ECAI 2014
T. Schaub et al. (Eds.)

doi:10.3233/978-1-61499-419-0-1025

An Algorithm for the Penalized Multiple Choice Knapsack Problem

Elizabeth M. Hilliard[1] and **Amy Greenwald**[2] and **Victor Naroditskiy**[3]

Abstract.

We present an algorithm for the penalized multiple choice knapsack problem (PMCKP), a combination of the more common penalized knapsack problem (PKP) and multiple choice knapsack problem (MCKP). Our approach is to converts a PMCKP into a PKP using a previously known transformation between MCKP and KP, and then solve the PKP greedily. For PMCKPs with well-behaved penalty functions, our algorithm is optimal for the linear relaxation of the problem.

1 Introduction

The knapsack problem (KP) is a classic optimization problem. Due to the large number of real-world problems that can be modeled as KPs, the problem comes in many flavors. We focus on a problem variation that combines two previously studied variations: the penalized knapsack (PKP) and the multiple choice knapsack (MCKP) problem.

2 Knapsack Problems, Global Penalty Functions, and Greedy Algorithms

We begin by presenting various knapsack problems, together with greedy algorithms that solve their linear relaxations optimally.

In addition to a problem instance defined by a set of items, each with a weight and value, and a total capacity, our algorithms take as input a metric m, that describes how to evaluate items (e.g., efficiency), a stopping rule f, that indicates when the algorithm should stop taking items and an item-taking rule g, which determines the fraction of the last item considered to take.

Knapsack Problem A KP is defined by a vector of item values, $\boldsymbol{v} \geq \boldsymbol{0}$, a vector of weights, $\boldsymbol{w} \geq \boldsymbol{0}$ and a hard total capacity, c. A solution is a vector $\boldsymbol{x}$ indicating the amount of each item taken. Thus, the objective is $\max_{\boldsymbol{x}} \boldsymbol{v} \cdot \boldsymbol{x}$ subject to $\boldsymbol{w} \cdot \boldsymbol{x} \leq c$ and each $x_i \in \{0, 1\}$ (for the discrete problem, in which items are indivisible) or each $x_i \in [0, 1]$ (for the relaxed problem, R(KP), in which items may be divisible). (The index i ranges over items.)

GreedyKP (Alg. 1) takes items in order of efficiency until the knapsack reaches capacity, or there are no more items with positive efficiencies.

Theorem [4]: GreedyKP, with efficiency as the metric m, the hard capacity stopping rule (Alg. 2) as f, and the soft taking rule (Alg. 3) as g, solves R(KP) optimally.

[1] Brown University, USA, email: betsy@cs.brown.edu
[2] Brown University, USA, email: amy@cs.brown.edu
[3] University of Southampton, UK, email: vn@ecs.soton.ac.uk

Algorithm 1 GreedyKP

Input: $\boldsymbol{v}, \boldsymbol{w}, c, m, f, g$
Output: $\boldsymbol{x}$
  $\boldsymbol{x} = \boldsymbol{0}$
  for all items i, CalcMetric($i, \boldsymbol{v}, \boldsymbol{w}, m$)
  i = BestUntakenItemIndex
  while $!f(i, \boldsymbol{x}, \boldsymbol{v}, \boldsymbol{w}, c)$ and MoreItemsToConsider **do**
    $x_i = 1$
    i = BestUntakenItemIndex
  $x_i = g(i, \boldsymbol{x}, \boldsymbol{v}, \boldsymbol{w}, c)$
  return $\boldsymbol{x}$

Algorithm 2 HardStoppingRule

Input: $i, \boldsymbol{x}, \boldsymbol{v}, \boldsymbol{w}, c$
Output: {boolean indicating whether to stop taking items or not}
  return $\boldsymbol{x} \cdot \boldsymbol{w} + w_i > c$

Multiple Choice Knapsack Problem An MCKP is defined similarly to a KP, with an additional constraint over a set of types $\mathcal{T}$, which ensures that only one item s is taken from each type set, $T \in \mathcal{T}$. Thus the objective is $\max_{\boldsymbol{x}} \boldsymbol{v} \cdot \boldsymbol{x}$ subject to $\boldsymbol{w} \cdot \boldsymbol{x} \leq c$, $\sum_{s \in T} x_s \leq 1$, $\forall\, T \in \mathcal{T}$, and each $x_i \in \{0, 1\}$, with R(MCKP) defined analogously.

Theorem [6]: GreedyMCKP, with efficiency as the metric m, the hard capacity stopping rule as f, the soft taking rule as g, solves R(MCKP) optimally.

[6]'s algorithm (Alg. 5) proceeds in three steps: first it transforms the given instance of MCKP into a KP (Alg. 4); second it solves the ensuing KP optimally using GreedyKP; third it maps the resulting KP solution back into an optimal solution to the MCKP (Alg. 6).

