A fast combinatorial algorithm for the bilevel knapsack interdiction problem

Ricardo Fukasawa

ICERM - Linear and Non-Linear Mixed Integer Optimization
Feb 28, 2023
Joint work with Noah Weninger

UNIVERSITY OF WATERLOO
FACULTY OF MATHEMATICS
Department of Combinatorics
and Optimization

Outline

Bilevel optimization

Basic optimization problem:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \mathcal{U} \end{aligned} \tag{OPT}$$

Bilevel optimization

Basic optimization problem:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \mathcal{U} \end{aligned} \tag{OPT}$$

Key assumption:

- Decision-maker has total control of ALL decision variables.

Bilevel optimization

Basic optimization problem:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & x \in \mathcal{U} \end{aligned} \tag{OPT}$$

Key assumption:

- Decision-maker has total control of ALL decision variables.
- Often not the case
  - Adversarial settings
  - Multiple (competing) decision makers
  - Not possible to coordinate efforts

Bilevel IP

Two decision-makers (DM):

$$\begin{array}{ll} \min & c^T x + d^T y \\ \text{s.t.} & Ax \leq b \\ & x \in \mathbb{Z}^n \\ & y \in \arg\min\{f^T y : Gx + Hy \leq g, y \in \mathbb{Z}^p\} \end{array} \qquad \text{(BIP)}$$

- x are upper level decision variables
- y are lower level decision variables
- (DM) that controls x acts first, (DM) that controls y reacts
- Applications in military, economics, transportation, electrical grid, etc.

Knapsack Interdiction

- Interdiction idea: Upper level DM can choose to block some of the decisions from lower level DM

Knapsack Interdiction

- Interdiction idea: Upper level DM can choose to block some of the decisions from lower level DM
- Knapsack interdiction:
  - Given a set of items $1, \dots, n$
  - upper weight, lower weight, profit (w_i^U, w_i^L, p_i)
  - Upper/lower knapsack capacities C^U, C^L

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} p_i y_i \\
\text{s.t.} \quad & \sum_{i=1}^{n} w_i^U x_i \le C^U \\
& x \in \{0,1\}^n \\
& y \in \arg\max \left\{ \sum_{i=1}^{n} p_i y_i : \begin{array}{l} \sum_{i=1}^{n} w_i^L y_i \le C^L \\ x_i + y_i \le 1, \forall i = 1, \dots, n \\ y \in \{0,1\}^n \end{array} \right\}
\end{aligned} \tag{BKP}
$$

Knapsack Interdiction

- Interdiction idea: Upper level DM can choose to block some of the decisions from lower level DM
- Knapsack interdiction:
  - Given a set of items $1, \ldots, n$
  - upper weight, lower weight, profit (w_i^U, w_i^L, p_i)
  - Upper/lower knapsack capacities C^U, C^L

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} p_i y_i \\
\text{s.t.} \quad & \sum_{i=1}^{n} w_i^U x_i \leq C^U \\
& x \in \{0,1\}^n \\
& y \in \arg\max \left\{ \sum_{i=1}^{n} p_i y_i : \begin{array}{l} \sum_{i=1}^{n} w_i^L y_i \leq C^L \\ x_i + y_i \leq 1, \forall i = 1, \ldots, n \\ y \in \{0,1\}^n \end{array} \right\}
\end{aligned}
\tag{BKP}
$$

The "Annoying Sibling Problem"

Knapsack Interdiction

- Interdiction idea: Upper level DM can choose to block some of the decisions from lower level DM
- Knapsack interdiction:
  - Given a set of items $1, \ldots, n$
  - upper weight, lower weight, profit (w_i^U, w_i^L, p_i)
  - Upper/lower knapsack capacities C^U, C^L

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} p_i y_i \\
\text{s.t.} \quad & \sum_{i=1}^{n} w_i^U x_i \leq C^U \\
& x \in \{0,1\}^n \\
& y \in \arg\max \left\{ \sum_{i=1}^{n} p_i y_i : \begin{array}{l} \sum_{i=1}^{n} w_i^L y_i \leq C^L \\ x_i + y_i \leq 1, \forall i = 1, \ldots, n \\ y \in \{0,1\}^n \end{array} \right\}
\end{aligned}
\qquad \text{(BKP)}
$$

The "Annoying Sibling Problem"

For each $x \in \mathcal{U} := \left\{ x \in \{0,1\}^n : \sum_{i=1}^{n} w_i^U x_i \leq C^U \right\}$, let $\ell^*(x)$ be an optimizer of the lower level problem.

