^n$ is called a *feasible* solution to KNAP$(\mathbf{P}, \mathbf{C}, B)$ (or simply, a solution) if it satisfies the budget constraint $\mathbf{C}\mathbf{x} \leq B$. An optimal or *maximum-profit* solution to KNAP$(\mathbf{P}, \mathbf{C}, B)$ is denoted by $\mathbf{x}_B^*$.

Stochastic versions of KNAP$(\mathbf{P}, \mathbf{C}, B)$ are of two forms, static and dynamic. Full information regarding the randomness of the problem elements is assumed to be known before the start of the solution process in the static version, while in the dynamic version, some of these parameters are unknown when the solution process starts. Steinberg and Park [13], Sniedovich [11, 12], Henig [7], Carraway *et al.* [4], and Yu [16] have studied static stochastic 0-1 knapsack problems in which the P vector is random. Cohn and Barnhart [5] studied a special case of these problems in which the P vector is a known linear function of the C vector and the C vector is random. Averbakh [1] studied probabilistic properties of a heuristic algorithm to solve integer linear programs with multiple knapsack constraints. Ghosh and Das [6] recently studied a general class of stochastic discrete optimization problems (of which KNAP$(\mathbf{P}, \mathbf{C}, B)$ with random P is a special case). Static stochastic knapsack problems can also be seen as a special case of general stochastic integer programming problems, especially in the literature on recourse-based approaches. Discussions on this topic can be found in Birge and Louveaux [3], and Van der Vlerk [15]. Dynamic stochastic knapsack problems have been studied by several authors, like Kleywegt and Papastavrou [8], Marchetti-Spaccamela and Vercellis [9] and Szkatuła [14].

In the current work we will study a static stochastic knapsack problem in which the budget is random. In the next section, we will introduce the problem, expand the notion of feasibility, and subsequently consider two different notions of optimality. In Section 2, we develop algorithms, both exact and heuristic, to solve these problems, and report the results of preliminary computational experimentations with these algorithms in Section 3. We conclude the work with a summary in Section 4 and point out possible directions for future research in this area.

1 Formulation of the Problem

We consider a stochastic version of KNAP($\mathbf{P}, \mathbf{C}, B$), denoted here by S-KNAP($\mathbf{P}, \mathbf{C}, B(F)$), in which the budget B is a random variable with a density function $f(\cdot)$ and a survival function $F(\cdot)$ defined by

$$F(b) = Pr[B \geq b].$$

We denote the support of B (equivalently F) by $[B_L, B_U]$, where B_L is non-negative, and B_U is not necessarily finite. The mean and upper α-th percentile of B are denoted by B_μ and $B_{(\alpha)}$, respectively, i.e

$$Pr[B \geq B_{(\alpha)}] = \alpha.$$

The stochasticity of the budget implies that (some) solutions will meet the budget requirements only some of the time. This calls for appropriate modifications in the notion of feasibility, as well as methods for dealing with the profits accrued from a solution when the random budget value falls below the cost of the solution.

Redefining Feasibility: There are several alternative ways of defining a feasible solution to stochastic knapsack problems with random budgets. Few obvious choices are to require that the budget constraint be met

- for all values in the range of B,
- for some values in the range of B,
- for an average value of B
- at least with a specified probability.

Accordingly, we may define the following notions of feasibility in the current setup:

Definition 1: A solution $\mathbf{x}$ is said to be a ***strongly feasible*** solution to S-KNAP($\mathbf{P}, \mathbf{C}, B(F)$) if $\mathbf{Cx} \leq B_L$, or equivalently, $F(\mathbf{Cx}) = 1$.

Definition 2: A solution $\mathbf{x}$ is said to be a ***weakly feasible*** solution to S-KNAP($\mathbf{P}, \mathbf{C}, B(F)$), if $\mathbf{Cx} \leq B_U$.

Definition 3: A solution $\mathbf{x}$ is said to be a ***mean feasible*** solution to S-KNAP($\mathbf{P}, \mathbf{C}, B(F)$), if $\mathbf{Cx} \leq B_\mu$.

Definition 4: A solution $\mathbf{x}$ is said to be ***feasible with a reliability coefficient*** α to S-KNAP($\mathbf{P}, \mathbf{C}, B(F)$), if $Pr[\mathbf{Cx} \leq B] \geq \alpha$, or equivalently, if $\mathbf{Cx} \leq B_{(\alpha)}$. A feasible solution with a reliability coefficient 0.5 is called a ***median feasible*** solution.

Remark 1: If $\mathcal{F}_S$, $\mathcal{F}_W$ and $\mathcal{F}_\mu$ denote the class of strongly-, weakly-, and mean-feasible solutions respectively, while $\mathcal{F}_{(\alpha)}$ stands for the class of feasible solutions with reliability coefficient α, it is easy to see that

$$\mathcal{F}_S \subseteq \mathcal{F}_{(\alpha)} \subseteq \mathcal{F}_W, \quad \mathcal{F}_S \subseteq \mathcal{F}_\mu \subseteq \mathcal{F}_W.$$

The concepts of mean and median feasibility coincide (i.e., $\mathcal{F}_\mu = \mathcal{F}_{(0.5)}$) in many cases, including when B has a symmetric distribution.

Remark 2: A sufficient (and almost necessary) condition for a solution $\mathbf{x}$ to be *weakly feasible* is $\mathrm{F}(\mathbf{Cx}) > 0$.

Remark 3: All the above-mentioned notions of feasibility in the stochastic knapsack problem reduce to the notion of feasibility in the deterministic problem when B is degenerate. This justifies our subsequent search for a suitably defined optimal solution to be restricted to one of the above defined class of solutions.

Remark 4: A binary vector $\mathbf{x}$ is a strongly feasible solution to S-KNAP$(\mathbf{P}, \mathbf{C}, \mathrm{B}(\mathrm{F}))$ if and only if it is a feasible solution of KNAP$(\mathbf{P}, \mathbf{C}, \mathrm{B}_\mathrm{L})$. Thus, the requirement of strong feasibility reduces the problem to its deterministic counterpart. Hence, we will be mostly restricting ourselves to weakly feasible solutions. However, the methodologies described here should be valid if one confines oneself to the classes $\mathcal{F}_{(\alpha)}$ or $\mathcal{F}_\mu$, with minor modifications.

Redefining Optimality: The definition of weak feasibility necessitates appropriate methods for dealing with the profits accrued from a solution when the random budget value falls below its cost. We use two approaches to deal with such situations. In the first approach, we discard any profit accrued from a solution if it is infeasible in a given scenario (i.e., for a given value b of B). In order to distinguish it from the profit in the non-stochastic problem, we will refer this profit

$$\Pi_\mathrm{T}(\mathbf{x}, \mathrm{b}) = \mathbf{Px} \times \mathbf{I}_{\mathbf{Cx} \leq \mathrm{b}}$$

as the *truncated profit* of the solution. (I is the usual indicator function.) In the second approach, we accept profits accrued from solutions whose costs exceed the budget, (but not B_U), but also include a penalty for the portion of the cost of the solution exceeding the budget. The profit value, thus obtained,

$$\Pi_\mathrm{P}(\mathbf{x}, \mathrm{b}) = \mathbf{Px} - \vartheta(\mathbf{Cx} - \mathrm{b}) \times \mathbf{I}_{\mathbf{Cx} > \mathrm{b}}$$

is called the *penalized profit* of the solution. $\vartheta(\cdot)$ is called a penalty (or recourse) function. In this work we restrict ourselves to linear penalty functions (i.e. $\vartheta(t) = \theta t$), although some situations may warrant more steep (viz. exponential) penalties. Note that both the truncated and penalized profits of a solution, being functions of B, are themselves random variables.

The most direct way of defining optimality for static stochastic 0-1 knapsack problems is to maximize the expected value of the truncated or penalized profits. The two profit criteria will, in general, lead to different optimal solutions. Another common approach to optimization in stochastic problems is in terms of the regret associated with a solution. Let us define the loss $\mathbb{L}(\mathbf{x}|\mathrm{B} = \mathrm{b})$ associated with a solution $\mathbf{x}$ as $\mathrm{L}(\Pi_\mathrm{J}(\mathbf{x}^*_\mathrm{b}, \mathrm{b}) - \Pi_\mathrm{J}(\mathbf{x}, \mathrm{b}))$, where $\mathbf{x}^*_\mathrm{b}$ is the maximum profit solution when the budget B equals to b, $\mathrm{J} = \mathrm{T}$ or P, and $\mathrm{L}(\cdot)$ is a non-decreasing function on $[0, \infty)$. An optimal solution may be defined as the one having minimum expected loss or *regret*, the expectation being taken over b. A third approach could be to find a solution with the minimum $\max_\mathrm{b} \mathbb{L}$ value. This corresponds to the minmax regret solution studied in Averbakh [2].

We confine ourselves in this paper to the first two approaches to optimization. The following lemma shows that the two are equivalent when $\mathrm{L}(\cdot)$ is linear (i.e. $\mathrm{L}(t) = \alpha t$, which is the most common form of loss and adopted here).

Lemma 1: In S-KNAP problems, maximizing the expected truncated (or penalized) profit is equivalent to minimizing the regret associated with the truncated (or penalized) profit provided the regret is defined through a *linear* loss function.

Proof: Let $L(t) = \alpha t$, where α is a positive constant. Then

$$\mathbb{L}_J(\mathbf{x}|b) = \alpha[\Pi_J(\mathbf{x}_b^*, b) - \Pi_J(\mathbf{x}, b)],$$

where J is either T for the truncated profit case, or P for the penalized profit case.

