

The Stochastic Knapsack Problem as a Tool for Robust Transportation Planning

June 22, 1998

Amy Mainville Cohn
Cynthia Barnhart

Overview

- Motivation
- Formulation and General Properties
- Solution Techniques
- Case Study
- Conclusions

Motivation

- Large-scale transportation problems often solved by decomposition
- Classical sub-problems: Shortest path
 MCF
 Assignment
- Logical to develop robust plans through decomposition as well

Motivation

- Knapsack problem often found in real-world applications
- Current research: -- objective coefficients are random (e.g. finance)
 - objects are dynamic (e.g. telecommunications, freight transport)
- We consider the case where weights are unknown

Formulation

- Deterministic knapsack:
Maximize value of chosen objects
subject to weight not exceeding capacity
- What if weights are random variables?

Formulation

Possible formulations:

- Maximize expected value subject to expected weight not exceeding capacity
- Maximize expected value subject to probability of exceeding capacity being less than a threshold

Formulation

Our approach:

- Objects are tasks or projects to be accepted/rejected.
- Weights represent resource utilization
- We assume that any excess capacity must be “purchased” at some higher rate -- incorporate this as penalty term in objective.

Formulation

Generalization:

- Reward is product of demand, reward scale factor
- Time / labor / physical materials / network capacity
- Penalty is lost good will / fine for failing to complete, or cost of sub-contracting

SEE NEW!!! Formulation

Problem statement:

Maximize expected reward minus expected penalty
subject to objects included exactly once or not at all

SEE NEW!!! Formulation

$$\max \left(\sum_{i=1}^n r_i * \left(\int_{-\infty}^{\infty} y * f_{x_i}(y) dy \right) * x_i \right) - d \left(\int_a^{\infty} (z - a) * f_{x_1, x_2, \dots, x_n}(z) dz \right)$$

$$s.t. \quad x_i \in \{0, 1\}$$

- r_i is the reward factor of object i
- x_i is 1 if object i is chosen, else 0
- d is the per-unit penalty factor
- a is the capacity of the knapsack
- $f_{\{x_1, x_2, \dots, x_n\}}(x)$ is the joint distribution for the chosen set

Formulation

If object weights are independently and Normally distributed, we can find a closed form solution value:

$$f(X) = \left(\sum_{i \in X} r_i \mu_i x_i \right) - d * \left[\sigma * f\left(\frac{a - \mu}{\sigma}\right) + (\mu - a) * G\left(\frac{a - \mu}{\sigma}\right) \right]$$

where mu and sigma are the parameters of the sum of the random weights associated with objects in set X

Properties

The bad news:

- At least as hard as the knapsack problem
- 2^n feasible solutions
- Complex objective function
- Linear relaxation ill-defined

Properties

The good news:

- In practice, complexity of knapsack come from problem size
- Closed form objective for Normal distribution
- Dominance
- $f(S) > f(S \cup i)$ implies $f(S \cup T) > f(S \cup T \cup i)$

Solution Techniques

When all objects are identically distributed and have the same reward factor:

- Question is no longer *which* but *how many*
- Value of including k objects is:

$$f(k) = k * r * \mu - d * \left[\sqrt{k} \sigma * f\left(\frac{a - k\mu}{\sqrt{k} \sigma}\right) + (k\mu - a) * G\left(\frac{a - k\mu}{\sqrt{k} \sigma}\right) \right]$$

- If $f(k) > f(k - 1)$ and $f(k) > f(k + 1)$, k optimal
- Linear search

Solution Techniques

Next, allow reward factors to vary:

- Rank in decreasing order
- Accept until first which decreases objective value

Solution Techniques

Same means and reward factors, varying standard deviations:

- Rank in increasing order
- Accept until first which decreases objective value

Solution Techniques

Varying means, variances, reward factors:

- Ranking no longer possible
- Deterministic approaches:
 - Shortest path?
 - Dynamic programming?

Solution Techniques

- Can't use shortest path -- not strictly additive
- DP is deterministic since weights aren't known until all decisions have been made
- Can't use capacity constraint to restrict size
- May consider all 2^n possibilities
- Need other ways to restrict -- pruning through bounds and dominance

Solution Techniques

- Evaluate tree where each node represents accepting or rejecting the next object in a ranked list.
- Whenever possible, we want to prune branches from the tree to reduce the number of nodes considered

Solution Techniques

Dominance:

- If objects i and j have the same mean, i has greater reward factor, and j has greater variance, then i dominates
- If object i has lower mean and standard deviation than object j but higher expected reward (i.e. the product of mean and reward factor), then i dominates
- If the objective value of the current set decreases when adding object i , then i can be rejected for any set including the current set