Penalized Knapsack Problems A PKP is defined by $\boldsymbol{v}$ and $\boldsymbol{w}$, and a penalty function p. The objective is $\max_{\boldsymbol{x}} \pi(\boldsymbol{x})$, where $\pi(\boldsymbol{x}) \equiv \boldsymbol{v} \cdot \boldsymbol{x} - p(\boldsymbol{x}, \boldsymbol{v}, \boldsymbol{w})$ subject to each $x_i \in \{0, 1\}$, with R(PKP) defined analogously. We refer to the penalty functions studied in [2] as *global* because their only input is the knapsack's total weight $\kappa \equiv \boldsymbol{w} \cdot \boldsymbol{x}$. In such cases, it suffices to search greedily with efficiency as the metric; however, for non-global penalty functions, other metrics may be more sensible.

Theorem [2]: If the penalty function is global and convex, then the GreedyPKP algorithm, which invokes GreedyKP with efficiency

Algorithm 3 SoftTakingRule

Input: $i, \boldsymbol{x}, \boldsymbol{v}, \boldsymbol{w}, c$
Output: {fraction of item i to take}
  return $(c - \boldsymbol{x} \cdot \boldsymbol{w})/w_i$

Algorithm 4 REDUCEMCKPTOKP
Input: $\boldsymbol{v}, \boldsymbol{w}, \mathcal{T}$
Output: $\boldsymbol{v}, \boldsymbol{w}$
  SORTBYWEIGHT($\boldsymbol{v}, \boldsymbol{w}$) {Reindex vectors}
  for $T \in \mathcal{T}$ **do**
    $(\boldsymbol{v}|_T, \boldsymbol{w}|_T) =$ REMOVELPDOMINATEDITEMS($\boldsymbol{v}, \boldsymbol{w}$)
    for $i \in [1, |T| - 1]$ **do**
      $((v|_T)_i, (w|_T)_i) = ((\boldsymbol{v}|_T)_{i+1} - (\boldsymbol{v}|_T)_i, (\boldsymbol{w}|_T)_{i+1}) - (\boldsymbol{w}|_T)_i)$
  return $\boldsymbol{v}, \boldsymbol{w}$

Algorithm 5 GREEDYMCKP
Input: $\boldsymbol{v}, \boldsymbol{w}, \mathcal{T}, c, m, f, g$
Output: $\boldsymbol{x}$
  $(\boldsymbol{v}', \boldsymbol{w}') =$ REDUCEMCKPTOKP($\boldsymbol{v}, \boldsymbol{w}, \mathcal{T}$)
  $\boldsymbol{x}' =$ GREEDYKP($\boldsymbol{v}', \boldsymbol{w}', c, m, f, g$)
  return CONVERTKPSOLTOMCKPSOL($\boldsymbol{x}', \boldsymbol{v}', \boldsymbol{w}', \boldsymbol{v}, \boldsymbol{w}$)

as the metric m, the penalized stopping rule (Alg. 7) as f, and the penalized taking rule (Alg. 8) as g, solves R(PKP) optimally.

Penalized Multiple Choice Knapsack Problem A PMCKP is defined by $\boldsymbol{v}$, $\boldsymbol{w}$, $\mathcal{T}$, and a penalty function, p. The objective is $\max_{\boldsymbol{x}} \pi(\boldsymbol{x})$ subject to $\sum_{s \in T} x_s \leq 1, \forall\, T \in \mathcal{T}$ and each $x_i \in \{0, 1\}$, with R(PMCKP) defined analogously.

Theorem 1. *If the penalty function is global, monotonic, non-increasing, and convex, then the* GreedyPMCKP *algorithm, which invokes* GreedyMCKP *with efficiency as the metric m, the penalized stopping rule as f, and the penalized taking rule as g, solves R(PMCKP) optimally.*

Lemma 1. *Let $\boldsymbol{x}^*$ denote an optimal solution to R(PMCKP) with penalty function p, and let κ^* and π^* denote the total weight and total value of $\boldsymbol{x}^*$, respectively. If p is global, monotonic, and non-decreasing, then $\boldsymbol{x}^*$ is also an optimal solution to the corresponding R(MCKP) with capacity κ^*. Furthermore, $\boldsymbol{v} \cdot \boldsymbol{x}^* = \pi^* + p(\kappa^*)$.*

Proof. Suppose not: i.e., suppose $\boldsymbol{x}^*$ is not an optimal solution to the corresponding R(MCKP) with capacity κ^*. Instead, suppose $\boldsymbol{x}$ is optimal, with total weight κ and total value π. Then $\boldsymbol{v} \cdot \boldsymbol{x} > \boldsymbol{v} \cdot \boldsymbol{x}^*$ and $\kappa \leq \kappa^*$. Now, because the penalty function is global, monotonic, and non-decreasing, $p(\kappa) \leq p(\kappa^*)$. But then $\pi = \boldsymbol{v} \cdot \boldsymbol{x} - p(\kappa) \geq \boldsymbol{v} \cdot \boldsymbol{x} - p(\kappa^*) > \boldsymbol{v} \cdot \boldsymbol{x}^* - p(\kappa^*) = \pi^*$. But this is a contradiction, since $\boldsymbol{x}^*$ is optimal. □

Proof of Theorem 1. The proof relies on two observations:

1. Let x denote an *optimal* solution to R(PMCKP), and let κ^x and π^x denote the total weight and total value of x. x is an optimal solution to R(MCKP) with capacity κ^x. The value of the optimal solution to R(MCKP) is $\pi^x + p(\kappa^x)$. (Lemma 1.)
2. Let y denote a *feasible* solution to R(KP), and let κ^y and π^y denote the total weight and total value of y. y is a feasible solution to R(PKP) with total value $\pi^y - p(\kappa^y)$.