Note that the regret of solution $\mathbf{x}$ is

$$R_J(\mathbf{x}) = \mathbb{E}\,[\alpha(\Pi_J(\mathbf{x}_b^*, b) - \Pi_J(\mathbf{x}, b))] = \alpha(\mathbb{E}\,[\Pi_J(\mathbf{x}_b^*, b)] - \mathbb{E}\,[\Pi_J(\mathbf{x}, b)]),$$

the expectation being taken over b values. Since $\mathbb{E}\,[\Pi_J(\mathbf{x}_b^*, b)]$ does not involve $\mathbf{x}$, finding a member of $\arg\min\{R_J(\mathbf{x})\}$ is equivalent to finding a member of $\arg\min\{-\mathbb{E}\,[\Pi_J(\mathbf{x}, b)]$ i.e., a member of $\arg\max\{\mathbb{E}\,[\Pi_J(\mathbf{x}, b)]\}$. ■

We now obtain the general expressions for $\mathbb{E}\,[\Pi_J(\mathbf{x}, b)]$. It is easy to see that the expected value of the truncated profit is

$$\mathbb{E}\,[\Pi_T(\mathbf{x}, b)] = \int_{B_L}^{B_U} \mathbf{Px} \times I_{\mathbf{Cx} \le b}(-dF(b)) = (\mathbf{Px}) \cdot F(\mathbf{Cx}), \tag{2}$$

which implies that if we adopt the policy of discarding the profit accrued from solutions when they are infeasible in a scenario, then S-KNAP can be formulated as

$$\text{T-KNAP}(\mathbf{P}, \mathbf{C}, B(F)) = \arg\max\left\{Z_T(\mathbf{x}) = \mathbf{Px} \cdot F(\mathbf{Cx}) : \mathbf{Cx} \le B_U,\ \mathbf{x} \in \{0, 1\}^n\right\}. \tag{3}$$

Given a specific budget value b of B, the penalized profit, assuming a linear penalty $\vartheta(t) = \theta t$, of a weakly feasible solution $\mathbf{x}$ is

$$\Pi_P(\mathbf{x}, b) = \begin{cases} \mathbf{Px} & \text{if } \mathbf{Cx} \le b \\ \mathbf{Px} - \theta[\mathbf{Cx} - b] & \text{if } b < \mathbf{Cx}. \end{cases} \tag{4}$$

Expression (4) implies that the expected penalized profit of a strongly feasible solution is the same as its profit. For any other weakly feasible solution $\mathbf{x}$

$$\begin{aligned} \mathbb{E}\,[\Pi_P(\mathbf{x}, b)] &= \mathbf{Px} - \theta \int_{B_L}^{\mathbf{Cx}} (\mathbf{Cx} - b)\,(-dF(b)) \\ &= \mathbf{Px} - \theta\mathbf{Cx}[1 - F(\mathbf{Cx})] + \theta\mu(\mathbf{Cx}), \end{aligned} \tag{5}$$

where $\mu(t) = \int_{B_L}^{t} b\,(-dF(b))$. So adopting the linear penalty approach, S-KNAP can be formulated as

$$\begin{aligned} \text{P-KNAP}(\mathbf{P}, \mathbf{C}, B(F)) = \arg\max\Big\{&Z_P(\mathbf{x}) = \mathbf{Px} + \theta I_{\mathbf{Cx} > B_L}\{\mu(\mathbf{Cx}) - \mathbf{Cx}[1 - F(\mathbf{Cx})]\} : \\ &\mathbf{Cx} \le B_U,\ \mathbf{x} \in \{0, 1\}^n\Big\}. \end{aligned} \tag{6}$$

Remark 5: In the T-KNAP$(\mathbf{P}, \mathbf{C}, B(F))$, there is no loss of generality in restricting the solution space to $\mathcal{F}_W$, because for $Z_T(\mathbf{x}) = 0$ for any $\mathbf{x}$ with $\mathbf{Cx} > B_U$. That is not the case for P-KNAP$(\mathbf{P}, \mathbf{C}, B(F))$, where for some problem instances with a low enough value of θ, there may exist a $\mathbf{x}_0$ with $\mathbf{Cx}_0 > B_U$ such that

$$Z_P(\mathbf{x}_0) < \max\left\{Z_P(\mathbf{x}) :\ \mathbf{Cx} \le B_U,\ \mathbf{x} \in \{0, 1\}^n\right\}.$$

The justification of excluding such solutions in our consideration lies with the fact that the optimal (in any formulation) of a stochastic knapsack problem should reduce to that of a maximum profit solution of deterministic knapsack problem when the budget B has a degenerate (single-point) distribution.

Remark 6: Literature on general stochastic integer programming problems suggests mainly two approaches to deal with randomness in constraint coefficients. The first is through recourse functions, a special case of which is essentially our P-KNAP formulation. The second practice is to limit to the feasible solution belonging to $\mathcal{F}_{(\alpha)}$, however without any alterations in the objective function (unlike our P-KNAP or T-KNAP formulation).

We conclude this section with deriving the functional forms of $Z_T(\mathbf{x})$ and $Z_P(\mathbf{x})$ for some common probability distributions. Since $Z_T(\mathbf{x}) = Z_P(\mathbf{x}) = \mathbf{Px}$ for all strongly feasible solutions, the expressions given below are valid for other weakly feasible solutions only.

Example 1: Suppose B has a Uniform distribution on $[B_L, B_U]$. Then

$$F(b) = \begin{cases} 1 & \text{if } b \le B_L \\ \frac{B_U - b}{B_U - B_L} & \text{if } B_L < b < B_U \\ 0 & \text{if } b \ge B_U \end{cases}$$

and $\mu(b) = \frac{b^2 - B_L^2}{2(B_U - B_L)}$ if $B_L \le b \le B_U$. Thus,

$$Z_T(\mathbf{x}) = \mathbf{Px}\,\frac{B_U - \mathbf{Cx}}{B_U - B_L}, \tag{7}$$

and

$$\begin{aligned} Z_P(\mathbf{x}) &= \mathbf{Px} - \theta\mathbf{Cx}\,\frac{\mathbf{Cx} - B_L}{B_U - B_L} + \theta\frac{(\mathbf{Cx})^2 - B_L^2}{2(B_U - B_L)} \\ &= \mathbf{Px} - \theta\frac{(\mathbf{Cx} - B_L)^2}{2(B_U - B_L)}. \end{aligned} \tag{8}$$

Example 2: Suppose B is normally distributed with mean μ and variance σ^2, i.e. its density is given by $f(b) = \frac{1}{\sigma}\phi(\frac{b-\mu}{\sigma})$, where

$$\phi(t) = \frac{1}{\sqrt{2\pi}}\exp(-\frac{t^2}{2}), \quad \text{and} \quad \Phi(t) = \int_{-\infty}^{t}\phi(u)du = 1 - \Phi(-t)$$

are the density and the distribution of the standard Normal distribution. Note that the survival function of B is $F(b) = \Phi(\frac{\mu-b}{\sigma})$, and $\mu(b) = \mu\Phi(\frac{b-\mu}{\sigma}) - \sigma\phi(\frac{b-\mu}{\sigma})$. Thus

$$Z_T(\mathbf{x}) = \mathbf{Px}\,\Phi(\frac{\mu - \mathbf{Cx}}{\sigma}), \tag{9}$$

and

$$Z_P(\mathbf{x}) = \mathbf{Px} - \theta\,\mathbf{Cx}\,\Phi(\frac{\mathbf{Cx} - \mu}{\sigma}) + \theta\mu\Phi(\frac{\mathbf{Cx} - \mu}{\sigma}) - \theta\sigma\phi(\frac{\mathbf{Cx} - \mu}{\sigma}). \tag{10}$$

Example 3: Suppose B has a shifted exponential distribution on $[B_L, \infty]$, i.e.

$$f(b) = \begin{cases} \lambda\exp[-\lambda(b - B_L)] & \text{if} \quad b > B_L \\ 0 & \text{otherwise.} \end{cases}$$

Hence

$$F(b) = \begin{cases} 1 & \text{if } b \le B_L \\ \exp[-\lambda(b - B_L)] & \text{otherwise,} \end{cases}$$

and

$$\mu(b) = \int_{B_L}^{b} t\lambda \exp[-\lambda(t - B_L)]\, dt = (B_L + \frac{1}{\lambda}) - (b + \frac{1}{\lambda})\exp[-\lambda(b - B_L)].$$

Thus,

$$Z_T(\mathbf{x}) = \mathbf{Px}\ \exp[-\lambda(\mathbf{Cx} - B_L)], \tag{11}$$

and

$$Z_P(\mathbf{x}) = \mathbf{Px} - \theta\mathbf{Cx}\Big[1 + 2\exp[-\lambda(\mathbf{Cx} - B_L)]\Big] + \frac{\theta}{\lambda}\Big[1 - \exp[-\lambda(\mathbf{Cx} - B_L)]\Big] + \theta B_L \tag{12}$$

2 Solution Techniques

We devise exact algorithms for solving T-KNAP($\mathbf{P}, \mathbf{C}, B(F)$) and P-KNAP($\mathbf{P}, \mathbf{C}, B(F)$) in the Subsection 2.1. While the general version of the latter problems are well-studied in the literature, and consequently several efficient exact algorithms exist, we also present an exact algorithm not only for the sake of completeness, but also because a presented variation of this algorithm can be useful when one has only limited knowledge of the probability distribution. We also experience the worth of knowing the exact form of the distribution in terms of computational speed. We present a heuristic of these problems in the Subsection 2.2.