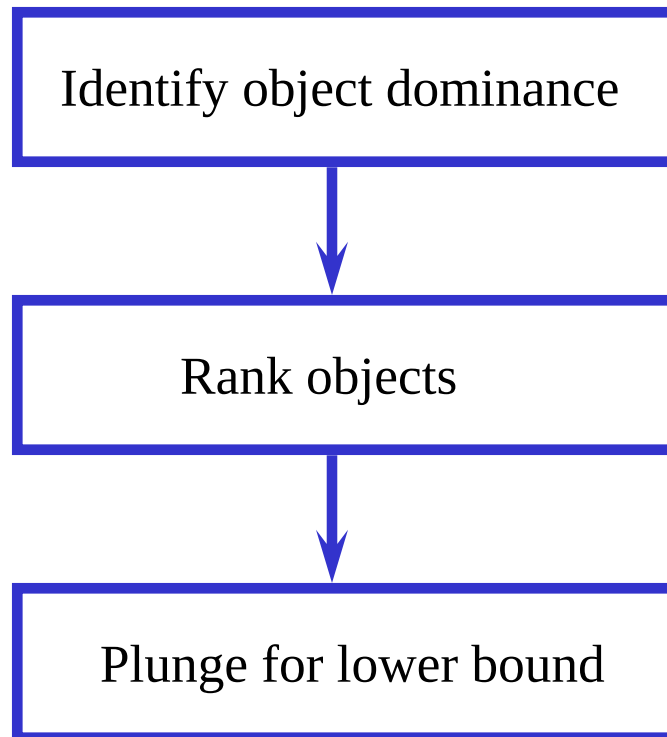
Solution Techniques

Some of the bounds:

- Value of a set of objects is at most the sum of their expected rewards
- Contribution of object i is at most $r_i \mu_i - G((a - \mu_S)/\sigma_S) * d \mu_i$
- Bound on remaining objects in branch is at most $\max\{0, r_i \mu_i - G((a - \mu_S)/\sigma_S) * d \mu_i\}$ summed over remaining i

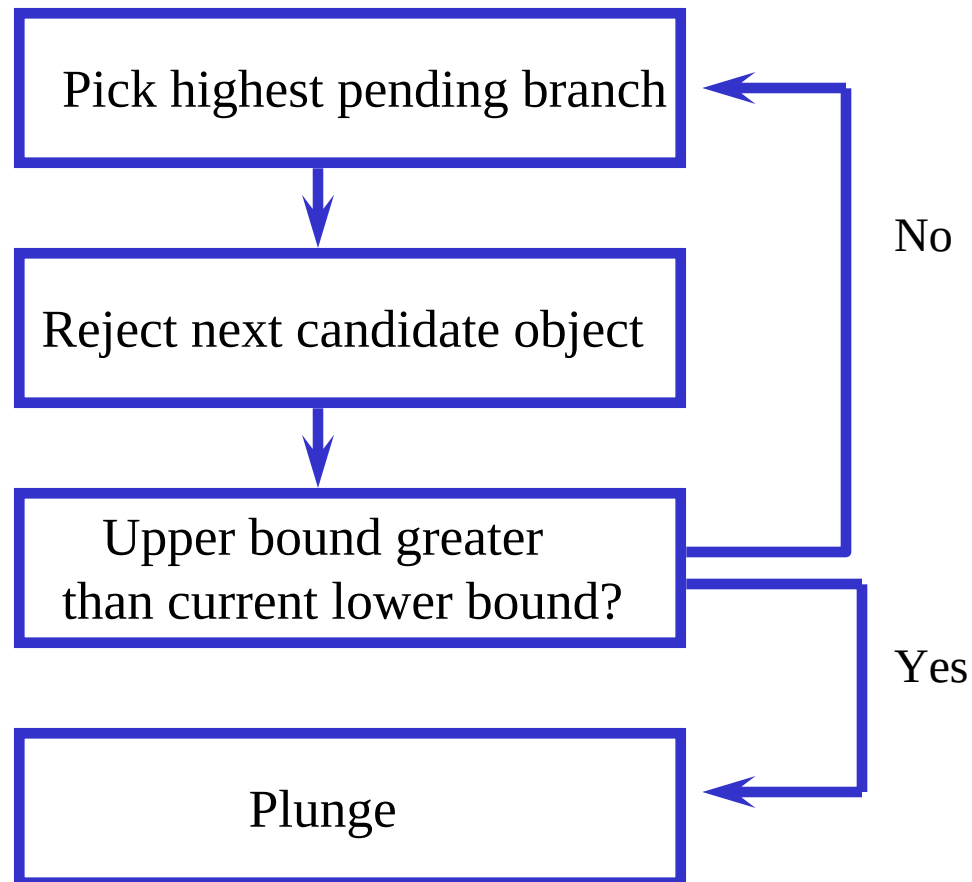
Solution Techniques

Initialization:



Solution Techniques

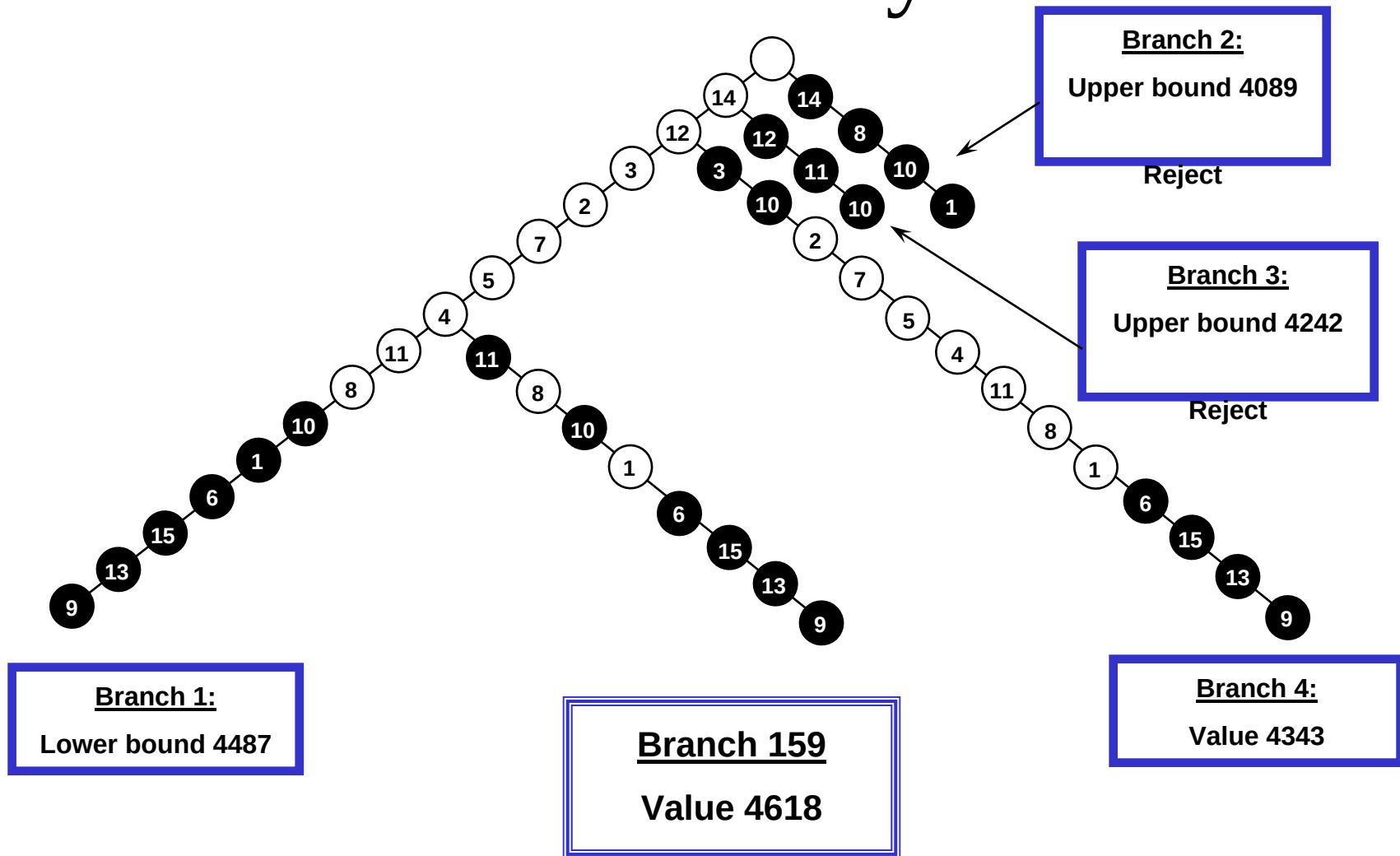
Investigating the tree:



Case Study

Object	Mean	Reward Factor	Variance
1	212	2	47
2	203	2	21
3	246	3	42
4	223	2	21
5	230	2	15
6	233	1	10
7	235	2	11
8	222	2	33
9	210	1	36
10	299	2	42
11	256	2	25
12	250	3	19
13	194	1	24
14	207	3	22
15	182	1	14

Case Study



Case Study

Can heuristics improve over deterministic solution?

One approach: Solve deterministically for many capacity values; compare “true” objective values

Capacity	Set	"True" Value
2000	3 4 5 7 10 11 12 14	4595
1800	1 2 3 4 7 8 12 14	4299
2200	3 4 5 7 8 10 11 12 14	4199
2400	1 3 4 5 7 8 10 11 12 14	3563
2050	1 2 3 4 5 8 11 12 14	4556
2030	3 4 5 7 10 11 12 14	4595

Case Study

Doesn't improve over deterministic solution --
need to incorporate variance.

Augment reward factor by theta times inverse variance

Weight	Set	"True" Value
1	3 4 5 8 10 11 12 14	4569
10	2 3 4 5 10 11 12 14	4531
100	1 2 3 4 5 7 8 12 14	4618
1000	1 2 3 4 5 7 8 12 14	4618
10000	2 3 4 5 6 7 12 14 15	4251

Can we approximate the optimal solution by
choosing the right capacity and variance factor?

Conclusions

Further SKPRW research topics:

- Ranking, branching, pruning rules
- Other penalty functions -- non-linear, object-specific
- Multiple-knapsacks, bin-packing, stochastic capacity

Conclusions

Stochastic programming and robust planning:

- Opportunities to improve implementation results
- De-coupling large problems to incorporate stochasticity
- Use of heuristics to solve stochastic programs
- Potential for studying other classical problems in a stochastic context