Enhanced Kernel Search applied to the Multidimensional Multiple-Choice Knapsack Problem

Renata Mansini, Roberto Zanotti

Department of Information Engineering, University of Brescia, Italy
renata.mansini@unibs.it, roberto.zanotti@unibs.it

Bologna, June 26, 2018

The Multidimensional Multiple-Choice Knapsack Problem

- Given
  - A collection $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$ of n disjoint sets of items
  - Each group G_i has n_i items
  - A set $M = \{1, 2, ..., m\}$ of resources
  - An availability $b_k > 0$ for each resource $k \in M$
  - A profit $p_{ij} > 0$ for each item $j \in G_i$ and $G_i \in \mathcal{G}$
  - A weight $w_{ij}^k \geq 0$ for each $k \in M$, $j \in G_i$ and $G_i \in \mathcal{G}$

The Multidimensional Multiple-Choice Knapsack Problem

- Given
  - A collection $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$ of n disjoint sets of items
  - Each group G_i has n_i items
  - A set $M = \{1, 2, ..., m\}$ of resources
  - An availability $b_k > 0$ for each resource $k \in M$
  - A profit $p_{ij} > 0$ for each item $j \in G_i$ and $G_i \in \mathcal{G}$
  - A weight $w_{ij}^k \geq 0$ for each $k \in M$, $j \in G_i$ and $G_i \in \mathcal{G}$
- Choose one item for each group, maximizing the overall profit, without exceeding the resources

The Multidimensional Multiple-Choice Knapsack Problem

$$\text{Maximize} \sum_{i=1}^{n} \sum_{j=1}^{n_i} p_{ij} x_{ij} \tag{1}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n_i} w_{ij}^k x_{ij} \leq b_k \qquad k = 1, \ldots, m \tag{2}$$

$$\sum_{j=1}^{n_i} x_{ij} = 1 \qquad i = 1, \ldots, n \tag{3}$$

$$x_{ij} \in \{0, 1\} \qquad i = 1, \ldots, n, \;\; j = 1, \ldots, n_i \tag{4}$$

Applications

- **Resource allocation** problems (Gavish & Pirkul [1982])
- **Capital budgeting** problems (Pisinger [2001])
- **Logistics** problems (Basnet & Wilson [2005])
- **Telecommunications** problems (Watson [2001])
- **QoS management** problems (Lee & others [1999])
- many others...

MMKP Literature

- **Heuristic algorithms:**
  - Chen & Hao (2014): group and variable fixing.
  - Mansi, Alves, de Carvalho & Hanafi(2011): iterative variable fixing + problem reformulation
  - Shojaei, Basten, Geilen & Davoodi (2013): compositional pareto-algebraic heuristic
  - Crevits, Hanafi, Mansi & Wilbaut (2012): iterative relaxation based heuristic
  - Hifi & Wu (2015): lagrangian based neighborhood search
- **Exact** algorithms:
  - Kahn (1998): 20 groups, 200 items, 5 resources (w.c.)
  - Sbihi (2007): 50 groups, 1000 items, 7 resources (w.c.)
  - Razzazi & Ghasemi (2008) : 10 groups, 200 items, 3 resources (s.c.).

A basic result

$$UB + \sum_{\{i,j\} \in H^-} b_{ij} x_{ij} - \sum_{\{i,j\} \in H^+} b_{ij}(1 - x_{ij}) + \sum_{k \in M} u_k s_k \geq LB$$

- LB value of a feasible solution (lower bound)
- UB optimal solution value of the LP relaxation
- $s \geq 0$ slack variables and $u \leq 0$ their reduced costs
- b_{ij} reduced cost of x_{ij} variable
- $H^- = \{\{i,j\} : b_{ij} < 0\}$
- $H^+ = \{\{i,j\} : b_{ij} > 0\}$