Knapsack interdiction

- Σ_2^p-complete problem (likely does not admit a poly-sized IP formulation) (Caprara et al., 2014)
- No pseudopolytime algorithm exists (unless $P = NP$)
- *DeNegre (2011) introduced the problem. Solved instances with $\leq$ 15 items
- Caprara, Carvalho, Lodi and Woeginger (2016): Solved instances with $\leq$ 50 items
- *Tang, Richard and Smith (2016): Solved instances with $\leq$ 30 items
- *Fischetti, Ljubic, Monaci, and Sinnl (2019). Solved instances with $\leq$ 55 items
- Lozano, Bergman and Cire (2022). Solved instances with $\leq$ 50 items.
- Della Croce and Scatamacchia (2018). Solved instances with $\leq$ 500 items

Knapsack interdiction

- Σ_2^p-complete problem (likely does not admit a poly-sized IP formulation) (Caprara et al., 2014)
- No pseudopolytime algorithm exists (unless $P = NP$)
- *DeNegre (2011) introduced the problem. Solved instances with $\leq$ 15 items
- Caprara, Carvalho, Lodi and Woeginger (2016): Solved instances with $\leq$ 50 items
- *Tang, Richard and Smith (2016): Solved instances with $\leq$ 30 items
- *Fischetti, Ljubic, Monaci, and Sinnl (2019). Solved instances with $\leq$ 55 items
- Lozano, Bergman and Cire (2022). Solved instances with $\leq$ 50 items.
- Della Croce and Scatamacchia (2018). Solved instances with $\leq$ 500 items

All approaches rely on MIP solvers

Outline

Greedy Heuristic

Procedure GREEDY:

1. Obtain x' by solving $\max\{\sum_{i=1}^{n} p_i x_i : \sum_{i=1}^{n} w_i^U x_i \leq C^U\}$

Greedy Heuristic

Procedure GREEDY:

1. Obtain x' by solving $\max\{\sum_{i=1}^{n} p_i x_i : \sum_{i=1}^{n} w_i^U x_i \leq C^U\}$
2. Obtain $y' = \ell^*(x')$

Greedy Heuristic

Procedure GREEDY:

1. Obtain x' by solving $\max\{\sum_{i=1}^{n} p_i x_i : \sum_{i=1}^{n} w_i^U x_i \leq C^U\}$
2. Obtain $y' = \ell^*(x')$
3. (x', y') is a feasible solution to our problem.

Greedy Heuristic

Procedure GREEDY:

1. Obtain x' by solving $\max\{\sum_{i=1}^{n} p_i x_i : \sum_{i=1}^{n} w_i^U x_i \leq C^U\}$
2. Obtain $y' = \ell^*(x')$
3. (x', y') is a feasible solution to our problem.

Notation:

- $f(S) := \sum_{i \in S} f_i$
- We will use interchangeably $X := \{i : x_i = 1\}$ and x
- $\overline{T} := \{1, \ldots, n\} \setminus T$

When is GREEDY optimal?

Della Croce and Scatamacchia (2018) and Caprara, Carvalho, Lodi and Woeginger (2016) note classes of easy instances.

Lemma (Weninger and F. '22)

GREEDY returns an optimal solution if there exists an optimal solution (X^*, Y^*) *for BKP where* $Y^* = \overline{X^*}$.

Proof.