2.1 The Exact Algorithm

The exact algorithm that we consider for solving both T-KNAP($\mathbf{P}, \mathbf{C}, B(F)$) and R-KNAP($\mathbf{P}, \mathbf{C}, B(F)$) is a depth first branch and bound algorithm (DFBB). The pseudocode for the general DFBB procedure is provided in Figure 1. There are two problem-specific functions in the procedure, CalculateObjective($\cdot$) and CalculateBound($\cdot$). The CalculateObjective($\cdot$) function is easily implemented using expression (2) for T-KNAP($\mathbf{P}, \mathbf{C}, B(F)$) and expression (5) for P-KNAP($\mathbf{P}, \mathbf{C}, B(F)$) respectively. Therefore the remainder of this subsection is dedicated to the definition of the CalculateBound($\cdot$) function for both the problems.

We will use the following notation in this subsection. A vector $\tilde{\mathbf{x}} = (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_a)$, $a < n$, $\tilde{\mathbf{x}} \in \{0,1\}^a$ is referred to as a ***partial solution***. Any vector $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$ such that $x_i = \tilde{x}_i$ for $i = 1, \ldots, a$ is called a ***realization*** of $\tilde{\mathbf{x}}$. We denote by ρ_j the ratio $\frac{p_j}{c_j}$ for $j = 1, \ldots, n$, assume that $\rho_1 \geq \rho_2 \geq \cdots \geq \rho_n$, and define $\rho_{n+1} = 0$. We also denote $\pi_a = \sum_{j=1}^{a} p_j\tilde{x}_j$, $b_a = \sum_{j=1}^{a} p_j\tilde{x}_j$, $\pi_m = \sum_{j=1}^{a} p_j\tilde{x}_j + \sum_{j=a+1}^{m} c_j$, and $b_m = \sum_{j=1}^{a} c_j\tilde{x}_j + \sum_{j=a+1}^{m} c_j$, for $a+1 \leq m \leq n$. Notice that π_a, π_m, b_a, and b_m are all functions of $\tilde{\mathbf{x}}$.

Computing CalculateBound($\tilde{\mathbf{x}}$) for T-KNAP($\mathbf{P}, \mathbf{C}, B(F)$): We will find an upper bound for $Z_T(\mathbf{x})$ for all realizations $\mathbf{x}$ of $\tilde{\mathbf{x}}$. Let $\mathbf{x}$ be a realization of $\tilde{\mathbf{x}}$ with cost b in the interval $[b_m, b_{m+1}]$. The objective function of a linear relaxation of KNAP($\mathbf{P}, \mathbf{C}, b$) (see Martello and Toth [10]), when $b \in [b_m, b_{m+1}]$, is:

$$\Pi_l(b) = \pi_m + \rho_{m+1}(b - b_m) = \lambda_{1,m}b + \lambda_{2,m}, \tag{13}$$

where $\lambda_{1,m} = \rho_{m+1}$ and $\lambda_{2,m} = \pi_m - \lambda_{1,m}b_m$. Therefore an upper bound for the maximum value of $Z_T(\mathbf{x})$ for all realizations $\mathbf{x}$ of $\tilde{\mathbf{x}}$ with $\mathbf{Cx} = b \in [b_m, b_{m+1}]$ is given by $(\lambda_{1,m}b + \lambda_{2,m})F(b)$, and the overall upper bound for all weakly feasible solutions is given by

$$\max_{b \leq B_U}\Big[\Big\{(\pi_m - \rho_{m+1}b_m) + \rho_{m+1}b\Big\} \cdot (\lambda_{1,m}b + \lambda_{2,m})F(b)\Big] = \max_{a \leq m \leq r}\{\Psi(m)\}, \tag{14}$$

Note:
n is the size of the problem, i.e the cardinality of $\mathbb{E}$;
BestSoFar *stores the best solution found, initialized to* $\emptyset$;
BestPsi *stores the objective function value of* BestSoFar, *initialized to* $-\infty$;

CalculateObjective(x) *calculates the objective function value of a given solution* **x**.
CalculateBound(x) *calculates an upper bound to the objective function value achievable from a given partial solution* **x**.

```
procedure DFBB
Parameters
    x : Solution;
    a : Index up to which values have been assigned in x;

begin
    if (a = n) then
    begin
        if CalculateObjective(x) > BestPsi) then
        begin
            BestSoFar := x;
            BestPsi := CalculateObjective(x);
        end;
    end;
    x_{a+1} := 1;
    if (Px ≤ B_U) and (CalculateBound(x) > BestPsi)
        BranchAndBound(x, a + 1);
    x_{a+1} := 0;
    if (CalculateBound(x) > BestPsi)
        BranchAndBound(x, a + 1);
end;
```

Fig. 1: Pseudocode for a depth first branch and bound algorithm

where $r = \max\{k : b_k \leq B_U\}$ and

$$\Psi(m) = \max_{b_m \leq b \leq b_{m+1}} (\lambda_{1,m} b + \lambda_{2,m}) F(b). \tag{15}$$

Notice that $\lambda_{2,m}$ is constant over the interval $[b_m, b_{m+1}]$. If one chooses to restrict oneself to mean or median feasible solutions, only the B_U in the definition of r need to be suitably replaced.

Upper bounds to $\Psi(m)$ can be calculated fast if we know in advance that $F(b)$ is either convex or concave over a given interval. The shape of $F(b)$ is known for all distributions. For instance, $F(b)$ is linear for Uniform distributions, while for Normal distributions, $F(b)$ is concave over $[B_L, \mu]$ and convex over $[\mu, B_U]$, where μ is the mean of the distribution.

Let us first consider the case when $F(b)$ is convex over $[b_m, b_{m+1}]$. Refer to Figure 2. The dashed line shows a linear approximation $F_l(b)$ of $F(b)$ such that $F_l(b) \geq F(b)$ in the interval of interest. Using elementary geometry we can represent $F_l(b)$ with the equation

$$F_l(b) = \lambda_{3,m} b + \lambda_{4,m} \tag{16}$$

where $\lambda_{3,m} = \frac{F(b_{m+1}) - F(b_m)}{b_{m+1} - b_m}$ and $\lambda_{4,m} = F(b_m) - \lambda_{3,m} b_m$.

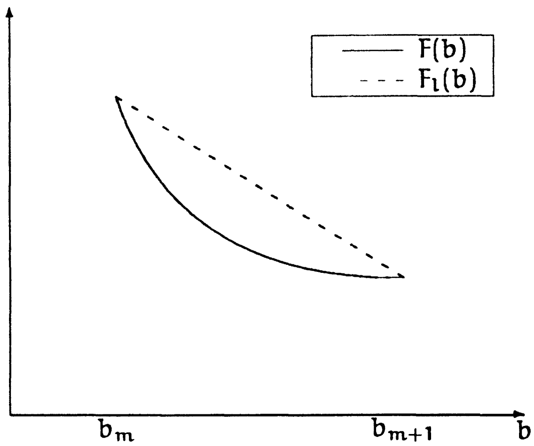

Fig. 2: Linear approximation (upper bound) of F(b) when F(b) is convex

The maxima of the function $\Pi_l(b) \cdot F_l(b)$ occurs at $b_m^\star = -\frac{\lambda_{1,m}\lambda_{4,m}+\lambda_{2,m}\lambda_{3,m}}{2\lambda_{1,m}\lambda_{3,m}}$. So an upper bound $\Psi_l(m)$ of $\Psi(m)$ is

$$\Psi_l(m) = \begin{cases} \pi_m F(b_m) & \text{if } b_m^\star \leq b_m \\ -\frac{(\lambda_{1,m}\lambda_{4,m}-\lambda_{2,m}\lambda_{3,m})^2}{4\lambda_{1,m}\lambda_{3,m}} & \text{if } b_m < b_m^\star < b_{m+1} \\ \pi_{m+1} F(b_{m+1}) & \text{if } b_m^\star \geq b_{m+1} \end{cases} \quad (17)$$

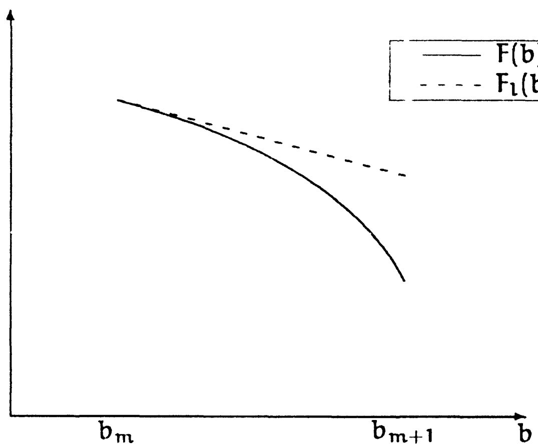

Fig. 3: Linear approximation (upper bound) of F(b) when F(b) is concave

Next, let us assume that F(b) is concave over $[b_m, b_{m+1}]$. Refer to Figure 3. The dashed line shows a linear approximation $F_l(b)$ of F(b) such that $F_l(b) \geq F(b)$ for $b_m \leq b \leq b_{m+1}$. $F_l(b)$ is tangential to F(b) at $b = b_m$. So its equation can be written as

$$F_l(b) = \lambda_{5,m} b + \lambda_{6,m}, \quad (18)$$

where $\lambda_{5,m} = f(b_m)$ and $\lambda_{6,m} = F(b_m) - \lambda_{5,m} b_m$.