Algorithm 6 CONVERTKPSOLTOMCKPSOL
Input: $\boldsymbol{x}', \boldsymbol{v}', \boldsymbol{w}', \boldsymbol{v}, \boldsymbol{w}$
Output: $\boldsymbol{x}$
  $\boldsymbol{x} = \boldsymbol{0}$
  for $T \in \mathcal{T}$ **do**
    $(v^*, w^*) = ((\boldsymbol{v}'|_T) \cdot (\boldsymbol{x}'|_T), (\boldsymbol{w}'|_T) \cdot (\boldsymbol{x}'|_T))$
    let $i^* \in T$ be the greatest i s.t. $(x'|_T)_i > 0$ (or NULL)
    if $i^* \neq$ NULL **then**
      $(x|_T)_{i^*} = (v|_T)_{i^*} / v^*$
  return $\boldsymbol{x}$

Algorithm 7 PENALIZEDSTOPPINGRULE
Input: $i, \boldsymbol{x}, \boldsymbol{v}, \boldsymbol{w}, p$
Output: {boolean indicating whether or not to stop taking items}
  $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{e}^i$ {$\boldsymbol{e}^i$ is a vector of 0s, except the ith entry is a 1}
  return $(\pi(\boldsymbol{x}') < \pi(\boldsymbol{x}))$

Algorithm 8 PENALIZEDTAKINGRULE
Input: $i, \boldsymbol{x}, \boldsymbol{v}, \boldsymbol{w}$
Output: {fraction of x_i to take}
  return $\arg\max_\alpha \boldsymbol{v} \cdot (\boldsymbol{x} + \alpha \boldsymbol{e}^i) - p(\boldsymbol{x} + \alpha \boldsymbol{e}^i, \boldsymbol{v}, \boldsymbol{w})$

Consider an instance of R(PMCKP). Let x denote an optimal solution to this problem, with total weight κ^x and total value π^x. Consider, as well, a corresponding instance of R(PKP) constructed via Zemel's transformation. Let y denote an optimal solution to this problem, with total value π^y.

We claim that $\pi^y \geq \pi^x$. Suppose not: i.e., suppose $\pi^y < \pi^x$. By Fact 1, x is an optimal solution to R(MCKP) with capacity κ^x and total value $\pi^x + p(\kappa^x)$. By Zemel's theorem, x can be converted into a solution, y', to R(KP) with capacity κ^x, such that the total value of y' is $\pi^x + p(\kappa^x)$. Finally, by Fact 2, y' is also a feasible solution to R(PKP) with total value $\pi^x > \pi^y$. This is a contradiction, because y was assumed to be an optimal solution to R(PKP). □

3 Conclusions and Future Work

PMCKP was originally proposed as a model of bidding in ad auctions [1]—specifically, in the context of the annual Trading Agent Competition [3]. Indeed, one of the top-scoring TAC AA agents [5] solved PMCKP using GreedyMCKP as a subroutine inside a search over capacities, but as the space of possible capacities is enormous, it is conceivable that GreedyPMCKP or a variant could perform better. In future work, we plan to investigate the performance of GreedyPMCKP in an ad auctions context.

4 Acknowledgments

This work was supported by NSF Grant #RI-1217761.

REFERENCES

[1] Jordan Berg, Amy Greenwald, Victor Naroditskiy, and Eric Sodomka, 'A first approach to bidding in ad auctions', in *TADA '10: Workshop on Trading Agent Design and Analysis*, (June 2010).
[2] Richard Freling, H. Edwin Romeijn, Dolores Romero Morales, and Albert P. M. Wagelmans, 'A branch-and-price algorithm for the multiperiod single-sourcing problem', *Operations Research*, **51**(6), 922–939, (2003).
[3] P.R. Jordan and M.P. Wellman, 'Designing an ad auctions game for the trading agent competition', in *Workshop on Trading Agent Design and Analysis*, (July 2009).
[4] Hans Kellerer, Ulrich Pferschy, and David Pisinger, *Knapsack problems*, Springer, 2004.
[5] David Pardoe, Doran Chakraborty, and Peter Stone, 'Tactex09: A champion bidding agent for ad auctions', in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, (May 2010).
[6] Eitan Zemel, 'The linear multiple choice knapsack problem', *Operations Research*, **28**(6), 1412–1423, (1980).