Difficult *KP01* Instances

* ***Algorithm PR*** **(Pandit and Ravi-Kumar, 1993) is a "specialized" exact algorithm for *KP01* designed to solve only *SCR instances*:**

* **it is able to solve to optimality *SCR* instances with up to 10,000 items in few CPU minutes,**

* **but it cannot solve *UCR* and *WCR* instances.**

Data Set	n	MT2	DPT	PR
UCR	50	0.01	0.02	-
	100	0.01	0.02	-
	500	0.05	0.09	-
	1000	0.09	0.24	-
	10000	0.50	3.73	-
WCR	50	0.01	0.06	-
	100	0.02	0.08	-
	500	0.08	0.49	-
	1000	0.12	1.66	-
	10000	0.31	5.86	-
SCR	50	0.13	0.51	0.01
	100	134.48 (9)	2.03	0.03
	500	642.62 (4)	45.01	0.50
	1000	-	124.78 (5)	1.94
	10000	-	-	207.48

*** VAXstation 3100 seconds; Time limit = 2000 seconds;**

*** Average time over 10 instances (solved instances if < 10)**

Data Set	*n*	MT2	DPT	PR	Cplex
UCR	50	0.01	0.02	-	0.12
	100	0.01	0.02	-	0.19
	500	0.05	0.09	-	0.72
	1000	0.09	0.24	-	1.51
	10000	0.50	3.73	-	27.84
WCR	50	0.01	0.06	-	0.14
	100	0.02	0.08	-	0.23
	500	0.08	0.49	-	1.29
	1000	0.12	1.66	-	2.54
	10000	0.31	5.86	-	21.81
SCR	50	0.13	0.51	0.01	4.22
	100	(9)	2.03	0.03	(8)
	500	(4)	45.01	0.50	(4)
	1000	-	(5)	1.94	(1)
	10000	-	-	207.48	-

* VAXstation 3100 seconds; Time limit = 2000 seconds;
* Average time over 10 instances (solved instances if < 10)

Additional Test Instances for KP01

* Given: n, generate k random instances as follows

* $C = 0.5 \sum_{j=1,\, n} W_j$

4) *Almost Strongly Correlated (ASCR) Instances:*

* W_j integer uniformly random in [1, 1000] ($j = i, \ldots, n$);

* P_j integer un. rand. in [W_j + 99, W_j + 101] ($j = i, \ldots, n$).

5) *Uncorrelated with Large Weights (ULWR) Instances:*

* W_j integer un. rand. in [100,001, 101,000] ($j = i, \ldots, n$);

* P_j integer uniformly random in [1, 1000] ($j = i, \ldots, n$).

* *Algorithm PR* cannot solve *ASCR* and *ULWR* instances.

Data Set	*n*	MT2	DPT	PR
ASCR	50	0.09	0.51	-
	100	(9)	2.11	-
	500	(5)	44.47	-
	1000	-	118.78 (5)	-
	10000	-	-	-
ULWR	50	0.10	(8)	-
	100	0.02	(5)	-
	500	(7)	-	-
	1000	(8)	-	-
	10000	-	-	-

*** VAXstation 3100 seconds; Time limit = 2000 seconds;**

*** Average time over 10 instances (solved instances if < 10)**

*** Better Upper Bounds for *KP01* are needed: strengthen a relaxed problem *RP* by adding valid inequalities which are redundant for the original problem, but could be violated by the optimal solution of *RP*.**

Stronger Upper Bound for *KP01 (M.-T. 1997)*

* Determine:

Kmax = maximum number of items in a feasible solution

* Sort the items so that $W_1 \leq W_2 \leq \ldots \leq W_n$

$$Kmax = \min \{ k : \Sigma_{j=1,k} W_j > C \} - 1$$

* Example:

$n = 6$; $C = 48$; $(P_j) = (15, 16, 19, 17, 19, 23)$;

$(W_j) = (10, 12, 15, 14, 17, 21)$.

* $s = 4$; $UB_D = 15 + 16 + 19 + [11 * 17 / 14] = 63$

* sorted (W_j): (10, 12, 14, 15, 17, 21), $Kmax = 3$.

* Optimal Solution $(x_j) = (0, 1, 1, 0, 0, 1)$, $z(KP01) = 58$

Stronger Upper Bound for *KP01* (2)

* Determine:

KMAX = maximum number of items in a feasible solution

* Sort the items so that $W_1 \leq W_2 \leq \ldots \leq W_n$

$$KMAX = \min \{ k : \Sigma_{j=1,k} W_j > C \} - 1$$

* Equivalent ILP model for *KP01*:

$$z(KP01) = \max \quad \Sigma_{j=1,n} P_j x_j$$

$$\Sigma_{j=1,n} W_j x_j \leq C$$

$$\Sigma_{j=1,n} x_j \leq KMAX$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n)$$

Stronger Upper Bound **for *KP01* (3)**

* **Equivalent ILP model for *KP01*:**

$$z(KP01) = \max \quad \Sigma_{j=1,n}\ P_j\ x_j \qquad (1)$$

$$\Sigma_{j=1,n}\ W_j\ x_j \ \leq\ C \qquad (2)$$

$$\Sigma_{j=1,n}\ x_j \ \leq\ KMAX \qquad (3)$$

$$x_j \in \{0, 1\} \qquad (j = 1, \ldots, n) \quad (4)$$

* **Lagrangian Relaxation of constraint (3) ($v \geq 0$), and LP Relaxation of the Lagrangian Relaxation:**

$$UB(v) = \max\ (\ \Sigma_{j=1,n}\ P_j\ x_j + v\ (KMAX - \Sigma_{j=1,n}\ x_j))\ \text{s.t. } (2),(4')$$

$$0 \ \leq\ x_j \leq\ 1 \qquad (j = 1, \ldots, n) \qquad (4')$$

Stronger Upper Bound for *KP01* (4)

$$UB(v) = v * KMAX \quad + \ \max \quad \Sigma_{j=1,n} \ P(v)_j \ x_j$$

$$\Sigma_{j=1,n} \ W_j \ x_j \ \le \ C$$

$$0 \ \le \ x_j \le \ 1 \qquad\qquad (j = 1, \ldots, n)$$

with $P(v)_j = P_j - v \qquad\qquad (j = 1, \ldots, n)$

* **The corresponding *Dantzig Upper Bound* is computed.**

* **The *best Lagrangian Multiplier v** (and the corresponding *UB*(*v**)) can be computed in *O(n * n)* time.**