Suppose $\hat{X} \in \mathcal{U}$ with $p(\hat{X}) > p(X^*)$. Let $\hat{Y} = \ell^*(\hat{X})$

$$p(\hat{Y}) \leq p\left(\overline{\hat{X}}\right) < p\left(\overline{X^*}\right) = p(Y^*)$$

which contradicts optimality of (X^*, Y^*).
Moreover,

$$p(Y') \leq p\left(\overline{X'}\right) = p\left(\overline{X^*}\right) = p(Y^*) \leq p(Y')$$

□

When is GREEDY optimal?

Idea similar from Della Croce and Scatamacchia (2018).

$$
\begin{aligned}
LB(c) = \quad \min \quad & \textstyle\sum_{i=1}^{c-1} p_i(1 - x_i) \\
\text{s.t.} \quad & \sum_{i=1}^{c-1} w_i^U x_i \le C^U \\
& C^L - w_c^L + 1 \le \sum_{i=1}^{c-1} w_i^L (1 - x_i) \le C^L \\
& x \in [0,1]^n
\end{aligned}
$$

If the LP is infeasible for some c, we define $LB(c) = \infty$.

Lemma (Weninger and F. '22)

Suppose GREEDY returns (X', Y') with value z'. If $z' \le \min\{LB(c) : 1 \le c \le n\}$ then (X', Y') is optimal for BKP.

Proof.

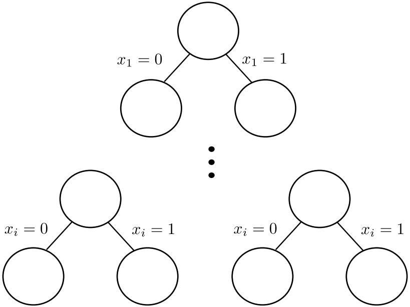
Let (X^*, Y^*) be an optimal solution to BKP. Let $k = \min\{c : \sum_{i:\, i \le c; x_i^* = 0} w_i^L > C^L\}$
If $k = \infty$, then by previous lemma, (X', Y') is optimal for BKP.
Let $xk^* := (x_1^*, \ldots, x_{k-1}^*)$
Then xk^* is feasible for $LB(k)$, so $LB(k) \le \sum_{i=1}^{k-1} p_i(1 - xk_i^*) \le p(y^*)$
So $z' \le \min\{LB(c) : 1 \le c \le n\} \le LB(k) \le p(y^*) \le z'$ □

Outline

Branching

Assume variables are ordered such that $\frac{p^1}{w_1^L} \geq \ldots \geq \frac{p^n}{w_n^L}$

- Branch at depth i will be done on variable x_{i+1}
- Done in DFS, branching first on the right branch
- Check for infeasibility before branching
- At leaf (depth n), we get a possible $\hat{x}$: Use $(\hat{x}, \ell^*(\hat{x}))$ to update primal (upper) bound if needed



depth i-1

Lower bounds

- BB node identified by $X \subseteq \{1, \ldots, i-1\}$ that has been picked by upper level: Identified as (X, i).
- Compute lower bound on node (X, i) by:
  1. solving a lower-level knapsack on items $\{1, \ldots, i-1\} \setminus X$,
  2. computing a lower bound for BKP restricted to items $\{i, \ldots, n\}$, and
  3. combining (1) and (2) into a lower bound for the descendants of (X, i).

Lower bounds

- BB node identified by $X \subseteq \{1, \ldots, i-1\}$ that has been picked by upper level: Identified as (X, i).
- Compute lower bound on node (X, i) by:
  1. solving a lower-level knapsack on items $\{1, \ldots, i-1\} \setminus X$,
  2. computing a lower bound for BKP restricted to items $\{i, \ldots, n\}$, and
  3. combining (1) and (2) into a lower bound for the descendants of (X, i).

Let $K(X, c) := \max\left\{p(Y) : Y \subseteq \{1, \ldots, n\} \setminus X \text{ and } w^L(Y) \leq c\right\}$.

Lower bounds

- BB node identified by $X \subseteq \{1, \ldots, i-1\}$ that has been picked by upper level: Identified as (X, i).
- Compute lower bound on node (X, i) by:
  1. solving $K(X \cup \{i, \ldots, n\}, c)$,
  2. computing a lower bound for BKP restricted to items $\{i, \ldots, n\}$, and
  3. combining (1) and (2) into a lower bound for the descendants of (X, i).