Using arguments similar to those used for developing (17) we get an upper bound $\Psi_l(m)$ for $\Psi(m)$ when $b_m \leq b \leq b_{m+1}$ given by

$$\Psi_l(m) = \begin{cases} \pi_m F(b_m) & \text{if } b_m^\dagger \leq b_m \\ -\frac{(\lambda_{1,m}\lambda_{6,m} - \lambda_{2,m}\lambda_{5,m})^2}{4\lambda_{1,m}\lambda_{5,m}} & \text{if } b_m < b_m^\dagger < b_{m+1} \\ \Pi_{m+1}(\lambda_{5,m} b_{m+1} + \lambda_{6,m}) & \text{if } b_m^\dagger \geq b_{m+1} \end{cases} \tag{19}$$

where

$$b_m^\dagger = -\frac{\lambda_{1,m}\lambda_{6,m} + \lambda_{2,m}\lambda_{5,m}}{2\lambda_{1,m}\lambda_{5,m}}.$$

A pseudocode for the CalculateBound$(\cdot)$ function for T-KNAP$(\mathbf{P}, \mathbf{C}, B(F))$ using expressions (17) and (19) is given in Figure 4.

```
function CalculateBound
Parameter
      x : Partial Solution
begin
      r := max{k : b_k ≤ B_u};
      for i = a to r do
      begin
            if F(b) is convex over [b_i, b_{i+1}] then
                  use (17) to calculate Ψ_l(i);
            else if F(b) is concave over [b_i, b_{i+1}] then
                  use (19) to calculate Ψ_l(i);
            else
                  use (15) to calculate Ψ_l(i) = Ψ(i);
      end;
      return max_{a≤i≤r} Ψ_l(i);
end;
```

Fig. 4: CalculateBound$(\cdot)$ function for T-KNAP$(\mathbf{P}, b\mathbf{C}, B(F))$

Computing CalculateBound$(\tilde{x})$ for P-KNAP$(\mathbf{P}, \mathbf{C}, B(F))$: We will now find an upper bound for $Z_P(\mathbf{x})$ for all realizations $\mathbf{x}$ of $\tilde{\mathbf{x}}$. Let $\mathbf{x}$ be a realization of $\tilde{\mathbf{x}}$ with cost b in the interval $[b_m, b_{m+1}]$. From the linear relaxation of KNAP$(\mathbf{P}, \mathbf{C}, b)$

$$\mathbf{Px} \leq \pi_m + \rho_{m+1}(b - b_m) \leq \pi_{m+1}.$$

Also since $\mu(\cdot)$ is non-decreasing and $F(\cdot)$ is non-increasing,

$$\mu(\mathbf{Cx}) - \mathbf{Cx}[1 - F(\mathbf{Cx})] \leq \begin{cases} \mu(b_{m+1}) - b_m[1 - F(B_L)] = \mu(b_{m+1}) & \text{if } b_m < B_L \leq b_{m+1} \\ \mu(b_{m+1}) - b_m[1 - F(b_m)] & \text{if } B_L < b_m. \end{cases}$$

Therefore an upper bound for the maximum value of $Z_P(\mathbf{x})$, for all realizations $\mathbf{x}$ of $\tilde{\mathbf{x}}$ is given by

$$\max_{m \leq r}\{\Psi(m)\}, \tag{20}$$

where

$$\Psi(m) = \begin{cases} \pi_{m+1} & \text{if } b_{m+1} < B_L \\ \pi_{m+1} + \theta\mu(b_{m+1}) & \text{if } b_m < B_L \leq b_{m+1} \\ \pi_{m+1} + \theta\{\mu(b_{m+1}) - b_m[1 - F(b_m)]\} & B_L < b_m. \end{cases} \tag{21}$$

and $r = \max\{k : b_k \leq B_U\}$.

From the discussion above we can construct the function CalculateBound(**x**) for P-KNAP(**P**, **C**, B(F)). The pseudocode for this function is given in Figure 5.

```
function CalculateBound
Parameter
      x : Partial Solution
begin
      if Cx < B_L then
            l := max{k : b_k < B_L},
      else
            l ← a;
      r = max{k : b_k ≤ B_U};
      return max_{l≤i≤r} Ψ_m (from (21));
end,
```

Fig. 5: CalculateBound(·) function for P-KNAP(**P**, **C**, B(F))

Notice that the bound calculation for P-KNAP(**P**, **C**, B(F)) described in this section does not use any special structure of the survival function $F(\cdot)$. Therefore we would not expect it to output good upper bounds. In practice, a better bound would be obtained by searching for the maximum value of $\pi_m + \rho_{m+1}(b - b_m) + \mu(b) - b[1 - F(b)]$ in each of the relevant intervals $[b_m, b_{m+1}]$. This is illustrated in the examples below.

Example 4: Suppose that B is uniformly distributed on $[B_L, B_U]$. Then using expression (8), an upper bound for $Z_P(\cdot)$ for any realization of $\tilde{x}$ with a cost in the interval $[b_m, b_{m+1}]$, $B_L < b_m$ is given by

$$\pi_m + \rho_{m+1}(b - b_m) - \theta\frac{(b - B_L)^2}{2(B_U - B_L)}$$

This expression is concave in b and reaches a maximum value at

$$b = b^\star_m = B_L + \rho_{m+1}\frac{B_U - B_L}{\theta}.$$

Thus, for the uniform distribution, we can modify the bound $\Psi(m)$ to

$$\Psi(m) = \begin{cases} \pi_{m+1} & \text{if } b_{m+1} < B_L \\ \pi_{m+1} + \theta\mu(b_{m+1}) & \text{if } b_m < B_L \leq b_{m+1} \\ \pi_m + \theta\{\mu(b_m) - b_m[1 - F(b_m)]\} & B_L < b_m \text{ and } b^\star_m \leq b_m \\ \pi_m + \rho_{m+1}(b^\star - b_m) + \theta\{\mu(b^\star_m) - b^\star_m[1 - F(b^\star_m)]\} & B_L < b_m \text{ and } b_m < b^\star_m < b_{m+1} \\ \pi_{m+1} + \theta\{\mu(b_{m+1}) - b_{m+1}[1 - F(b_{m+1})]\} & B_L < b_m \text{ and } b^\star_m \geq b_{m+1}, \end{cases} \tag{22}$$

using $F(\cdot)$ and $\mu(\cdot)$ as defined in Example 1.

Example 5: Suppose that B follows a shifted exponential distribution supported on $[B_L, \infty]$ with parameter λ. Then using expression (12), an upper bound for $Z_P(\cdot)$ for any realization of $\tilde{x}$ with a cost in the interval $[b_m, b_{m+1}]$, $B_L < b_m$ is given by

$$\pi_m + \rho_{m+1}(b - b_m) - \theta\mathbf{C}\mathbf{x}\Big[1 + 2\exp[-\lambda(b - B_L)]\Big] + \frac{\theta}{\lambda}\Big[1 - \exp[-\lambda(b - B_L)]\Big] + \theta B_L$$

This expression reaches a maximum value at

$$b = b_m^\star = -\frac{1}{2\lambda}\Big(2\,\mathrm{LambertW}\big(\frac{\theta-\rho_{m+1}}{2\theta}\exp[\frac{1}{2}-\lambda B_L]\big)-1\Big),$$

where $\mathrm{LambertW}(x)$ satisfies $\mathrm{LambertW}(x)\cdot\exp[\mathrm{LambertW}(x)] = x$.

Thus, for the shifted exponential distribution, we can modify the bound $\Psi(m)$ to

$$\Psi(m) = \begin{cases} \pi_{m+1} & \text{if } b_{m+1} < B_L \\ \pi_{m+1} + \theta\mu(b_{m+1}) & \text{if } b_m < B_L \leq b_{m+1} \\ \pi_m + \theta\{\mu(b_m) - b_m[1-F(b_m)]\} & B_L < b_m \text{ and } b_m^\star \leq b_m \\ \pi_m + \rho_{m+1}(b^\star - b_m) + \theta\{\mu(b_m^\star) - b_m^\star[1-F(b_m^\star)]\} & B_L < b_m \text{ and } b_m < b_m^\star < b_{m+1} \\ \pi_{m+1} + \theta\{\mu(b_{m+1}) - b_{m+1}[1-F(b_{m+1})]\} & B_L < b_m \text{ and } b_m^\star \geq b_{m+1}, \end{cases} \tag{23}$$

using $F(\cdot)$ and $\mu(\cdot)$ as defined in Example 3.

2.2 The Heuristic

The heuristic developed for both T-KNAP(**P**, **C**, B(F)) and P-KNAP(**P**, **C**, B(F)) is based on local search using a 2-swap neighborhood structure. The initial solution for local search was obtained following a greedy procedure.

The greedy procedure considers the elements of **E** one by one in a non-increasing order of profit to cost ratios and adds them to the knapsack if such an addition improved the objective function value (i.e. $Z_T(\cdot)$ in the case of T-KNAP(**P**, **C**, B(F)) and $Z_P(\cdot)$ in the case of P-KNAP(**P**, **C**, B(F))). It stops when all the elements in **E** have been considered.

Once the greedy solution is obtained, the local search procedure is started using the greedy solution as the current solution. The local search procedure performs iterations until a stopping condition is reached, after which it outputs the current solution at that stage and terminates. In each iterations, all 2-swap neighbors of the current solution, i.e. solutions obtained by either throwing out one of the elements in the current solution, or adding one element into the current solution, or both, are examined to see if any of them have a better objective function value than the current solution (i.e. are better neighbors). If a better neighbor is found, then the neighbor with the best objective function value is denoted the current solution and the iteration is over. If no neighbor is better than the current solution, then the stopping condition is said to have been reached. A pseudocode of our local search heuristic is presented in Figure 6.