A basic result

$$UB + \sum_{\{i,j\} \in H^-} b_{ij} x_{ij} - \sum_{\{i,j\} \in H^+} b_{ij}(1 - x_{ij}) + \sum_{k \in M} u_k s_k \geq LB$$

- LB value of a feasible solution (lower bound)
- UB optimal solution value of the LP relaxation
- $s \geq 0$ slack variables and $u \leq 0$ their reduced costs
- b_{ij} reduced cost of x_{ij} variable
- $H^- = \{\{i,j\} : b_{ij} < 0\}$
- $H^+ = \{\{i,j\} : b_{ij} > 0\}$

Then:

$$-\sum_{\{i,j\} \in H^-} b_{ij} x_{ij} + \sum_{\{i,j\} \in H^+} b_{ij}(1 - x_{ij}) - \sum_{k \in M} u_k s_k \leq UB - LB$$

Reduced costs inequality

$$\sum_{\{i,j\}\in H^-} |b_{ij}|x_{ij} + \sum_{\{i,j\}\in H^+} b_{ij}(1-x_{ij}) \leq UB - LB$$

Reduced costs inequality

$$\sum_{\{i,j\}\in H^-} |b_{ij}|x_{ij} + \sum_{\{i,j\}\in H^+} b_{ij}(1-x_{ij}) \leq UB - LB$$

a) Variable Fixing:

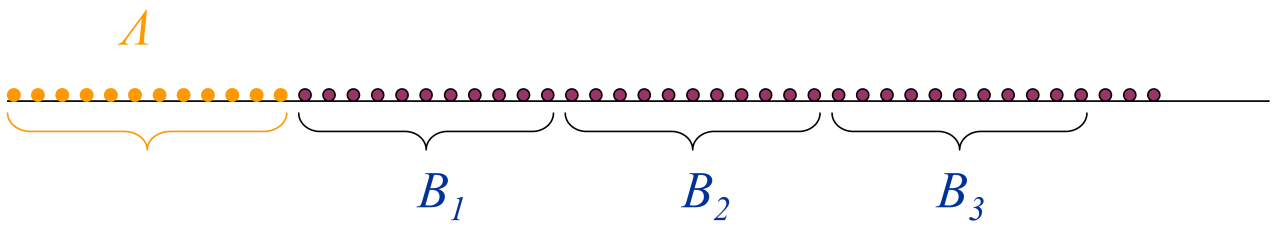
Given two variables x_{ip} and x_{iq} such that $x_{ip}^{LP} = 1$ and $x_{iq}^{LP} = 0$

- If $b_{ip} + \min_{t\in G_i\setminus\{p\}} |b_{it}| > UB - LB$, then $x_{ij} = x_{ij}^{LP}$ for each $j \in G_i$ in every optimal solution
- If $|b_{iq}| + b_{ip} > UB - LB$, then $x_{iq} = x_{iq}^{LP}$ in every optimal solution

Kernel Search 1/3

Main features (Angelelli, Mansini, Speranza [2010], [2012]):