Let $K(X, c) := \max \left\{ p(Y) : Y \subseteq \{1, \ldots, n\} \setminus X \text{ and } w^L(Y) \leq c \right\}$.

Lower bounds

- BB node identified by $X \subseteq \{1, \ldots, i-1\}$ that has been picked by upper level: Identified as (X, i).
- Compute lower bound on node (X, i) by:
  1. solving $K(X \cup \{i, \ldots, n\}, c)$,
  2. computing a lower bound for BKP restricted to items $\{i, \ldots, n\}$, and
  3. combining (1) and (2) into a lower bound for the descendants of (X, i).

Let $K(X, c) := \max \left\{ p(Y) : Y \subseteq \{1, \ldots, n\} \setminus X \text{ and } w^L(Y) \leq c \right\}$.

Let $\omega(i, c^U, c^L) \leq \min\{K(X' \cup \{1, \ldots, i-1\}, c^L) : X' \subseteq \{i, \ldots, n\}, w^U(X') \leq c^U\}$.

Lower bounds

- BB node identified by $X \subseteq \{1, \dots, i-1\}$ that has been picked by upper level: Identified as (X, i).
- Compute lower bound on node (X, i) by:
  1. solving $K(X \cup \{i, \dots, n\}, c)$,
  2. computing $\omega(i, c^U, c^L)$, and
  3. combining (1) and (2) into a lower bound for the descendants of (X, i).

Let $K(X, c) := \max\left\{p(Y) : Y \subseteq \{1, \dots, n\} \setminus X \text{ and } w^L(Y) \leq c\right\}$.

Let $\omega(i, c^U, c^L) \leq \min\{K(X' \cup \{1, \dots, i-1\}, c^L) : X' \subseteq \{i, \dots, n\}, w^U(X') \leq c^U\}$.

Combining (1) and (2)

Lemma

Let (X, i) *be a subproblem. For all* $c \in \{0, \ldots, C^L\}$,

$$K\left(X \cup \{i, \ldots, n\}, c\right) + \omega\left(i, C^U - w^U(X), C^L - c\right) \leq$$

$$\min\left\{p(Y') : (X', Y') \text{ is feasible for BKP and } X' \cap \{1, \ldots, i-1\} = X\right\}.$$

Combining (1) and (2)

Lemma

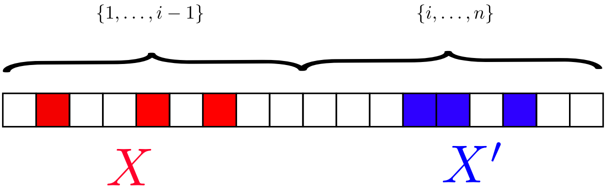
Let (X, i) be a subproblem. For all $c \in \{0, \ldots, C^L\}$,

$$K\left(X \cup \{i, \ldots, n\}, c\right) + \omega\left(i, C^U - w^U(X), C^L - c\right) \leq$$

$$\min\left\{p(Y') : (X', Y') \text{ is feasible for BKP and } X' \cap \{1, \ldots, i-1\} = X\right\}.$$

Proof:

Fix $X \subseteq \{1, \ldots, i-1\}$ and $X' \subseteq \{i, \ldots, n\}$



Then

$$K(X \cup \{i, \ldots, n\}, c) + K(X' \cup \{1, \ldots, i\}, C^L - c) \leq K(X \cup X', C^L)$$

Combining (1) and (2)

Lemma

Let (X, i) be a subproblem. For all $c \in \{0, \ldots, C^L\}$,

$$K\left(X \cup \{i, \ldots, n\}, c\right) + \omega\left(i, C^U - w^U(X), C^L - c\right) \leq$$

$$\min\left\{p(Y') : (X', Y') \text{ is feasible for BKP and } X' \cap \{1, \ldots, i-1\} = X\right\}.$$

Proof:

Fix $X \subseteq \{1, \ldots, i-1\}$ and $X' \subseteq \{i, \ldots, n\}$

$$K(X \cup \{i, \ldots, n\}, c) + K(X' \cup \{1, \ldots, i\}, C^L - c) \leq K(X \cup X', C^L)$$

So:

$$K(X \cup \{i, \ldots, n\}, c) + \min\{K(X' \cup \{1, \ldots, i\}, C^L - c) : X' \subseteq \{i, \ldots, n\}\} \leq$$

$$\leq \min\{K(X \cup X', C^L) : X' \subseteq \{i, \ldots, n\}\}$$

Obtaining ω

Recall: $\omega(i, c^U, c^L) \leq \min\{K(X' \cup \{1, \ldots, i-1\}, c^L) : X' \subseteq \{i, \ldots, n\},\ w^U(X') \leq c^U\}$

Obtaining ω

Recall: $\omega(i, c^U, c^L) \leq \min\{K(X' \cup \{1, \ldots, i-1\}, c^L) : X' \subseteq \{i, \ldots, n\},\ w^U(X') \leq c^U\}$

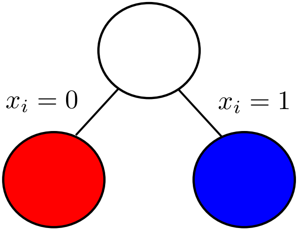
Idea: Apply greedy policy to both children nodes, and pick the lowest.

Obtaining ω

Recall: $\omega(i, c^U, c^L) \leq \min\{K(X' \cup \{1, \ldots, i-1\}, c^L) : X' \subseteq \{i, \ldots, n\},\ w^U(X') \leq c^U\}$

Idea: Apply greedy policy to both children nodes, and pick the lowest.

$$\omega_g(i, c^U, c^L) = \begin{cases} \infty & \text{if } c^U < 0, \\ 0 & \text{if } c^U \geq 0, c^L \geq 0 \text{ and } i > n, \\ \omega_g(i+1, c^U, c^L) & \text{if } c^U \geq 0,\ w_i^L > c^L \text{ and } i \leq n. \\ \min \left\{ \begin{matrix} \omega_g(i+1, c^U - w_i^U, c^L), \\ \omega_g(i+1, c^U, c^L - w_i^L) + p_i \end{matrix} \right\} & \text{if } c^U \geq 0,\ w_i^L \leq c^L \text{ and } i \leq n. \end{cases}$$



Improving ω_g

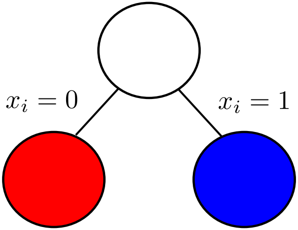
Greedy: Follower always takes item i if possible.

Improving ω_g

Greedy: Follower always takes item i if possible.

Improvement: Allow follower a different choice

$$\omega(i, c^U, c^L) = \begin{cases} +\infty, & \text{if } c^U < 0 \\ \dots & \\ \min \left\{ \begin{array}{l} \omega(i+1, c^U - w_i^U, c^L) \\ \max \left\{ \begin{array}{l} \omega(i+1, c^U, c^L), \\ \omega(i+1, c^U, c^L - w_i^L) + p_i \end{array} \right\} \end{array} \right\} & \text{if } \dots \end{cases}$$



Validity of lower bound

Theorem

For all $1 \leq i \leq n$, $c^U \geq 0$ *and* $c^L \geq 0$,

$$\omega(i, c^U, c^L) \leq \min_{X' \subseteq \{i,\ldots,n\} \,:\, w^U(X') \leq c^U} K(X \cup \{1, \ldots, i-1\}, c^L).$$

Validity of lower bound

Theorem

For all $1 \le i \le n$, $c^U \ge 0$ *and* $c^L \ge 0$,

$$\omega(i, c^U, c^L) \le \min_{X' \subseteq \{i,\ldots,n\} \,:\, w^U(X') \le c^U} K(X \cup \{1, \ldots, i-1\}, c^L).$$

Another interpretation of what is happening:

- Leader and follower are taking turns:
- After leader chooses $x_i = 0$ or 1, follower reacts

Becomes a $2n$-stage game (where each stage is very simple).

