Heuristics and MILP models

M.G. Speranza

University of Brescia

IPDU, Newcastle, July 6-8, 2011

Exact methods for MILP

- ✓General
- ✓More and more powerful
- ✓Commercial software
- ✓Inadequate for instances of ‘large’ size

Heuristics

- ✓Needed where MILP inadequate
- ✓Designed for a specific problem
- ✓Require implementation effort

And if the problem slightly changes?

The heuristic needs to be re-designed

Heuristics + MILP

Can we combine strength of heuristics and strength of MILP?

Heuristics  MILP

Heuristics  MILP

Heuristics → MILP

Goal: To bring into MILP heuristic concepts

- ✓ Local Branching

  Fischetti, Lodi (*Math. Programming* 2003)

- ✓ Relaxation Induced Neighborhood Search (RINS)

  Danna, Rothberg, Le Pape (*Math. Programming* 2005)

Class of MILP problems

$$\max \sum_{j=1}^{n} p_j z_j + \sum_{ij} c_{ij} x_{ij}$$

$$Bz + Ax \le b$$

$$z_j \in \{0,1\} \quad j = 1,\ldots,n$$

Portfolio optimization

Mansini, Speranza, EJOR (1999)

Angelelli, Mansini, Speranza, JCOA (2010)

Multi-dimensional knapsack

Angelelli, Mansini, Speranza, *C&OR* (2010)

Capacitated facility plant location

Guastaroba, Speranza (2011)

Index tracking

Guastaroba, Speranza (2011)

Kernel search

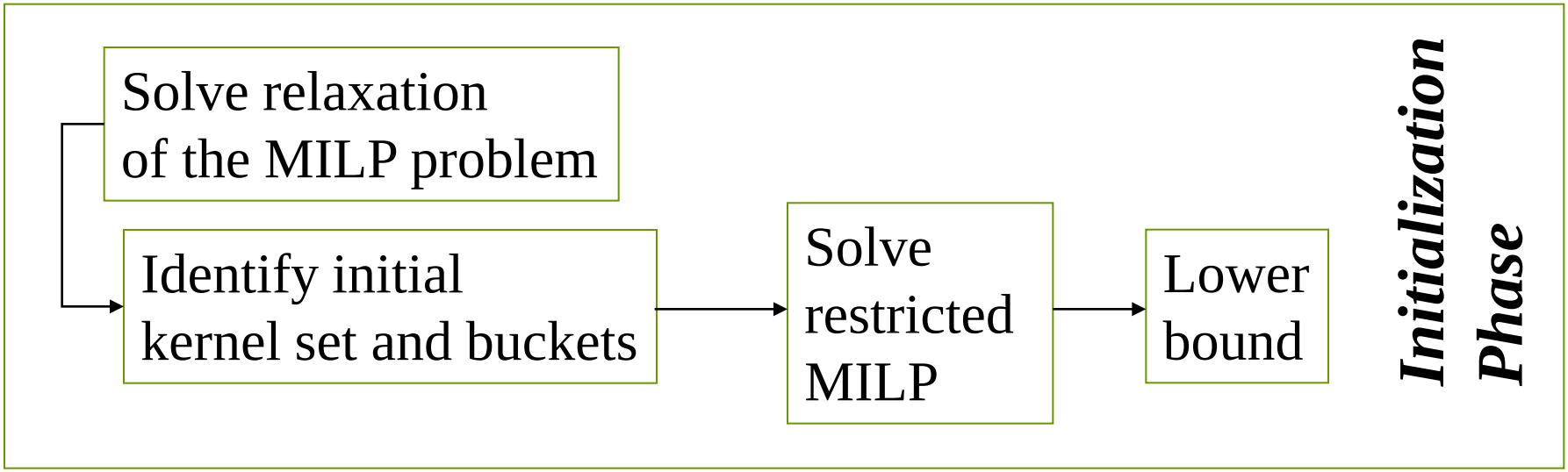
- ✓Heuristic
- ✓Makes use of a MILP solver
- ✓Applicable to a class of MILP problems

Basic concepts:

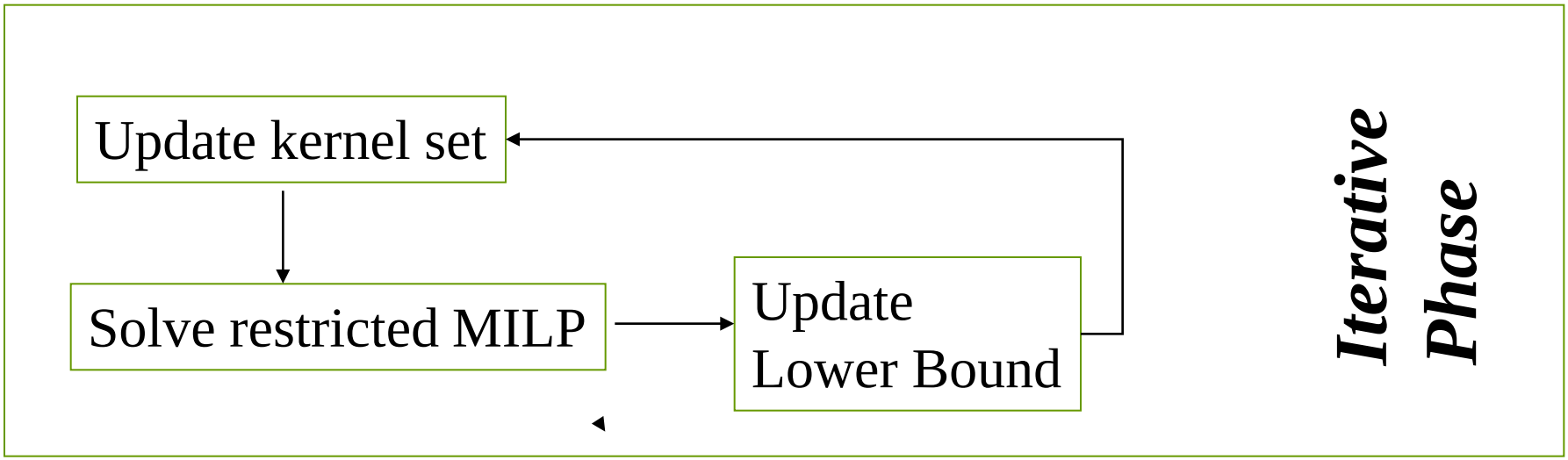
Kernel set = set of promising variables

Iterative solution of MILP restricted to the kernel set

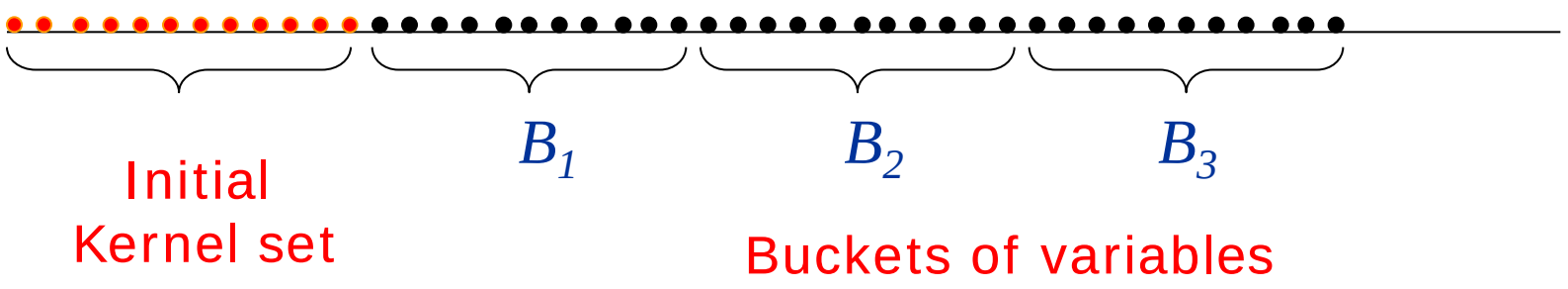
Kernel search



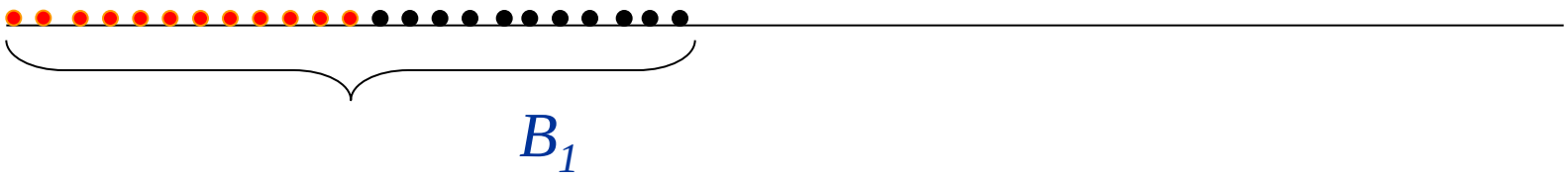
Kernel search



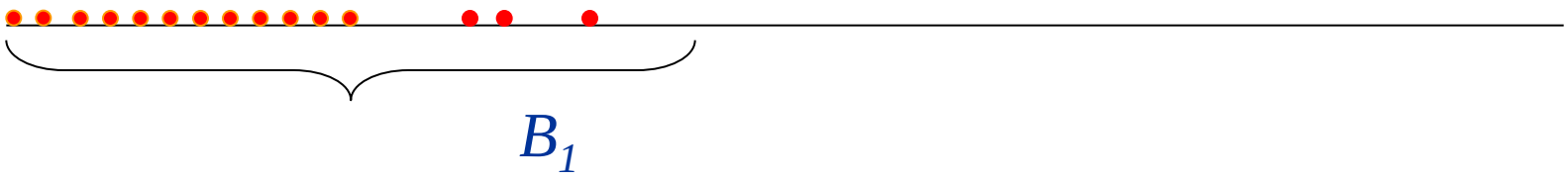
Kernel search - initialization



Kernel search – iterative phase

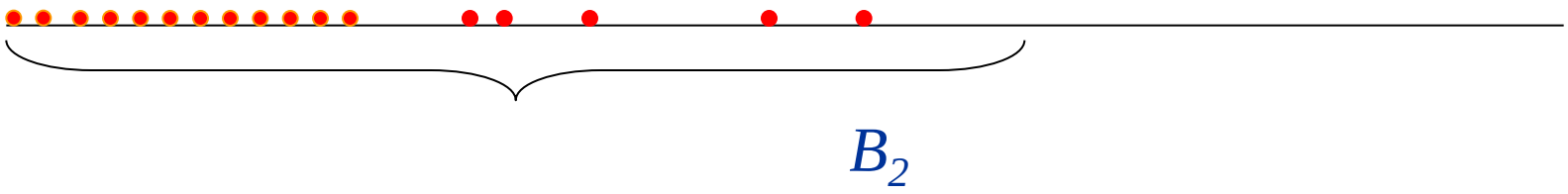


Kernel search – iterative phase

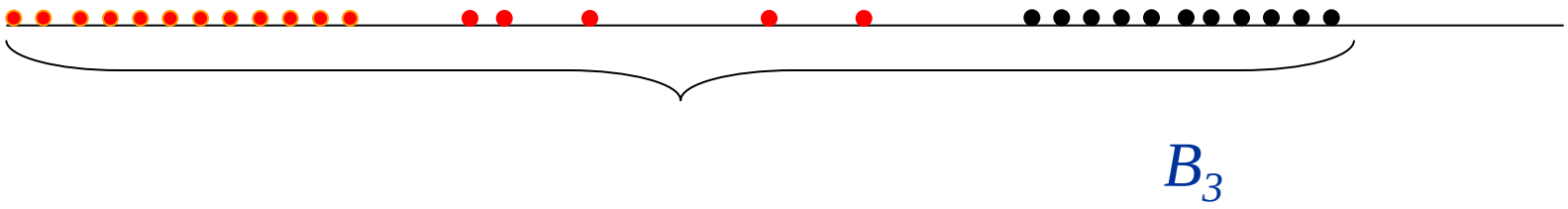


Kernel search – iterative phase

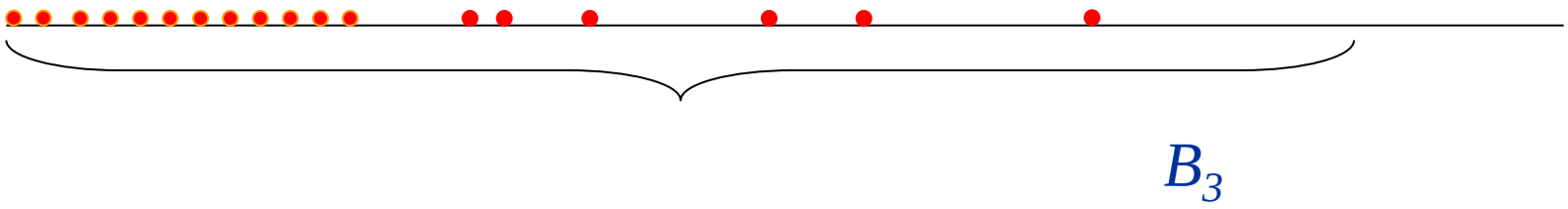
Kernel search – iterative phase



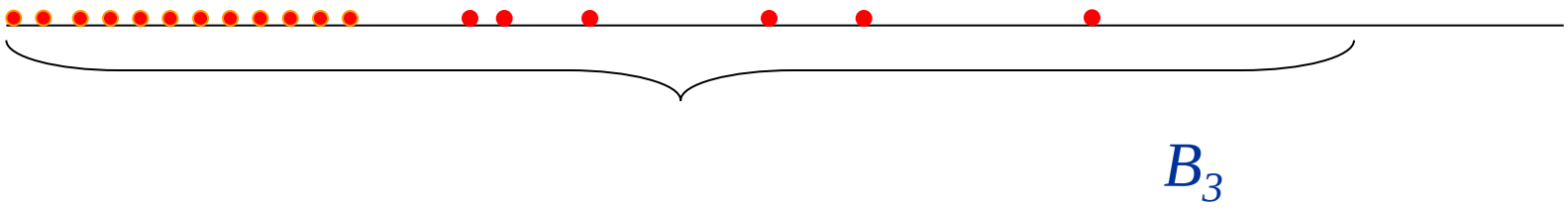
Kernel search – iterative phase



Kernel search – iterative phase



Kernel search – iterative phase



The iterative phase can be repeated

Kernel search

- ✓Initial kernel set
- ✓Sorting of variables
- ✓Creation of buckets

Basic LP variables

Reduced costs

Small or big?

Fixed or variable?

Disjoint or overlapping?

Multi-dimensional knapsack

$$\max \sum_{j=1}^{n} p_j z_j$$

$$\sum_{j=1}^{n} w_{ij} z_j \leq c_i \quad i = 1,\ldots,m$$

$$z_j \in \{0,1\} \quad j = 1,\ldots,n$$

- ✓ Pure binary problem
- ✓ Strongly NP-hard problem
- ✓ Playground for heuristics and meta-heuristics
- ✓ Benchmark instances available