In the next section, we report the results of computations using the algorithms developed in this section.

3 Computational Experience

We performed some computational experiments to evaluate the performance of the algorithms developed in the previous section. We report our observations here. It should be noted that these observations are preliminary in nature, and need to be validated by more extensive computational experiments.

We generated ten problems, two each of sizes 15, 20, 25, 40, and 60. The cost values (c_j's) for each of the problems were chosen from a discrete uniform distribution supported on $\{1, 2, \ldots, 100\}$. In one set of problems, consisting of one problem of each size, the profit values (p_j's) were generated

Note:
The elements of E *are assumed to be ordered in non-increasing order of* $\frac{p_j}{c_j}$ *ratios;*
n *is the size of the problem, i.e the cardinality of* E*;*

CalculateObjective(x) *calculates the objective function value of a given solution* **x**.

```
procedure DFBB
begin
    /* Greedy procedure */
    x_greedy := ∅;
    for i:= 1 to n do
    begin
        x := x_greedy ∪ {e_j};
        if (CalculateObjective(x) > CalculateObjective(x_greedy) then
            x_greedy := x;
    end;

    /* Local Search */
    x_current := x_greedy;
    StoppingCondition := FALSE;
    while (StoppingCondition = FALSE) do
    begin
        x_best := the best 2-swap neighbor of x_current;
        if (CalculateObjective(x_best) ≥ CalculateObjective(x_current) then
            x_current := x_best;
        else
            StoppingCondition := TRUE;
    end;
    return x_current;
end;
```

Fig. 6: Pseudocode for a local search heuristic

independently from a discrete uniform distribution supported on $\{1, 2, \ldots, 100\}$. The problems in this set were called the uncorrelated problems. The $\frac{p_j}{c_j}$ ratios in these problems were observed to vary between 0.02 and 50.0. In the remaining problems, referred to as strongly correlated problems, the profit values were chosen so that the $\frac{p_j}{c_j}$ ratios were from a uniform distribution supported on $[0.9, 1.1]$. We label each of the problems using the nomenclature "xy" where x was "u" or "s" depending on whether the problem was uncorrelated or strongly correlated, and y denoted the problem size. For example, the problem "u25" refers to the uncorrelated problem with 25 elements.

In our computation, we considered two probability distributions to model the randomness of the budget B, namely the Uniform and the Normal distribution. For the ease of comparison, the effective supports of these distributions were taken to be $[B_L = b_l \sum_j c_j,\ B_U = b_u \sum_j c_j]$. In the case of the normal distribution we chose the mean (μ) and the variance (σ^2) such that $\mu \pm 3\sigma = (B_L,\ B_U)$.) For each problem, we experimented with three sets of $[b_l, b_u]$ values, viz. [0.2, 0.8], [0.3, 0.7], and [0.4, 0.6], which were chosen to ensure that the means of B remained the same for all the problems.

The computations were conducted on a Pentium 200 MHz computer running the Linux operating system. The maximum time allowed to solve a problem was set to 500 CPU seconds. If a run did not complete in the time allotted, then the corresponding entry is marked with a '—' in our tables. The CalculateBound(·) function for T-KNAP problems was implemented using expression (17) when

B was uniformly distributed. For normally distributed B, expression (19) was used in the interval $[B_L, \frac{B_U+B_L}{2}]$ and expression (17) in the interval $(\frac{B_U+B_L}{2}, B_U]$. Preliminary experimentation with the CalculateBound(·) function for P-KNAP problems showed that expression (21) did lead to upper bounds of extremely poor quality. Therefore when B was uniformly distributed, expression (22) was used. When B was normally distributed, the bound was obtained by using a search algorithm to find the position of the maximum value of the relaxation of $Z_P(\cdot)$ in the various intervals.

Characteristics of the Optimal Solution: The profit sums, costs, and the respective objective function values of an optimal solution $\mathbf{x}^*$ for T-KNAP problems and P-KNAP problems are presented in Tables 1 and 2. Note that the optimal solution, in either approach, need not be unique, and there may exist other optimal solutions with different profit and cost sums. In this part the optimal solution that we refer to was generated by our algorithms described in the Subsection 2.1.

The costs of the optimal solutions were observed to be closer to B_L in all cases than to B_U. This closeness (to B_L) was measured by computing the expression $\frac{C^*-B_L}{B_U-B_L}$ where C^* was the cost of our optimal solution. For P-KNAP problems, increasing the θ value amounts to increasing the penalty for infeasibility, and hence the costs of the optimal solutions were closer to B_L at higher θ values than at lower θ values as seen in Table 2. Disregarding profit accrued from an infeasible solution is a way of (severely) penalizing infeasibility; hence the costs of the optimal solutions to T-KNAP problems were even closer to B_L. The closeness was also more pronounced for uniformly distributed B than for normally distributed B, since the heavier left tail of the uniform distribution imposed a stronger penalty for exceeding the budget. Similar behaviour is expected for all distributions with a equal or heavier left tail. Not surprisingly, in a few of these problems, especially when the range of the Uniform distribution was taken to be relatively small, the optimal solution was observed to be even strongly feasible. Another interesting observation regarding the measure of closeness of optimal solutions to B_L values for P-KNAP problems was that it was not affected by the size of the problems.

The objective values of the optimal solutions were seen to increase with a decrease in the length of the interval $[b_l, b_u]$. This is actually a direct effect of an increase in b_l, which by our choice is associated with the reduction in length of the $[b_l, b_u]$ intervals.

Note that for all the problems that we considered, the optimal solutions output, for T-KNAP as well as P-KNAP problems, were mean feasible (and consequently median feasible, since both the uniform and the normal distributions are symmetric). Hence, we could have restricted ourselves to solutions in $\mathcal{F}_\mu$ or $\mathcal{F}_{(0.5)}$ for T-KNAP problems, as well as for P-KNAP problems with at least moderately large θ values. Such a restriction would have made the calculation of bounds faster. We believe that this observation would be valid for many common distributions encountered in real-life problems.

Performance of the Exact Algorithm: Table 3 presents the number of nodes expanded by the DFBB algorithm and its execution time in CPU seconds for the case in which the budget B was uniformly distributed. Table 4 presents the same observations for the case in which the budget B followed a normal distribution.

In both tables we observe that the time taken to solve strongly correlated problems was much higher than the time taken to solve an uncorrelated problem of the same size. This observation is in line with similar observations for deterministic 0-1 knapsack problems (refer Martello and Toth [10]).

For P-KNAP problems, in general, the number of nodes expanded and the execution time of the DFBB algorithm increased with increasing θ value. We also saw that the execution time and the number of nodes expanded in the DFBB tree increased with increasing length of the $[b_l, b_u]$ interval in case of T-KNAP problems but decreased in case of P-KNAP problems. Exceptions to this trend were noticed in strongly correlated T-KNAP problems in which B was uniformly distributed and in P-KNAP problems with $\theta = 10$ and normally distributed B.

The time needed to expand a node was much higher for P-KNAP problems in which B was normally distributed. This was because, in these problems, the calculation of upper bounds involved a search procedure, which took more time. The bounds found in this manner are however seen to be very effective, since the number of nodes expanded in these problems were much lower than the number of nodes expanded in similar size problems in which B followed a uniform distribution supported on the same interval.

Performance of the Heuristic: The local search heuristic developed in the previous section performed very well, both in terms of solution quality as well as execution times. Our computational experience with this heuristic is summarized in Tables 5 and 6. Notice that it took less than 0.05 CPU seconds on each of the T-KNAP problems, and less than 0.15 CPU seconds on each of the P-KNAP problems. The average suboptimality was less than 0.07% for T-KNAP problems and 0.048% for R-KNAP problems. Local search performed better when B was normally distributed than when it was uniformly distributed.

4 Summary and Directions of Future Research

In this paper we consider a static stochastic 0-1 knapsack problem in which the budget is random. The relevant literature is very briefly surveyed in the introductory section. In Section 1, we formally define the knapsack problems that we study here. We extend the concept of feasibility of a solution for deterministic knapsack problems, to define strongly feasible solutions that are feasible for all possible values that the budget may assume, and weakly feasible solutions that are feasible only for a range of the possible values of the budget. We also define alternative concepts of solution feasibility like mean feasibility and feasible with a reliability coefficient. We revise the expression of the objective function value of the deterministic knapsack problem to incorporate two different methods of penalizing infeasibilities. We show that maximizing the expected value of the objective function is equivalent to minimizing the expected value of the regret associated with a solution to the static stochastic knapsack problem under very reasonable assumptions. We conclude the section by defining two problems, T-KNAP and P-KNAP, based on two different ways of handling infeasibilities.

In Section 2 we devise an exact algorithm and a heuristic to solve T-KNAP and P-KNAP problems. The exact algorithm is based on depth first branch and bound (DFBB), and the heuristic is based on local search, starting with a greedy solution. Most of this section is devoted to methods for computing upper bounds for the DFBB algorithm. We do not need to use the functional form of the survival function to derive these bounds; consequently the bounds are useful even when the exact functional form of the survival function is unknown. While the computed bounds (consequently the algorithm) are meant explicitly for weakly feasible solutions, they can be improved (in terms of computational speed) for smaller classes of feasible solutions (like $\mathcal{F}_\mu$ or $\mathcal{F}_{(\alpha)}$), by adjusting the value of r in (14) and (21) appropriately.