Stronger Upper Bound for *KP01* (5)

$$UB(v) = v * KMAX \; + \; \max \; \Sigma_{j=1,n} \; P(v)_j \, x_j$$

$$\Sigma_{j=1,n} \, W_j \, x_j \; \leq \; C \; ; \quad 0 \; \leq \; x_j \leq \; 1 \quad (\, j = 1, \ldots, n)$$

with $P(v)_j = P_j - v \qquad\qquad (\, j = 1, \ldots, n)$

*** Example:**

$n = 6$; $C = 48$; $(P_j) = (15, 16, 19, 17, 19, 23)$;

$(W_j) = (10, 12, 15, 14, 17, 21)$; $Kmax = 3$.

* $v = 0$, $s(0) = 4$; $UB(0) = 15 + 16 + 19 + [11 * 17 / 14] = 63$

* $v = 5$, sorted items $(P(5)_j) = (10, 14, 11, 12, 18, 14)$;

$(W'_j) = (10, \underline{15}, \underline{12}, 14, \underline{21}, \underline{17})$.

$s(5) = 4$; $UB(5) = 5 * 3 + 10 + 14 + 11 + [11 * 12 / 14] = 59$

Stronger Upper Bound for *KP01* (6)

$$UB(v) = v * KMAX \;+\; \max \;\; \Sigma_{j=1,n}\; P(v)_j\, x_j$$

$$\Sigma_{j=1,n}\; W_j\, x_j \;\le\; C\,; \quad 0 \;\le\; x_j \le\; 1 \quad (\, j = 1, \ldots, n)$$

with $P(v)_j = P_j - v \qquad\qquad (\, j = 1, \ldots, n)$

*** Example:**

$n = 6$; $C = 48$; $(P_j) = (15, 16, 19, 17, 19, 23)$;

$(W_j) = (10, 12, 15, 14, 17, 21)$; $Kmax = 3$.

* $v = 0$, $s(0) = 4$, $UB(0) = 63$; * $v = 5$, $s(5) = 4$, $UB(5) = 59$;

* $v = 10$, sorted items $(P(5)_j) = (13, 9, 9, 5, 6, 7)$;

$(W''_j) = (21, 15, 17, 10, 12, 14)$.

$s(10) = 3$; $UB(10) = 10 * 3 + 13 + 9 + [12 * 9 / 17] = 58$

Alternative Stronger Upper Bound for *KP01*

* Determine: *KMIN* = minimum number of items in an optimal solution.

* Sort the items so that $P_1 \geq P_2 \geq \ldots \geq P_n$

$$KMIN = \min \{ k : \Sigma_{j=1,k} \; P_j \;\; > \;\; LB \}$$

where *LB* (Lower Bound) is the value of a feasible solution.

* Equivalent ILP model for *KP01*:

$$z(KP01) = \max \quad \Sigma_{j=1,n} \; P_j \, x_j$$

$$\Sigma_{j=1,n} \; W_j \, x_j \;\; \leq \;\; C$$

$$\Sigma_{j=1,n} \; x_j \geq KMIN$$

$$x_j \in \{0, 1\} \qquad\qquad (j = 1, \ldots, n)$$

Algorithm MTH for *KP01* (M.-T., Oper. Res. 1997)

*** At each node of the branch-decision tree:**

a) compute the "stronger upper bound" or the "alternative stronger upper bound" by using a parametric technique;

b) if the node is not fathomed, apply the Reduction Procedure, and try to fathom the node through Dominance Criteria;

c) try to fathom the node through a "Partial Dynamic Programming" list.

Data Set	*n*	MT2	DPT	PR	MTH
UCR	50	0.01	0.02	-	0.03
	100	0.01	0.02	-	0.04
	500	0.05	0.09	-	0.08
	1000	0.09	0.24	-	0.14
	10000	0.50	3.73	-	1.59
WCR	50	0.01	0.06	-	0.03
	100	0.02	0.08	-	0.04
	500	0.08	0.49	-	0.09
	1000	0.12	1.66	-	0.15
	10000	0.31	5.86	-	1.28
SCR	50	0.13	0.51	0.01	0.10
	100	(9)	2.03	0.03	0.15
	500	(4)	45.01	0.50	0.71
	1000	-	(5)	1.94	1.31
	10000	-	-	207.48	2.31

* VAXstation 3100 seconds; Time limit = 2000 seconds;
* Average time over 10 instances (solved instances if < 10)

Data Set	*n*	MT2	DPT	PR	MTH
ASCR	50	0.09	0.51	-	0.11
	100	(9)	2.11	-	0.64
	500	(5)	44.47	-	0.78
	1000	-	(5)	-	5.65
	10000	-	-	-	102.83
ULWR	50	0.10	0.02 (8)	-	0.07
	100	0.02	0.06 (5)	-	0.04
	500	(7)	-	-	3.30
	1000	(8)	-	-	0.76
	10000	-	-	-	327.58

*** VAXstation 3100 seconds; Time limit = 2000 seconds;**

*** Average time over 10 instances (solved instances if < 10)**