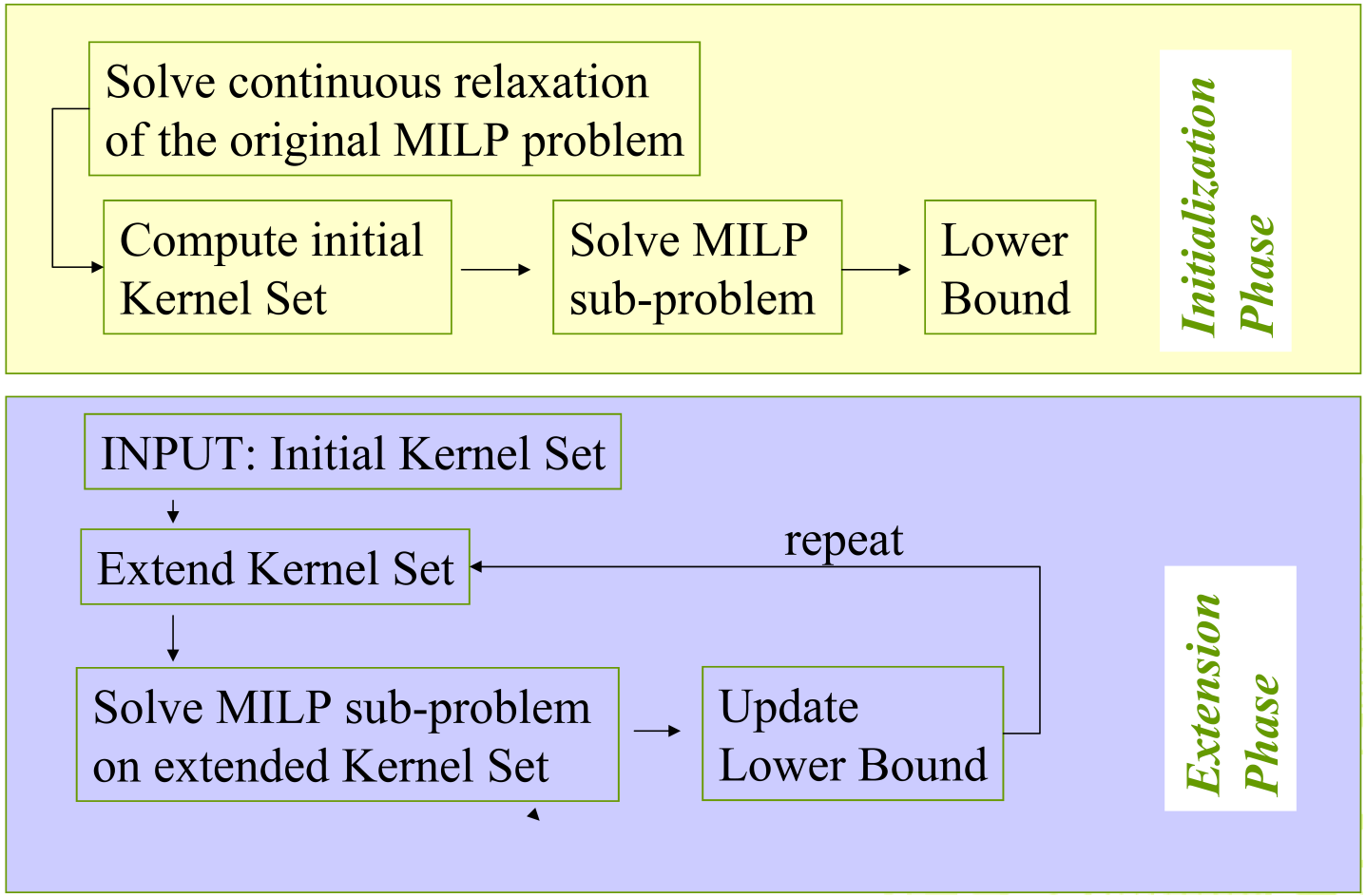
1. Sorting of items (variables) according to a predefined criterion (variable LP values, reduced costs)
2. Identification of a Kernel Set (Λ) of promising variables (more probably selected in an optimal solution) and partition of remaining variables into buckets



3. Construction and solution of a sequence of small subproblems by a MIP solver:

$$\text{Subproblem} = \text{MIP}(\Lambda \cup B) \qquad B \in \mathcal{B}$$

Kernel Search 2/3



Kernel Search 3/3

Critical aspects:

- Manage trade-off between solution quality and computational effort (appropriate size of Kernel Set and buckets)
- Find alternatives to reduced costs when they do not provide a clear signal on promising variables
- Find an effective way for variables to enter/leave the Kernel Set after its initial construction
- Optimize the breakdown of the global solution time used by the method

Adaptive Kernel Search
(Guastaroba, Savelsbergh, Speranza [2017])

