A Heuristic Algorithm for General Multiple Nonlinear Knapsack Problem

Luca Mencarelli
mencarelli@lix.polytechnique.fr

Poster Session: Oral Presentations
Mixed-Integer Nonlinear Programming 2014
Carnegie Mellon University, Pittsburgh, USA

June 2, 2014

Joint work with Claudia D'Ambrosio, Angelo Di Zio and Silvano Martello

The problem of the poster

The general multiple nonlinear knapsack problem is

$$
\begin{aligned}
\max \quad & \sum_{i \in M} \sum_{j \in N} f_j(x_{ij}) & & \\
\text{s.t.} \quad & \sum_{j \in N} g_j(x_{ij}) \leq c_i & & \forall i \in M = \{1, \ldots, m\} \\
& \sum_{i \in M} x_{ij} \leq u_j & & \forall j \in N = \{1, \ldots, n\} \qquad \text{(MNKP)} \\
& x_{ij} \geq 0 & & \forall i \in M, \text{ and } \forall j \in N \\
& x_{ij} \in \mathbb{Z} & & \forall i \in M \text{ and } j \in \overline{N} \subseteq N
\end{aligned}
$$

where $x = (x_{11}, x_{12}, \ldots x_{mn}) \in \mathbb{R}^{nm}$, $f_j(x_{ij})$ and $g_j(x_{ij})$ are nonlinear non-negative non-decreasing functions.

The problem of the poster

The general multiple nonlinear knapsack problem is

$$
\begin{aligned}
\max \quad & \sum_{i \in M} \sum_{j \in N} f_j(x_{ij}) & & \\
\text{s.t.} \quad & \sum_{j \in N} g_j(x_{ij}) \leq c_i & & \forall i \in M = \{1, \ldots, m\} \\
& \sum_{i \in M} x_{ij} \leq u_j & & \forall j \in N = \{1, \ldots, n\} \\
& x_{ij} \geq 0 & & \forall i \in M, \text{ and } \forall j \in N \\
& x_{ij} \in \mathbb{Z} & & \forall i \in M \text{ and } j \in \overline{N} \subseteq N
\end{aligned}
\tag{MNKP}
$$

where $x = (x_{11}, x_{12}, \ldots x_{mn}) \in \mathbb{R}^{nm}$, $f_j(x_{ij})$ and $g_j(x_{ij})$ are nonlinear non-negative non-decreasing functions.

No further assumption: $f_j(x_{ij})$ and $g_j(x_{ij})$ can be nonconvex/nonconcave.

NP-hard (generalization of single linear knapsack problem).

Our heuristic algorithm

Generalization of heuristic procedure for single (non)linear knapsack problem [D'Ambrosio and Martello, 2011].

Fast constructive heuristic plus post-processing procedure (local search).

Constructive Heuristic:

Local Search:

Our heuristic algorithm

Generalization of heuristic procedure for single (non)linear knapsack problem [D'Ambrosio and Martello, 2011].

Fast constructive heuristic plus post-processing procedure (local search).

Constructive Heuristic:

- Sort the capacities in non-increasing way: $c_i \leq c_{i+1}$ $(i = 1, \ldots, m-1)$.

Local Search:

Our heuristic algorithm

Generalization of heuristic procedure for single (non)linear knapsack problem [D'Ambrosio and Martello, 2011].

Fast constructive heuristic plus post-processing procedure (local search).

Constructive Heuristic:

- Sort the capacities in non-increasing way: $c_i \leq c_{i+1}$ $(i = 1, \ldots, m-1)$.

- Sampling the profit-to-weight functions and compute the best profit-to-weight ratio for each item.

Local Search:

Our heuristic algorithm

Generalization of heuristic procedure for single (non)linear knapsack problem [D'Ambrosio and Martello, 2011].

Fast constructive heuristic plus post-processing procedure (local search).

Constructive Heuristic:

- Sort the capacities in non-increasing way: $c_i \leq c_{i+1}$ $(i = 1, \ldots, m-1)$.
- Sampling the profit-to-weight functions and compute the best profit-to-weight ratio for each item.
- Apply greedy algorithm for general integer knapsack problem.

Local Search:

Our heuristic algorithm

Generalization of heuristic procedure for single (non)linear knapsack problem [D'Ambrosio and Martello, 2011].

Fast constructive heuristic plus post-processing procedure (local search).

Constructive Heuristic:

- Sort the capacities in non-increasing way: $c_i \leq c_{i+1}$ $(i = 1, \ldots, m-1)$.

- Sampling the profit-to-weight functions and compute the best profit-to-weight ratio for each item.

- Apply greedy algorithm for general integer knapsack problem.

- If current c_i doesn't allow to introduce other items, then $i = i + 1$.

Local Search:

Our heuristic algorithm

Generalization of heuristic procedure for single (non)linear knapsack problem [D'Ambrosio and Martello, 2011].

Fast constructive heuristic plus post-processing procedure (local search).

Constructive Heuristic:

- Sort the capacities in non-increasing way: $c_i \leq c_{i+1}$ $(i = 1, \dots, m-1)$.
- Sampling the profit-to-weight functions and compute the best profit-to-weight ratio for each item.
- Apply greedy algorithm for general integer knapsack problem.
- If current c_i doesn't allow to introduce other items, then $i = i + 1$.

Local Search:

- For a capacity, evaluate local feasible changes between two items.

Our heuristic algorithm

Generalization of heuristic procedure for single (non)linear knapsack problem [D'Ambrosio and Martello, 2011].

Fast constructive heuristic plus post-processing procedure (local search).

Constructive Heuristic:

- Sort the capacities in non-increasing way: $c_i \leq c_{i+1}$ $(i = 1, \ldots, m-1)$.
- Sampling the profit-to-weight functions and compute the best profit-to-weight ratio for each item.
- Apply greedy algorithm for general integer knapsack problem.
- If current c_i doesn't allow to introduce other items, then $i = i + 1$.

Local Search:

- For a capacity, evaluate local feasible changes between two items.
- Take the best one, if it improves the objective function.

Results and conclusions

Test bed: 1680 challenging instances, randomly generated according to [Martello and Toth, 1990] and [D'Ambrosio, Lee and Wächter, 2009].

The heuristic algorithm was compared with:

- heuristic solvers: **Ipopt** for real instance, **Bonmin** for integer instances.
- exact solvers: **Scip**, both for real and integer instances

Results and conclusions

Test bed: 1680 challenging instances, randomly generated according to [Martello and Toth, 1990] and [D'Ambrosio, Lee and Wächter, 2009].

The heuristic algorithm was compared with:

- heuristic solvers: **Ipopt** for real instance, **Bonmin** for integer instances.
- exact solvers: **Scip**, both for real and integer instances

Very difficult multiple nonlinear knapsack problem that nonlinear solvers are unable to handle for instances of realistic size.

Constructive heuristic can provide a high-quality feasible solution to global optimization methods.