How to use good solutions for the follower?

Consider the problem

$$z^* := \min_{x \in \mathcal{U}} \max_{y \in \mathcal{L}(x)} c(x, y) \tag{1}$$

For each $x \in \mathcal{U}$, let $\ell^*(x)$ be an optimizer of $c(x, y)$ for $y \in \mathcal{L}(x)$ (we assume it exists).

Lemma (Weninger and F., 22)

Suppose we have a function $f(x)$ such that for all $x \in \mathcal{U}$:

- $f(x) \in \mathcal{L}(x)$
- $c(x, f(x)) \leq c(x, \ell^*(x)) \leq \alpha c(x, f(x))$, *for some* $\alpha \geq 1$

Let $\tilde{x} \in \arg\min c(x, f(x))$.
Then

$$c(\tilde{x}, \ell^*(\tilde{x})) \leq \alpha z^*$$

How to use good solutions for the follower?

Consider the problem

$$z^* := \min_{x \in \mathcal{U}} \max_{y \in \mathcal{L}(x)} c(x, y) \qquad (1)$$

For each $x \in \mathcal{U}$, let $\ell^*(x)$ be an optimizer of $c(x, y)$ for $y \in \mathcal{L}(x)$ (we assume it exists).

Lemma (Weninger and F., 22)

Suppose we have a function $f(x)$ such that for all $x \in \mathcal{U}$:

- $f(x) \in \mathcal{L}(x)$
- $c(x, f(x)) \leq c(x, \ell^*(x)) \leq \alpha c(x, f(x))$, *for some* $\alpha \geq 1$

Let $\tilde{x} \in \arg\min c(x, f(x))$.
Then

$$c(\tilde{x}, \ell^*(\tilde{x})) \leq \alpha z^*$$

Proof:
Let $(x^*, \ell^*(x^*))$ be the optimal solution to (1).
$\frac{1}{\alpha} c(\tilde{x}, \ell^*(\tilde{x})) \leq c(\tilde{x}, f(\tilde{x})) \leq c(x^*, f(x^*)) \leq c(x^*, \ell^*(x^*)) = z^*$

Instances from the literature

Instances:

- **CCLW:** From Caprara et al (2016), $n \in [35, 55]$
- **DCS:** From DellaCroce and Scatamacchia (2020). $n \in [100, 500]$
- **DeNegre:** From DeNegre (2011). $n \in [10, 50]$
- **FMS:** From Fischetti et al (2018). $n \in [100, 500]$
- **TRS:** Tang et al (2016). $n \in [15, 30]$

Algorithms:

- DCS: Previous best from literature
- Comb: Ours

		DCS				Comb			
Group	#Inst	#Opt	#Best	Avg	Max	#Opt	#Best	Avg	Max
All	1,340	1,271	55	200.49	3,600	1,324	1,269	44.11	3,600
CCLW	50	50	2	0.21	1.27	50	48	0.04	0.07
DCS	500	500	0	3.87	15.73	500	500	0.7	8.59
DeNegre	160	160	50	0.17	1.62	160	110	0.08	1.73
FMS-easy	150	150	0	13.35	79.85	150	150	0.38	7.1
FMS-hard	300	231	0	882.2	3,600	284	284	195.61	3,600
TRS	180	180	3	0.14	2.04	180	177	0.04	0.05

Table: Summary of results for all instances from the literature.