Enhanced Kernel Search

Iterative variant: scroll the buckets more than once (bucket iterations)

Innovative aspects:

1. New sorting rule that can be used once a good feasible solution is found
2. Dynamic procedure to eliminate variables from Kernel Set (using information from solved subproblems)
3. Redefinition of the Kernel Set at the beginning of each bucket iteration
4. Dynamic adjustment of subproblems solution time

Variables Sorting: First and Second Iteration

1. Solve the LP relaxation and sort items in non-increasing order of x_{ij}^{LP} value
2. Sort items with $x_{ij}^{LP} = 0$ in non-decreasing order of their absolute reduced cost value

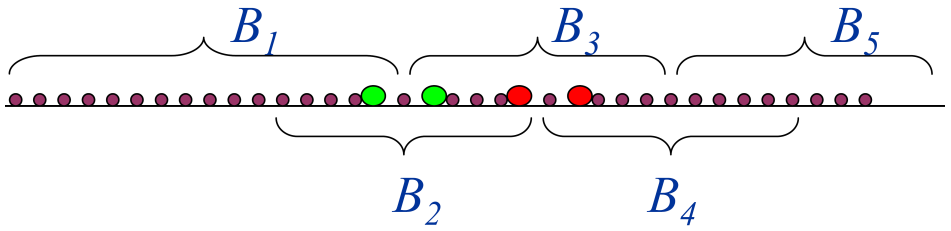
Variables Sorting: Third Iteration

Assuming a feasible solution x^{best} is available

- For each group $G_i \in \mathcal{G}$, if $\exists\ j \mid x_{ij}^{best} \neq x_{ij}^{LP}$
  - solve a new LP relaxation($\tilde{x}^{LP}$) where all variables in group G_i s.t. $x_{ij}^{best} \neq x_{ij}^{LP} \wedge x_{ij}^{LP} > 0$ are fixed to 0.
  - add all items s.t. $\tilde{x}_{ij}^{LP} \neq x_{ij}^{LP} \wedge (i,j) \notin \Lambda$, $i = 1, \ldots, n, j = 1, \ldots, n_i$ to an empty bucket $\tilde{B}$
  - if $\tilde{x}_{ij}^{LP} = x_{ij}^{best}$, $\forall j \in G_i$, **stop**. Otherwise, repeat the procedure adding additional constraints to the LP relaxation.
- if $|\tilde{B}| < \gamma$, add $\gamma - |\tilde{B}|$ items to the bucket, according to the reduced costs sorting, prioritizing groups with smaller $(|\Lambda \cup \tilde{B}) \cap G_i|$
- sort the remaining items in non-decreasing order of absolute reduced cost value.

Buckets and Kernel Set Construction

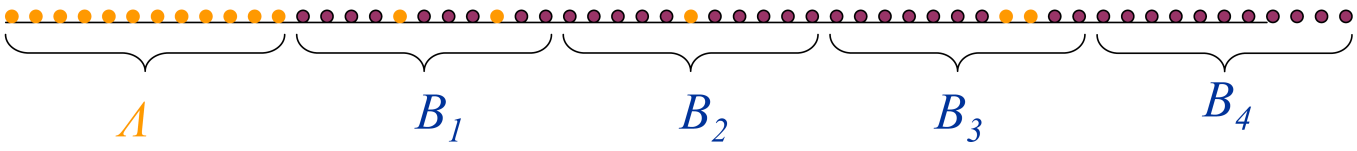
- The first Kernel Set contains all items with value > 0 in the LP relaxation, plus a fixed percentage of the items with value 0 and reduced cost 0 in the LP relaxation.
- At the beginning of each bucket iteration:
  - Kernel Set is reconstructed by including all items of the best integer solution found so far plus the items in the original kernel set
  - Remaining items are divided into buckets of increasing size that partially overlap:



Varying Kernel Set

– Insert procedure:

- $\textsc{SolveSubproblem}(\Lambda \cup B)$ for each $B \in \mathcal{B}$
- Add to Λ all items selected in the subproblem solution



– Eject procedure:

- If $e_{ij} - s_{ij} \geq threshold$, remove item (i, j) from Λ

$\mathcal{S}_{ij} = \{$set of solutions found by KS after item (i, j) entry in $\Lambda\}$

s_{ij} (e_{ij}) number of times (i, j) belongs (does not) to solutions in $\mathcal{S}_{ij}$

Dynamic adjustment of time

For each iteration:

- Kernel Set: $(1+\beta)$ (Global time Iteration)/(max(# buckets - θ, 2))
- Each Bucket: $(1+\beta)$ (Residual time)/(max(residual # buckets - θ, 1))

Motivation: Kernel Set and initial buckets should contain more promising items and thus corresponding subproblems should receive higher computational time

Benchmark problems

- Class A (Khan [1998], Hifi, Michrafy & Sbihi [2006]):
  27 instances, 50 to 400 groups, 10 to 30 items per group, 10 resources.
- Class B (Shojaei, Basten, Geilen & Davoodi [2013]):
  10 instances, 100 to 500 groups, 10 to 20 items per group, 10 to 40 resources

  Strongly correlated instances
  p_{ij} correlated to $\sum_{k \in M} w_{ij}^k$

Computational results

Hardware/software:

- Algorithm Kernel Search has been written in Java 8.0;
- All computational experiments have been conducted on an Intel Core i7 2.93 GHZ with 8GB of RAM. Gurobi Optimizer 7.5.2 is used to solve LP relaxations and MIP subproblems;
- Gurobi 7.5.2 is used as an exact solver for comparison.

Instance	Obj	Ttb	#Sub	avKS	maxKS	avSub	maxSub	ImpIt3+
I07	24595	19	35	115.6	116	287.5	518	0
I08	36895	299	29	171.6	175	402.5	669	1
I09	49186	711	24	222.3	228	509.5	890	1
I10	61480	326	29	276.2	279	662.7	1105	1
I11	73784	123	19	321.5	323	705.4	1008	0
I12	86091	142	24	367.3	368	871.8	1529	0
I13	98439	815	19	421.4	431	934.6	1339	1
INST03	10950	598	32	85.7	92	179.9	281	3
INST04	14449	712	19	93.8	101	181.6	244	3
INST05	17061	456	19	101.8	107	195.5	265	1
INST06	16840	593	26	102.4	108	212.1	350	1
INST07	16444	619	33	100.3	106	233.3	424	1
INST08	17514	650	25	99.8	115	217.0	357	3
INST09	17760	335	19	106.9	111	206.7	279	1
INST10	19321	25	19	105.4	106	219.2	306	0
INST11	19446	728	19	111.7	117	224.9	311	1
INST12	21744	732	24	125.1	140	266.5	452	2
INST13	21580	277	29	120.4	123	620.7	1189	1
INST14	32875	255	24	172.9	176	869.6	1779	1
INST15	39165	763	24	204.0	211	1040.8	2140	2
INST16	43365	188	24	219.7	223	1150.0	2366	1
INST17	54360	95	24	267.3	268	1431.5	2953	0
INST18	60464	198	18	304.3	311	1053.9	1649	1
INST19	64935	275	24	319.8	322	1234.2	2427	1
INST20	75618	316	24	378.2	382	1444.2	2838	1
INST21	44280	883	19	127.4	134	191.5	249	2
INST22	41986	653	19	126.1	131	188.1	228	1
INST23	42524	319	15	132.5	139	200.6	329	0
INST24	41922	641	12	140.5	145	205.9	259	1
INST25	44159	764	37	122.9	131	275.2	471	2
INST26	44878	799	18	127.1	133	229.2	311	1
INST27	87636	625	39	229.4	232	407.6	618	2
INST28	134648	725	18	324.4	331	512.2	670	1
INST29	179226	697	19	426.2	436	684.3	881	2
INST30	214230	662	19	522.8	529	835.7	1087	2
	53423.9	486.2	23.4	205.6	210.9	545.3	936.3	1.2