MKP: Kernel search

- ✓ At each iteration:
  - Impose selection of at least one item from the current bucket
  - Impose improvement of the current best solution
- ✓ Increasing size of the kernel set
- ✓ Time limit on the solution of restricted MILP
- ✓ All buckets explored

MKP: Computational results

MILP Solver: CPLEX 9.0

PC AMD Athlon ™ 2000+ Pentium 3GHz, RAM 2GB

Fixed-bucket-I(1), Fixed-bucket-I(0.1); Fixed-bucket-I(0.2)
Time limit: 1 hour

(Buckets of length equal to the number of basic variables, 10% of the number, 20% of the number)

MKP: Chu-Beasley instances

270 instances:

- n = 100, 250, 500; m = 5, 10, 30
  - 30 instances for each pair n,m
  - w_{ij} integer drawn in U(0,1000)
  - $c_i = \alpha \sum_j w_{ij}$ tightness ratio α=0.25, 0.50 and 0.75
  - $p_j = \alpha \sum_i w_{ij} / m + 500\, q_j$ where q_j drawn in U(0,1)

MKP: Instances solved to optimality

Optimal solutions:

$n=100,\ m=5,10,30$

‘easy’ instances

$n=250,\ m=5,10$

$n=500,\ m=5$

Vimont, Boussier, Vasquez

J. of Combinatorial Optimization (2008)

$n=500,\ m=10$

Boussier et al, VI ALIO/EURO, 2008

MKP: Best known solutions

Best known solutions:

$n = 250,\ m = 30$

$n = 500,\ m = 30$

Chu and Beasley (1998)

Vasquez and Vimont (2005)