Results per instance type

		DCS				Comb						
Group	#Inst	#Opt	#Best	Avg	Max	#Opt	#Best	Avg	Max	Root	Nodes	Root gap (%)
uncorrelated	940	940	64	2.32	15.73	940	876	0.31	6.74	0.3	12,053.23	5.4
weak correlated	50	50	0	13.49	72.64	50	50	0.26	3.59	0.25	7,130.98	2.04
strong correlated	50	41	0	689.58	3,600	50	50	0.34	3.89	0.26	$5.71 \cdot 10^5$	2.98
inverse strong corr.	50	38	0	919.91	3,600	50	50	1.06	34.24	0.38	$4.75 \cdot 10^6$	0.39
almost strong corr.	50	40	0	815.4	3,600	50	50	0.24	3.17	0.24	735.76	2.99
subset-sum	50	35	0	1,087.18	3,600	42	42	586.33	3,600	0.29	$8.74 \cdot 10^7$	$1.83 \cdot 10^{-2}$
even-odd subset-sum	50	36	0	1,033.98	3,600	42	42	581.43	3,600	0.29	$5.14 \cdot 10^7$	$1.43 \cdot 10^{-2}$
even-odd strong corr.	50	41	0	747.12	3,600	50	50	0.6	16.86	0.26	$2.55 \cdot 10^6$	2.95
similar weight uncorr.	50	50	0	22.89	79.85	50	50	$4.64 \cdot 10^{-2}$	$7.92 \cdot 10^{-2}$	$4.64 \cdot 10^{-2}$	0	0

Table: Results on all instances, by instance type

Variations of the algorithm

	Solution to $K(X \cup \{i, \dots, n\}, c)$	Lower bound on $K(X' \cup \{1, \dots, i-1\}, c^L)$
Comb	Exact	ω
Comb-weak	Exact	ω_g
Comb-greedy	Greedy	ω

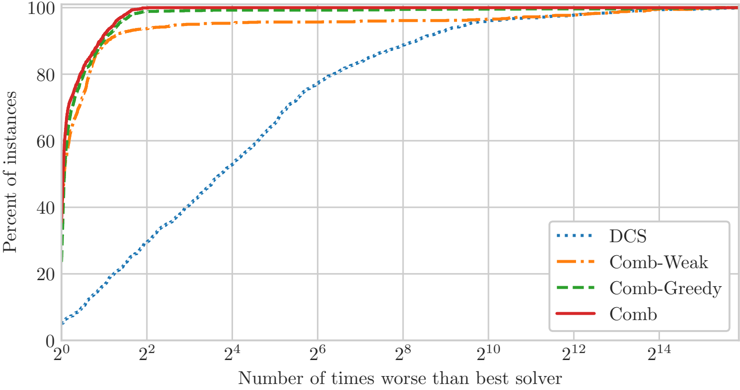


Figure: Performance profile for all instances from the literature.

Influence of threads

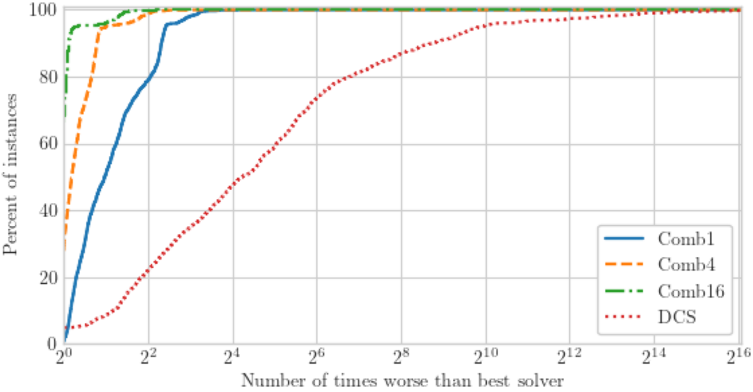


Figure: Performance profile for all instances from the literature, different numbers of threads.

(DCS ran with 16 threads)

New instances

$n \in [10, 10000]$, both easy and hard, also allowing correlation with w_i^U.

		DCS				Comb			
n	#Inst	#Opt	#Best	Avg	Max	#Opt	#Best	Avg	Max
10	250	250	101	0.13	3.41	250	149	0.05	0.13
25	250	238	8	58.36	900	250	242	0.05	0.33
50	250	203	1	178.63	900	247	246	17.83	900
100	250	184	3	253.42	900	222	219	104.77	900
1000	167	109	12	302.26	900	136	124	169.82	900
10000	26	23	0	357.43	900	26	26	12.55	25.21
Class									
uncorrelated	241	239	25	12.37	900	241	216	0.97	25.21
lower subset-sum	256	174	13	318.09	900	237	224	70.2	900
upper subset-sum	232	232	31	2.58	89.25	232	201	0.8	18.29
both subset-sum	232	130	23	417.68	900	189	166	175.93	900
equal weights	232	232	33	2.12	120.55	232	199	0.67	14.97

Table: Summary of results for new instances, grouped by n (upper half) and by class (lower half)

New instances

$n \in [10, 10000]$, both easy and hard, also allowing correlation with w_i^U.