Class A+B, time limit 900s vs 3600s (I)

Instance	EKS	Gurobi	CHMW	CPH	LHBNS	MACH	PEGF	PERC
$I07$	**24595**	24592	24592	24592	24592	24590	24590	24592
$I08$	36895	36886	**36896**	36886	36895	36889	36894	36892
$I09$	**49186**	49177	49183	49185	49182	49182	49185	49185
$I10$	**61480**	61472	61471	61465	**61480**	**61480**	61477	61478
$I11$	73784	73781	73784	73782	73787	73789	**73791**	73789
$I12$	86091	86083	86091	86084	**86097**	86094	86095	86091
$I13$	98439	98434	**98445**	98437	98444	98440	98441	98441
$INST03$	10950	**10955**	10949	**10955**	10949	10949	10947	10947
$INST04$	14449	**14456**	14446	14452	14445	**14456**	14447	**14456**
$INST05$	**17061**	17060	17058	17059	17060	17057	17057	**17061**
$INST06$	**16840**	16832	16835	16830	16834	16838	**16840**	**16840**
$INST07$	**16444**	16440	16440	16440	16440	16442	**16444**	**16444**
$INST08$	**17514**	17511	17511	17509	17512	17510	17508	**17514**
$INST09$	17760	17761	17760	17757	17761	17760	17763	17763
$INST10$	**19321**	19320	19320	19316	19315	19316	19314	19316
$INST11$	19446	19441	19446	19441	19439	19441	**19449**	**19449**
$INST12$	**21744**	21741	21738	21738	21737	21738	21738	21741
$INST13$	**21580**	21577	**21580**	21577	21577	21577	21578	21578
$INST14$	**32875**	**32875**	32873	32872	32874	**32875**	**32875**	**32875**

Class A+B, time limit 900s vs 3600s (II)

Instance	EKS	Gurobi	CHMW	CPH	LHBNS	MACH	PEGF	PERC
$INST15$	**39165**	39160	39162	39161	**39165**	39161	39161	39162
$INST16$	43365	43364	43366	43363	43366	43366	**43367**	43366
$INST17$	54360	54360	54361	54360	54361	**54363**	**54363**	**54363**
$INST18$	60464	**60468**	60467	60465	**60468**	60467	60467	60467
$INST19$	**64935**	64931	64932	64932	64931	64932	64932	64931
$INST20$	**75618**	75615	75613	75615	75615	**75618**	75616	75614
$INST21$	**44280**	44278	-	44270	-	-	**44280**	**44280**
$INST22$	**41986**	41940	-	41976	-	-	41966	41952
$INST23$	42524	42520	-	42562	-	-	42522	**42584**
$INST24$	**41922**	41890	-	41918	-	-	41876	41860
$INST25$	**44159**	**44159**	-	44156	-	-	44156	**44159**
$INST26$	44878	44865	-	44869	-	-	**44879**	44861
$INST27$	**87636**	87616	-	87616	-	-	87630	87630
$INST28$	**134648**	134644	-	134634	-	-	134642	134642
$INST29$	179226	179220	-	179206	-	-	**179228**	179224
$INST30$	**214230**	214190	-	214198	-	-	214216	**214230**
# best	21	5	3	1	4	5	10	12

Conclusions and Future Developments

Summarizing main results

- LP relaxation is a good starting point for both exact and heuristic methods and reduced costs are a good *"efficiency measure"* to select promising items;
- EKS compares favorably to the State-of-the-Art MMKP heuristics;

To be done

- Determine other ways to identify promising items
- Determine ways to identify promising items that are problem independent