→ Very large computational times (days)

MKP: Computational results

% average deviations from best known solutions

n	m	F-B-I(1)	F-B-I(0.2)	F-B-I(0.1)
250	5	0	0.003	0.008
250	10	0.001	0.003	0.009
250	30	0.013	0.026	0.025
All		**0.005**	**0.011**	**0.014**
500	5	0.002	0.004	0.004
500	10	0.019	0.021	0.020
500	30	0.062	0.062	0.047
All		**0.028**	**0.029**	**0.024**

MKP: Computational results

Number of equal (improved) solutions
with respect to Vasquez and Vimont (2005)

n	m	F-B-I(1)	F-B-I(0.2)	F-B-I(0.1)
250	5	30(12)	28(12)	24(11)
250	10	29(20)	26(20)	26(16)
250	30	26(20)	25(19)	26(20)
500	5	23(0)	19(0)	19(0)
500	10	8(0)	6(0)	5(1)
500	30	0(0)	0(0)	1(0)

Capacitated Facility Location

$$\min z = \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + \sum_{j \in J} f_j y_j$$

$$s.t. \quad \sum_{i \in I} x_{ij} \leq s_j y_j \qquad j \in J$$

$$\sum_{j \in J} x_{ij} = d_i \qquad i \in I$$

$$x_{ij} \leq d_i \qquad i \in I, j \in J$$

$$x_{ij} \geq 0 \qquad i \in I, j \in J$$

$$y_j \in \{0,1\} \qquad j \in J$$

CFLP: Kernel search

- ✓ At each iteration:
  - Impose selection of at least one item from the current bucket
  - Impose improvement of the current best solution
- ✓ Kernel set includes subsets of x for selected y
- ✓ Subset of buckets explored