		DCS				Comb			
n	#Inst	#Opt	#Best	Avg	Max	#Opt	#Best	Avg	Max
10	250	250	101	0.13	3.41	250	149	0.05	0.13
25	250	238	8	58.36	900	250	242	0.05	0.33
50	250	203	1	178.63	900	247	246	17.83	900
100	250	184	3	253.42	900	222	219	104.77	900
1000	167	109	12	302.26	900	136	124	169.82	900
10000	26	23	0	357.43	900	26	26	12.55	25.21
Class									
uncorrelated	241	239	25	12.37	900	241	216	0.97	25.21
lower subset-sum	256	174	13	318.09	900	237	224	70.2	900
upper subset-sum	232	232	31	2.58	89.25	232	201	0.8	18.29
both subset-sum	232	130	23	417.68	900	189	166	175.93	900
equal weights	232	232	33	2.12	120.55	232	199	0.67	14.97

Table: Summary of results for new instances, grouped by n (upper half) and by class (lower half)

NOTE: Some instances we ran out of memory (excluded from this table)

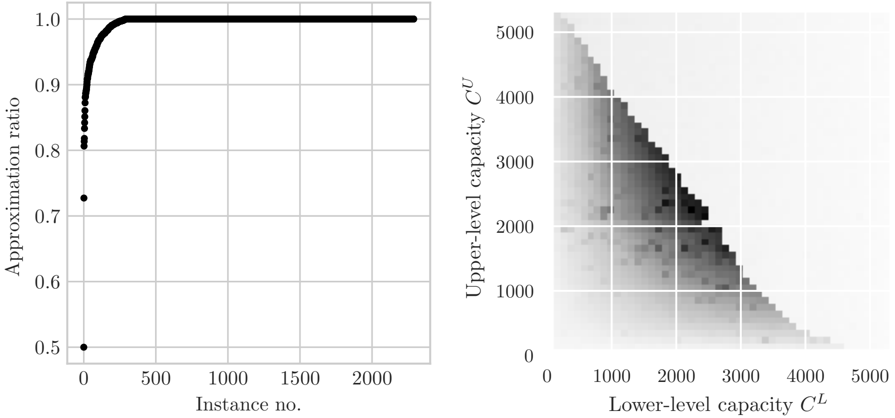


Figure: Left: scatter plot of the lower bound approximation ratio $\omega(1, C^U, C^L)/\text{OPT}$ for all instances (sorted by approximation ratio). Right: running time of Comb (darker = longer time) as a function of C^U and C^L, for an uncorrelated instance with 100 items.

Outline

1. Introduction
2. Our combinatorial algorithm
   - Heuristic
   - Branching
   - Bounding
3. Computational experiments
4. Conclusion

Conclusion

Key ideas:

- Exploit strong lower bounds via good heuristics to lower level
- Idea of using $2n$ rounds of a simpler game
- Lower bounds that are extendable from each other (DP)
- Synchronization with how we branch

Challenges / future research:

- Memory intensive
- Parallelization
- Extension to other problems

Code and new instances available at
`https://github.com/nwoeanhinnogaehr/bkpsolver`

Conclusion

Key ideas:

- Exploit strong lower bounds via good heuristics to lower level
- Idea of using $2n$ rounds of a simpler game
- Lower bounds that are extendable from each other (DP)
- Synchronization with how we branch

Challenges / future research:

- Memory intensive
- Parallelization
- Extension to other problems

Code and new instances available at
`https://github.com/nwoeanhinnogaehr/bkpsolver`

THANK YOU!