Section 3 contains the results of preliminary computations with T-KNAP and P-KNAP problems. We see that the costs of the optimal solutions to both T-KNAP and P-KNAP problems are almost always very close to the lowest possible value of the budget. This shows that considering mean or median feasible solutions instead of weakly feasible solutions does not affect the quality of the output. We also see that the time taken by the exact algorithm almost always increases when the extent of penalization of infeasibilities in P-KNAP problems increases. The execution times increase for T-KNAP problems when the length of the support for the distribution of the budget increases, but decrease in case of P-KNAP problems. The local search heuristic is seen to output solutions with objectives within 2% of that of the optimal solution within 0.15 CPU seconds.

We believe that there is need for much more elaborate computational experiments with T-KNAP and P-KNAP problems. The results that we report in Section 3 are based on ten problems. These results need to be verified for a much larger data set — containing problems in which the profit and cost values come from distributions other than uniform, and ones in which the budget follows distributions other than uniform and normal. These results can be used, for example, to examine whether our observation regarding the exceptional hardness of strongly correlated problems in which the budget follows a uniform distribution with $b_u = 0.3$ and $b_l = 0.7$, and the easy solvability of P-KNAP problems in which the budget follows a normal distribution with $b_u = 0.3$ and $b_l = 0.7$ is really valid, or whether it appears so due to our small sample size of observations.

Note that both T-KNAP and P-KNAP problems penalize infeasibilities. An interesting question is whether there exists a θ value for P-KNAP problems, for which an optimal solution to the T-KNAP problem is also an optimal solution to the P-KNAP problem? Intuitively, it seems that such a θ value should exist, since for very low valus of θ, the costs of optimal solutions to P-KNAP problems should be very close to $x^*_{B_u}$, the maximum profit solution to KNAP($\mathbf{P}, \mathbf{C}, B_U$), while for very high valus of θ, the optimal solutions to P-KNAP problems are strongly feasible. Assuming that such a θ value exists, questions regarding how it depends on the size of the problem, the distribution of the profit and cost variables, and the randomness of the budget arise. We could not answer these questions with our limited experimentation, but we think that more elaborate computational experiments would help.

Another possible direction of future research in these problems is in the area of algorithm development. In this paper, we have used a DFBB algorithm with rather simplistic bounds. More complex, and possibly distribution specific bounds can markedly reduce the execution times for exact algorithms. Development of such bounds promises to be interesting. Apart from development of new bounds for DFBB, one can also look for ways to adapt other specialized algorithms for deterministic knapsack problems to solve the stochastic version of the problem. Another possible avenue of research in algorithm development could be to develop algorithms to solve dynamic stochastic knapsack problems with random budgets.

A third direction of research would be to analyze these problems under non-linear penalty assumptions. One could also analyze related problems, like the integer knapsack problem, and the bin-packing problem, under similar budget or capacity variations. It follows from Ghosh and Das [6] that if the profit vector, in addition to the budget, is random, then the optimization process is unaffected if the random elements in P are replaced by their expected values. In future, we plan to generalize the study by Cohn and Barnhart [5] and analyze a more general class of 0-1 knapsack problems with random cost elements.

References

[1] I.L. Averbakh. "On an algorithm for solving the m-dimensional knapsack problem with random coefficients" *Diskretnaya Mathematica 2*, (1990) 3–9.

[2] I.L. Averbakh. "Minimax regret solutions for minmax optimization problems with uncertainty" *Operations Reseach Letters 27*, (2000) 57–65.

[3] J.R. Birge and F. Louveaux. *Introduction to Stochastic Programming.* Springer Series in Operations Research, Springer, (1997).

[4] R.L. Carraway, R.L. Schmidt, and L.R. Weatherford. "An algorithm for maximizing target achievement in the stochastic knapsack problem with normal returns" *Naval Research Logistics 40*, (1993), 161–173.

[5] A.M. Cohn and C. Barnhart. "The stochastic knapsack problem with random weights: A heuristic approach to robust transportation planning" Triennial Symposium on Transportation Analysis (TRISTAN III), San Juan, Puerto Rico, June 17-23, (1998).

[6] D. Ghosh and S. Das. "Discrete optimization problems with random cost coefficients" *Report # 00A33, SOM Graduate School/Research Institute, University of Groningen*, (2000).

[7] M.I. Henig. "Risk criteria in a stochastic knapsack problem" *Operations Research 38*, (1990), 820–835.

[8] A.J. Kleywegt and J.D. Papastavrou. "The dynamic and stochastic knapsack problem" *Operations Research 46*, (1998), 17–35.

[9] A. Marchetti-Spaccamela and C. Vercellis. "Stochastic on-line knapsack problems" *Mathematical Programming 68 (Series A)*, (1995), 73–104.

[10] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Complexity.* Kluwer Academic Publishers, (1990).

[11] M. Sniedovich. "Preference order stochastic knapsack problems: Methodological issues" *Journal of the Operational Research Society 31*, (1980), 1025–1032.

[12] M. Sniedovich. "Some comments on preference order dynamic programming models" *Journal of Mathematical analysis and applications 79*, (1981) 489–501.

[13] E. Steinberg and M.S. Park. "A preference order dynamic program for a knapsack problem with stochastic rewards" *Journal of the Operational Research Society 30*, (1979), 141–147.

[14] K. Szkatuła. "The growth of multi-constraint random knapsacks with large right-hand sides of the constraints" *Operations Research Letters 21*, (1997), 25–30.

[15] M.H. an der Vlerk. *Stochastic Programming with Integer Recourse.* Ph.D. thesis, University of Groningen, The Netherlands, (1995).

[16] G. Yu. "On the max-min 0-1 knapsack problem with robust optimization applications" *Operations Research 44*, (1996), 407–415.

Problem	$[b_l, b_u]$	B_L	B_U	T-KNAP			P-KNAP ($\theta = 10$)			P-KNAP ($\theta = 50$)			P-KNAP ($\theta = 100$)		
				Px*	**Cx***	Objective	**Px***	**Cx***	Objective	**Px***	**Cx***	Objective	**Px***	**Cx***	Objective
u15	[0.2,0.8]	142.4	569.6	511	177	469.61	511	177	496.99	452	149	449.45	452	149	446.9
	[0.3,0.7]	213.6	498.4	575	228	545.93	575	228	571.36	575	228	556.8	543	215	542.66
	[0.4,0.6]	284.8	427.2	607	266	607.00	627	293	624.64	627	293	615.2	607	266	607.00
u20	[0.2,0.8]	156.6	626.4	607	159	603.9	662	202	640.06	612	164	609.09	607	159	606.39
	[0.3,0.7]	234.9	548.1	706	245	683.23	723	266	707.56	706	245	697.86	706	245	689.71
	[0.4,0.6]	313.2	469.8	767	309	767.00	777	324	773.28	767	309	767.00	767	309	767.00
u25	[0.2,0.8]	234.6	938.4	892	291	820.52	892	291	869.4	838	258	818.55	815	232	815.00
	[0.3,0.7]	351.9	821.1	965	355	958.62	1025	416	981.21	965	355	964.49	965	355	963.98
	[0.4,0.6]	469.2	703.8	1060	465	1060.00	1083	491	1072.87	1060	465	1060.00	1060	465	1060.00
u40	[0.2,0.8]	390.4	1561.6	1244	542	1082.98	1244	542	1145.88	1086	423	1063.31	—	—	—
	[0.3,0.7]	585.6	1366.4	1278	586	1277.35	1343	649	1317.26	—	—	—	—	—	—
	[0.4,0.6]	780.8	1171.2	1449	776	1449.00	—	—	—	—	—	—	—	—	—
u60	[0.2,0.8]	579.6	2318.4	1919	643	1849.03	2064	772	1957.55	—	—	—	—	—	—
	[0.3,0.7]	869.4	2028.6	2149	866	2149.00	—	—	—	—	—	—	—	—	—
	[0.4,0.6]	1159.2	1738.8	2419	1150	2419.00	—	—	—	—	—	—	—	—	—
s15	[0.2,0.8]	148.0	592.0	304	285	210.20	218	202	185.16	167	154	164.97	167	154	162.95
	[0.3,0.7]	222.0	518.0	269	251	242.65	269	215	254.79	240	224	239.66	240	224	239.32
	[0.4,0.6]	296.0	444.0	313	294	313.00	334	314	323.05	321	302	314.92	314	296	314.00
s20	[0.2,0.8]	217.0	868.0	451	429	304.13	300	281	268.54	249	231	241.47	241	223	238.23
	[0.3,0.7]	325.5	759.5	390	370	350.01	388	368	367.19	353	333	349.76	347	327	346.74
	[0.4,0.6]	434.0	651.0	457	435	454.89	477	456	465.85	457	435	456.89	457	435	456.77
s25	[0.2,0.8]	242.2	968.8	502	467	346.69	342	316	303.52	279	257	271.46	270	248	267.69
	[0.3,0.7]	363.3	847.7	455	422	399.86	443	411	419.51	402	372	398.09	398	368	395.72
	[0.4,0.6]	484.4	726.6	518	483	518.00	545	509	532.51	523	489	520.82	522	487	520.60
s40	[0.2,0.8]	404.8	1619.2	829	789	566.73	571	537	499.04	—	—	—	—	—	—
	[0.3,0.7]	607.2	1416.8	727	689	653.55	—	—	—	—	—	—	—	—	—
	[0.4,0.6]	809.6	1214.4	850	810	849.16	—	—	—	—	—	—	—	—	—
s60	[0.2,0.8]	540.0	2160.0	1147	1077	766.79	757	702	676.00	—	—	—	—	—	—
	[0.3,0.7]	810.0	1890.0	981	916	884.72	—	—	—	—	—	—	—	—	—
	[0.4,0.6]	1080.0	1620.0	1151	1081	1148.87	—	—	—	—	—	—	—	—	—