CFLP: Instances

49 instances from the OR-library

Optimal solutions are known

100 instances from Avella and Boccia (2009)

Optimal solutions are known for 98 out of 100 instances

295 instances from Avella *et al.* (2009) Only a best bound is known

- Test Bed A: 150 instances with fixed costs two orders of magnitude bigger than the supplying costs
- Test Bed B: 145 instances with fixed costs one order of magnitude bigger than the supplying costs

150 instances generated as in Avella et al. (2009) with fixed costs and supplying costs of the same order of magnitude

(new instances)

Optimal solution is known

Instances		# Inst.	$\|J\|$	$\|I\|$
OR-Library Beasley (1988)	OR-Library-1	13	16	50
	OR-Library-2	12	25	50
	OR-Library-3	12	50	50
	OR-Library-4	12	100	1000
TBED1 Avella and Boccia (2009)	*TBED1*-1	20	300	300
	TBED1-2	20	300	1500
	TBED1-3	20	500	500
	TBED1-4	20	700	700
	TBED1-5 *	20	1000	1000
Test Bed A Avella *et al*. (2009)	Test Bed A-1	30	800	4400
	Test Bed A-2	30	1000	1000
	Test Bed A-3	30	1000	4000
	Test Bed A-4	30	1200	3000
	Test Bed A-5	30	2000	2000
Test Bed B Avella *et al*. (2009)	Test Bed B-1	25	800	4400
	Test Bed B-2	30	1000	1000
	Test Bed B-3	30	1000	4000
	Test Bed B-4	30	1200	3000
	Test Bed B-5	30	2000	2000
Test Bed C generated as in Avella *et al*. (2009)	Test Bed C-1	30	800	4400
	Test Bed C-2	30	1000	1000
	Test Bed C-3	30	1000	4000
	Test Bed C-4	30	1200	3000
	Test Bed C-5	30	2000	2000

Instances	# Inst.	*B-KS* # Opt.	*B-KS* CPU (sec.)
OR-Library-1	13	13	0.3
OR-Library-2	12	12	0.6
OR-Library-3	12	12	0.9
OR-Library-4	12	12	2158.8

Instances	# Inst.	# Opt.	Worst Gap %	CPU (sec.)
TBED1-1	20	20	0.00%	*57.8*
TBED1-2	20	20	0.00%	*68.7*
TBED1-3	20	19	0.02% **	*225.7*
TBED1-4	20	20	0.00%	*795.8*
TBED1-5	20	20	0.00%	*1745.9*

****I-KS** found the optimal solution

CFLP: Computational results

Instances	# Inst.	Impr. %	# Impr.	Opt. Gap %	CPU (sec.)
Test Bed A-1	30	-0.22	26	0.33	1349.9
Test Bed A-2	30	-0.13	26	0.1	336.6
Test Bed A-3	29	-0.34	28	0.27	1539.8
Test Bed A-4	29	-0.3	29	0.18	1571.0
Test Bed A-5	29	-0.09	28	0.07	1382.7

		B-KS			
Instances	# Inst.	Impr. %	# Impr.	Opt. Gap %	CPU (sec.)
Test Bed B-1	25	-1.39	25	0.33	1497.2
Test Bed B-2	30	-0.13	27	0.34	1409.5
Test Bed B-3	29	-0.82	29	0.34	1519.9
Test Bed B-4	30	-0.38	30	0.36	1727.2
Test Bed B-5	30	-0.43	27	0.4	2073.4

CFLP: Computational results

Instances	# Inst.	Opt. Gap %	CPU (sec.)
Test Bed C-1	30	1.51	265.9
Test Bed C-2	30	3.57	1358.4
Test Bed C-3	30	2.09	465.6
Test Bed C-4	30	2.94	1001.2
Test Bed C-5	30	4.65	1833.2

CFLP: A summary

✓B-KS found the optimal solution 146 times out of 147;

✓B-KS improved the best known solution for 275 instances out of 293;

✓improvements are tangible, on average 0.425%, max 5.07%;

✓the few errors are very small (max 0.46%).

Heuristics ← MILP

Goal: To bring into heuristics MILP models

To improve a feasible solution

To intensify the search in some areas of the solution space

.....

⬇

Matheuristics **at next IPDU!**

Conclusions

Kernel search can be improved/extended