Progress and issues with SDP-based convexification and convex hull matheuristic for 0-1 quadratic NLPs

Monique Guignard-Spielberg (UPenn),
joint work with Lucas Letocart (Paris 13) and Michael Bussieck (GAMS)

MINLP Workshop -- CMU -- June 2014

Wharton
UNIVERSITY of PENNSYLVANIA

We consider combinatorial optimization problems of the form

$$
\text{(Q01)} \qquad \text{Min} \quad q(x) = x^T Qx + c^T x
$$

$$
\begin{aligned}
\text{s.t.} \quad & Ax = b \\
& A'x \leq b' \\
& x \in \{0,1\}^n
\end{aligned}
$$

The objective function is quadratic:

$$
\begin{aligned}
q(x) &= c^{\mathrm{T}} x + x^{\mathrm{T}} Q x \\
&= \sum_i c_i x_i + \sum_{i,j} x_i Q_{i,j} x_j
\end{aligned}
$$

in addition to the individual costs c_i, there are extra costs incurred when x_i and x_j are simultaneously equal to 1 (interaction costs).

For instance: correlated decisions (portfolio selection)

traffic between locations (quadratic assignment)

traffic inside a cross-dock depends on door assignments

Examples considered in this talk:

QAP or quadratic assignment problem (oldest, very hard)

GQAP or generalized quadratic assignment problem (just as hard or maybe harder)

CDAP or cross-dock assignment problem (a new problem in logistics, very hard too)

QKP and EkQKP or quadratic knapsack problems without or with a cardinality constraint.

FIRST PROBLEM: COMPUTING (IMPROVED) BOUNDS

- for nonconvex problems:
  - reformulation via linearization
  - via convexification

- for convex problems
  - improved bounds via convexification of the set of 0-1 feasible solutions

SIMILAR REFORMULATION TOOLS

(1) use the fact that $x_j \in \{0,1\} \Leftrightarrow (x_j)^2 = x_j$

(2) **introduce new variables** y_{ij} **(RLT)** to replace the products $x_i x_j$, do not write $y_{ij} = x_i x_j$ because it is nonlinear, but add constraints that come from the definition of the new variables (***y*** **is viewed as a vector**, like x):

- nonlinear equations: $y_{ij} = \min(x_i, x_j)$

  $y_{ij} = \max(0, x_i + x_j - 1)$

- or linear inequalities $y_{ij} \le x_i,\ y_{ij} \le x_j$

  $y_{ij} \ge 0,\ y_{ij} \ge x_i + x_j - 1$

(3) **introduce new variables X_{ij} (SDP)** to replace the products $x_i x_j$, do not write $X_{ij} = x_i x_j$ because it is nonlinear, but add constraints that come from the definition of these new variables (***X*** **is viewed as a matrix**)

defined by
$$X = \begin{pmatrix} x_1 x_1 & \cdots & x_1 x_n \\ \vdots & \ddots & \vdots \\ x_n x_1 & \cdots & x_n x_n \end{pmatrix} = x x^{\mathrm{T}}$$

this implies that

- X is of rank 1 (ignore)
- $X - xx^{\mathrm{T}} = 0$ (still nonlinear !) therefore relaxed as $\boldsymbol{X - xx^{\mathrm{T}} \succeq 0}$ (nonlinear, or equivalently (*))
  $$\begin{pmatrix} \mathbf{1} & \boldsymbol{x}^{\mathrm{T}} \\ \boldsymbol{x} & \boldsymbol{X} \end{pmatrix} \succeq \mathbf{0}$$
- $X_{ii} = (x_i)^2 = x_i$

(*) Schur's lemma says that

$$\begin{pmatrix} A & y \\ x & C \end{pmatrix} \succeq 0 \Leftrightarrow C - xA^{-1}y \succeq 0$$

(4) **linearize the objective function:**

in RLT:

$$f(x) = x^{\mathrm{T}} Q\, x = \sum_{i,j} x_i Q_{ij} x_j = \sum_{i,j} Q_{ij} x_i x_j = \sum_{i,j} Q_{ij} y_{ij}$$

in SDP:

$$f(x) = x^{\mathrm{T}} Q\, x = \sum_{i,j} x_i Q_{ij} x_j = \sum_{i,j} Q_{ij} x_i x_j = \sum_{i,j} Q_{ij} X_{ij} = Q \bullet X = \mathrm{tr}(QX)$$