Tab. 1: Characteristics of the Optimal Solution. Uniform Distribution

Problem	$[b_l, b_u]$	B_L	B_U	T-KNAP Px*	T-KNAP Cx*	T-KNAP Objective	P-KNAP (θ = 10) Px*	P-KNAP (θ = 10) Cx*	P-KNAP (θ = 10) Objective	P-KNAP (θ = 50) Px*	P-KNAP (θ = 50) Cx*	P-KNAP (θ = 50) Objective	P-KNAP (θ = 100) Px*	P-KNAP (θ = 100) Cx*	P-KNAP (θ = 100) Objective
u15	[0.2,0.8]	142.4	569.6	575	228	554.22	607	266	572.55	575	228	523.97	511	177	497.97
	[0.3,0.7]	213.6	498.4	607	266	589.03	627	293	606.99	595	255	581.28	575	228	570.16
	[0.4,0.6]	284.8	427.2	627	293	624.44	636	320	629.46	627	293	625.59	627	293	624.18
u20	[0.2,0.8]	156.6	626.4	711	250	685.79	762	304	710.94	706	245	660.51	667	207	643.52
	[0.3,0.7]	234.9	548.1	762	304	725.72	767	309	754.42	733	281	717.45	723	266	709.02
	[0.4,0.6]	313.2	469.8	777	324	773.19	797	354	788.28	787	339	776.31	777	324	773.07
u25	[0.2,0.8]	234.6	938.4	1025	416	949.35	1025	416	987.40	965	355	912.49	915	317	873.41
	[0.3,0.7]	351.9	821.1	1048	442	1013.65	1083	491	1041.25	1025	416	1005.11	1025	416	985.23
	[0.4,0.6]	469.2	703.8	1083	491	1074.93	1102	522	1094.06	1083	491	1078.44	1083	491	1073.87
u40	[0.2,0.8]	390.4	1561.6	1393	703	1280.16	1393	703	1321.25	1264	569	1200.30	1244	542	1155.82
	[0.3,0.7]	585.6	1366.4	1425	741	1374.35	1470	793	1423.03	1393	703	1352.10	1357	675	1312.98
	[0.4,0.6]	780.8	1171.2	1502	831	1482.34	1540	882	1518.68	—	—	—	—	—	—
u60	[0.2,0.8]	579.6	2318.4	2277	993	2144.39	2355	1076	2221.83	2124	833	2039.74	2064	772	1972.29
	[0.3,0.7]	869.4	2028.6	2355	1076	2290.66	2419	1150	2369.24	2332	1050	2265.08	—	—	—
	[0.4,0.6]	1159.2	1738.8	2473	1230	2443.35	2523	1298	2498.91	—	—	—	—	—	—
s15	[0.2,0.8]	148.0	592.0	304	285	265.62	291	272	259.62	232	216	207.57	218	202	189.39
	[0.3,0.7]	222.0	518.0	326	306	293.93	326	306	303.83	291	272	269.5	269	251	256.16
	[0.4,0.6]	296.0	444.0	343	323	333.06	356	336	346.60	343	323	329.96	334	314	324.46
s20	[0.2,0.8]	217.0	868.0	451	429	383.98	423	402	373.99	337	317	300.88	306	287	273.34
	[0.3,0.7]	325.5	759.5	457	435	425.21	469	449	436.16	417	396	388.65	396	376	370.69
	[0.4,0.6]	434.0	651.0	495	474	480.28	516	494	501.18	488	467	476.29	479	458	467.56
s25	[0.2,0.8]	242.2	968.8	502	467	438.07	482	449	426.99	392	362	342.30	358	331	310.63
	[0.3,0.7]	363.3	847.7	528	493	484.77	539	503	500.43	471	438	443.83	455	422	424.05
	[0.4,0.6]	484.4	726.6	567	531	548.33	598	560	572.17	561	525	543.67	549	513	534.41
s40	[0.2,0.8]	404.8	1619.2	823	783	716.00	789	750	697.49	635	599	559.35	577	543	509.38
	[0.3,0.7]	607.2	1416.8	863	823	793.38	877	837	816.32	771	732	725.64	—	—	—
	[0.4,0.6]	809.6	1214.4	—	—	—	—	—	—	—	—	—	—	—	—
s60	[0.2,0.8]	540.0	2160.0	1114	1045	968.73	1077	1009	946.04	860	800	757.98	781	725	690.16
	[0.3,0.7]	810.0	1890.0	1168	1098	1073.68	1188	1117	1106.61	1042	976	981.97	—	—	—
	[0.4,0.6]	1080.0	1620.0	1260	1188	1214.73	1307	1234	1265.81	—	—	—	—		—

Tab. 2: Characteristics of the Optimal Solution. Normal Distribution

Problem	$[b_l, b_u]$	T-KNAP		P-KNAP ($\theta = 10$)		P-KNAP ($\theta = 50$)		P-KNAP ($\theta = 100$)	
		Nodes	Time	Nodes	Time	Nodes	Time	Nodes	Time
u15	[0.2,0.8]	67	<0.01	62	0.01	1269	0.04	2039	0.06
	[0.3,0.7]	53	<0.01	718	0.03	5547	0.17	6567	0.18
	[0.4,0.6]	33	<0.01	3833	0.11	11612	0.31	12103	0.32
u20	[0.2,0.8]	169	0.01	162	0.01	8627	0.33	19063	0.74
	[0.3,0.7]	109	0.01	2573	0.10	33008	1.16	41301	1.39
	[0.4,0.6]	62	0.01	83227	2.80	233950	7.51	265364	8.42
u25	[0.2,0.8]	161	0.01	556	0.03	59850	2.99	168840	7.97
	[0.3,0.7]	126	<0.01	6659	0.32	152176	6.26	695996	28.02
	[0.4,0.6]	89	<0.01	854320	33.61	6919235	257.93	8886362	327.35
u40	[0.2,0.8]	761	0.07	176	0.02	5200212	424.41	—	—
	[0.3,0.7]	657	0.05	863127	70.37	—	—	—	—
	[0.4,0.6]	406	0.03	—	—	—	—	—	—
u60	[0.2,0.8]	1381	0.18	82	0.01	—	—	—	—
	[0.3,0.7]	1097	0.12	—	—	—	—	—	—
	[0.4,0.6]	558	0.04	—	—	—	—	—	—
s15	[0.2,0.8]	127	0.01	676	0.02	1201	0.04	1226	0.03
	[0.3,0.7]	142	<0.01	3528	0.11	4202	0.12	4211	0.13
	[0.4,0.6]	169	<0.01	7583	0.21	7777	0.22	7805	0.21
s20	[0.2,0.8]	168	<0.01	2528	0.11	12567	0.49	16844	0.62
	[0.3,0.7]	2182	0.14	38864	1.44	79208	2.68	90505	3.14
	[0.4,0.6]	148	0.01	166893	5.53	190361	6.24	190361	6.28
s25	[0.2,0.8]	1078	0.07	5659	0.29	151410	6.55	185191	7.91
	[0.3,0.7]	2691	0.20	197615	8.56	1046595	43.02	1194176	48.39
	[0.4,0.6]	1120	0.05	2237358	87.86	3308797	129.00	3308794	129.36
s40	[0.2,0.8]	4227	0.08	848	0.11	—	—	—	—
	[0.3,0.7]	643354	78.29	—	—	—	—	—	—
	[0.4,0.6]	4381	0.27	—	—	—	—	—	—
s60	[0.2,0.8]	13165	1.73	9701	1.94	—	—	—	—
	[0.3,0.7]	314623	65.20	—	—	—	—	—	—
	[0.4,0.6]	12564	1.17	—	—	—	—	—	—

Tab. 3: Performance of the Exact Algorithm. Uniform Distribution

Problem	$[b_l, b_u]$	T-KNAP Nodes	T-KNAP Time	P-KNAP ($\theta = 10$) Nodes	P-KNAP ($\theta = 10$) Time	P-KNAP ($\theta = 50$) Nodes	P-KNAP ($\theta = 50$) Time	P-KNAP ($\theta = 100$) Nodes	P-KNAP ($\theta = 100$) Time
u15	[0.2,0.8]	56	0.01	30	0.04	39	0.06	87	0.19
	[0.3,0.7]	43	0.01	24	0.03	36	0.07	315	0.36
	[0.4,0.6]	24	<0.01	1043	0.79	2504	1.50	3759	2.05
u20	[0.2,0.8]	112	0.01	56	0.08	70	0.13	161	0.35
	[0.3,0.7]	110	0.01	110	0.20	908	1.32	1276	1.56
	[0.4,0.6]	55	<0.01	3789	3.71	29819	19.31	61986	33.35
u25	[0.2,0.8]	123	0.02	80	0.19	87	0.26	274	1.18
	[0.3,0.7]	94	<0.01	65	0.15	420	1.22	2692	6.51
	[0.4,0.6]	68	0.01	493	1.07	61640	75.23	433406	406.94
u40	[0.2,0.8]	619	0.08	378	1.00	358	1.15	536	2.97
	[0.3,0.7]	541	0.06	304	0.74	1308	7.64	94995	412.40
	[0.4,0.6]	312	0.03	1163	4.10	—	—	—	—
u60	[0.2,0.8]	964	0.18	552	2.60	527	2.94	539	3.39
	[0.3,0.7]	741	0.12	503	2.11	1634	15.07	—	—
	[0.4,0.6]	536	0.06	742	4.17	—	—	—	—
s15	[0.2,0.8]	995	0.01	67	0.07	110	0.16	537	0.49
	[0.3,0.7]	2892	0.24	63	0.05	2078	1.25	3063	1.74
	[0.4,0.6]	92	0.01	1435	0.78	6824	2.56	7373	2.69
s20	[0.2,0.8]	4223	0.54	112	0.17	180	0.53	1181	2.40
	[0.3,0.7]	14367	1.43	124	0.21	5232	5.86	21703	18.11
	[0.4,0.6]	2376	0.18	5621	4.45	90494	42.94	145536	68.27
s25	[0.2,0.8]	1265	0.15	743	1.09	536	0.87	601	1.32
	[0.3,0.7]	7160	0.93	675	0.90	3716	5.17	41489	36.33
	[0.4,0.6]	9123	0.83	985	1.32	460812	204.91	770976	304.98
s40	[0.2,0.8]	72532	18.88	3110	8.41	2182	5.99	2046	5.88
	[0.3,0.7]	1218899	245.81	2773	7.36	83253	279.27	—	—
	[0.4,0.6]	—	—	—	—	—	—	—	—
s60	[0.2,0.8]	33245	12.12	11169	40.27	7486	28.25	6593	25.36
	[0.3,0.7]	35705	11.06	9874	32.82	19040	136.72	—	—
	[0.4,0.6]	357683	73.08	7558	22.47	—	—	—	—

Tab 4: Performance of the Exact Algorithm. Normal Distribution

Problem	$[b_l, b_u]$	T-KNAP Objective	T-KNAP Time	P-KNAP ($\theta = 10$) Objective	P-KNAP ($\theta = 10$) Time	P-KNAP ($\theta = 50$) Objective	P-KNAP ($\theta = 50$) Time	P-KNAP ($\theta = 100$) Objective	P-KNAP ($\theta = 100$) Time
u15	[0.2,0.8]	469.61	<0.01	496.99	<0.01	449.45	0.01	446.9	0.01
	[0.3,0.7]	545.93	<0.01	571.36	<0.01	556.80	<0.01	542.66	0.01
	[0.4,0.6]	607.00	<0.01	624.64	<0.01	615.20	0.01	607.00	0.01
u20	[0.2,0.8]	603.90	0.01	640.06	<0.01	609.09	0.01	606.39	0.01
	[0.3,0.7]	683.23	<0.01	707.56	0.01	697.86	0.01	689.71	<0.01
	[0.4,0.6]	767.00	<0.01	773.28	0.01	767.00	0.01	767.00	0.01
u25	[0.2,0.8]	820.52	<0.01	869.40	<0.01	818.55	0.01	815.00	0.01
	[0.3,0.7]	958.62	<0.01	981.21	<0.01	964.49	<0.01	963.98	0.01
	[0.4,0.6]	1060.00	0.01	1072.87	<0.01	1060.00	0.01	1060.00	0.02
u40	[0.2,0.8]	1082.98	0.01	1145.88	0.01	1063.31	0.01	1049.10	0.01
	[0.3,0.7]	1275.00	0.01	1317.26	0.01	1290.34	0.02	1276.26	0.02
	[0.4,0.6]	1436.00	0.02	1470.32	0.02	1460.47	0.01	1451.08	0.02
u60	[0.2,0.8]	1849.03	0.02	1957.55	0.03	1866.68	0.03	1844.36	0.04
	[0.3,0.7]	2149.00	0.02	2212.59	0.03	2169.85	0.04	2160.70	0.03
	[0.4,0.6]	2419.00	0.02	2440.33	0.03	2422.79	0.06	2419.00	0.03
s15	[0.2,0.8]	210.20	<0.01	185.16	<0.01	164.97	<0.01	162.95	<0.01
	[0.3,0.7]	242.65	0.01	254.79	<0.01	239.66	0.01	237.32	0.01
	[0.4,0.6]	306.80	<0.01	323.05	<0.01	312.78	0.01	311.00	0.01
s20	[0.2,0.8]	304.13	<0.01	267.55	<0.01	241.47	0.01	238.23	0.01
	[0.3,0.7]	350.00	<0.01	367.19	0.01	349.76	0.01	346.74	0.01
	[0.4,0.6]	454.89	<0.01	465.42	0.01	456.89	0.01	456.77	<0.01
s25	[0.2,0.8]	346.69	<0.01	303.52	0.01	271.46	<0.01	267.67	0.01
	[0.3,0.7]	399.86	<0.01	419.43	<0.01	398.09	0.01	395.72	0.01
	[0.4,0.6]	518.00	<0.01	532.51	<0.01	520.82	0.01	519.63	0.01
s40	[0.2,0.8]	566.73	0.02	499.04	0.01	444.94	0.02	438.87	0.01
	[0.3,0.7]	653.45	0.01	684.68	0.05	651.28	0.03	647.22	0.01
	[0.4,0.6]	848.16	0.01	870.84	0.02	852.54	0.03	847.57	0.04
s60	[0.2,0.8]	766.79	0.02	676.00	0.05	603.19	0.05	593.39	0.04
	[0.3,0.7]	884.72	0.02	929.00	0.03	881.80	0.06	875.40	0.05
	[0.4,0.6]	1148.00	0.04	1177.00	0.06	1154.40	0.06	1152.47	0.03

Tab 5: Performance of Local Search. Uniform Distribution

Problem	$[b_l, b_u]$	T-KNAP Objective	T-KNAP Time	P-KNAP ($\theta = 10$) Objective	P-KNAP ($\theta = 10$) Time	P-KNAP ($\theta = 50$) Objective	P-KNAP ($\theta = 50$) Time	P-KNAP ($\theta = 100$) Objective	P-KNAP ($\theta = 100$) Time
u15	[0.2,0.8]	554.22	0.01	572.55	<0.01	523.97	<0.01	497.97	<0.01
	[0.3,0.7]	589.03	0.01	606.99	<0.01	581.28	<0.01	570.16	<0.01
	[0.4,0.6]	624.44	<0.01	626.72	<0.01	625.59	<0.01	624.18	<0.01
u20	[0.2,0.8]	685.79	<0.01	710.94	<0.01	660.51	<0.01	643.52	<0.01
	[0.3,0.7]	725.72	<0.01	754.42	<0.01	717.45	<0.01	707.95	<0.01
	[0.4,0.6]	773.19	<0.01	788.28	<0.01	775.52	<0.01	773.07	<0.01
u25	[0.2,0.8]	949.35	0.01	987.40	0.01	912.49	<0.01	873.41	<0.01
	[0.3,0.7]	1013.65	<0.01	1041.25	<0.01	1005.11	<0.01	985.23	<0.01
	[0.4,0.6]	1074.93	0.01	1090.21	<0.01	1078.44	<0.01	1073.87	<0.01
u40	[0.2,0.8]	1280.16	0.01	1321.25	0.02	1200.30	0.03	1155.82	0.01
	[0.3,0.7]	1374.35	0.01	1423.03	0.02	1352.10	0.02	1312.98	0.01
	[0.4,0.6]	1482.34	0.01	1518.68	0.03	1487.55	0.01	1473.11	0.01
u60	[0.2,0.8]	2144.39	0.05	2221.83	0.04	2039.74	0.04	1972.29	0.05
	[0.3,0.7]	2290.66	0.02	2369.24	0.04	2265.08	0.04	2219.56	0.05
	[0.4,0.6]	2443.35	0.05	2498.91	0.04	2450.65	0.07	2439.51	0.04
s15	[0.2,0.8]	265.62	<0.01	259.62	0.01	207.57	<0.01	189.39	<0.01
	[0.3,0.7]	293.93	0.01	303.83	<0.01	269.50	<0.01	256.16	<0.01
	[0.4,0.6]	331.96	<0.01	346.60	<0.01	329.96	<0.01	324.46	<0.01
s20	[0.2,0.8]	383.98	<0.01	373.97	0.01	300.88	0.01	273.34	<0.01
	[0.3,0.7]	425.21	<0.01	436.14	<0.01	388.34	0.01	370.68	0.01
	[0.4,0.6]	480.28	<0.01	501.18	<0.01	475.29	<0.01	467.50	<0.01
s25	[0.2,0.8]	438.07	0.01	426.99	<0.01	342.30	0.01	308.56	0.01
	[0.3,0.7]	483.85	0.01	500.43	<0.01	443.83	0.01	424.05	<0.01
	[0.4,0.6]	547.22	0.01	572.17	<0.01	543.67	<0.01	534.41	<0.01
s40	[0.2,0.8]	716.00	0.01	697.49	0.04	559.35	0.01	509.38	0.02
	[0.3,0.7]	793.29	0.01	816.32	0.04	725.64	0.01	692.25	0.03
	[0.4,0.6]	898.00	0.01	936.15	0.04	889.30	0.03	874.13	0.01
s60	[0.2,0.8]	968.10	0.05	946.04	0.03	757.98	0.03	690.16	0.07
	[0.3,0.7]	1072.76	0.05	1106.61	0.08	981.97	0.15	936.77	0.07
	[0.4,0.6]	1214.73	0.05	1265.81	0.04	1204.99	0.04	1181.38	0.08

Tab. 6: Performance of Local Search. Normal Distribution