(5) **add constraints that strengthen the continuous bound (RLT)**

multiply each previous constraint by each variable and by (1-that variable) and linearize:

$$x_j * (A_k x = b_k) \;\Rightarrow\; \sum_i x_j A_{ki} x_i = b_k x_j \qquad \Rightarrow\; \sum_i A_{ki} y_{ij} = b_k x_j$$

$$(1\text{-}x_j)*(A_k x = b_k) \quad \Rightarrow \quad \sum_i A_{ki}x_i - \sum_i x_j A_{ki} x_i = b_k - b_k x_j \quad \Rightarrow$$

$$\sum_i A_{ki}x_i + b_k x_j - \sum_i A_{ki} y_{ji} = b_k$$

one can also use symmetry:

$$y_{ij} = y_{ji}$$

(6) the general idea is that **the 0-1 models are "equivalent"** in the following sense:

$\forall x \in FS(P), \exists y$: the o.f. are equal and $(x, y) \in$ FS(RLT)

and

$\forall (x, y) \in FS(RLT)$, the o.f. are equal and $x \in$ FS(P)

(7) one can apply the same process “expanded” to the new set of variables and **iterate**:

$$v(\overline{\mathrm{P}}) \otimes v(\mathrm{RLT\text{-}1}) \otimes v(\mathrm{RLT\text{-}2}) \otimes \ldots\ldots\ldots \otimes v(\mathrm{RLT\text{-}n}) = v(\mathrm{P})$$

where $\otimes$ stands for "is at best as strong as "

(8) **reformulation by convexification of the objective:**

add to the objective function terms which are 0 for every 0-1 feasible solution of the problem and such that

- the expanded objective function is convex
- it yields the best continuous bound among all convex functions of the same type.

(9) if the objective function is convex, one may try to **flatten** it to improve the continuous bound.

(10) if the objective function is pseudoconvex, one can compute the continuous bound over the convex hull of all 0-1 feasible solutions, this is the **convex hull relaxation** of the original model, and this bound is at least as good as the continuous relaxation bound.

QUADRATIC CONVEX REFORMULATION (QCR)

P.L. Hammer, A. Rubin: Some remarks on quadratic programming with 0-1 variables, *RIRO*, (3), 67-79, 1970.

A. Billionnet, S. Elloumi: Using a Mixed Integer Quadratic Programming Solver for the Unconstrained Quadratic 0-1 Problem, *Math. Program.* 109(1), 55-68, 2007.

A. Billionnet, S. Elloumi, M.-Ch. Plateau: Improving the performance of standard solvers for quadratic 0-1 programs by a tight convex reformulation: The QCR method, *Discrete Applied Mathematics* 157(6), 1185-1197, 2009.

GAMS Dev. and M. Guignard, GQAPSDP: SDP Convexifications of the Generalized Quadratic Assignment Problem (GQAPSDP,SEQ=339), *GAMS Model Library*, 2008.

GAMS Dev. and M. Guignard, KQKPSDP: SDP Convexifications of the Cardinality Constraint Quadratic Knapsack Problem (KQKPSDP,SEQ=355), *GAMS Model Library*, 2011.

MQCR

S. H. Ji, X. J. Zheng, X. L. Sun: An improved convex 0-1 quadratic program reformulation for quadratic knapsack problems,*Pacific Journal of Optimization*, 8(1), 75-87, 2012.

M. Guignard, L. Letocart, G. Plateau: MQCR: improved QCR for the 0-1 exact k-item quadratic knapsack problem, *Working Paper*, LIPN, Univ. Paris Nord, Nov. 2013

Convexify $q(x) = x^TQx$

First attempt: Hammer and Rubin

If Q is not sdp, its smallest eigenvalue μ is negative, add to Q a matrix $|\mu\text{-}\varepsilon|I$ and subtract $|\mu\text{-}\varepsilon|x$. The "new" matrix $Q+|\mu\text{-}\varepsilon|I$ will have all its eigenvalues positive and the new objective function will be equal to the old one for any binary vector x. This amounts to changing all diagonal elements by the same amount $(-\mu+\varepsilon)$.

Second attempt: Billionnet and Elloumi. Add different amounts to all diagonal elements of Q, call u_i the change in Q_{ii}, and choose the vector u to get the best lower bound among all possible u's such that $Q+uI$ is psd: this is an SDP problem.

Third attempt: MC Plateau. Add to the previous change a multiple of $x_j\,(A_i\,x-b_i)$ where $A_i\,x=b_i$ is the i^{th} equality constraint. Let the coefficient be α_{ij}, chosen as to yield the best lower bound among all new psd quadratic matrices.

Fourth attempt: Ji, Zheng and Sun (2011)

In addition, enforce some of the conditions that make X_{ij} more like the product of the 0-1 variables x_i and x_j. Indeed, without that, one gets X_{ij} different from x_ix_j.

GQAP properties of matrix X (gams runs calling CSDP, #339, on a Thinkpad T431s Core i7, 64 bits, 8GB RAM, 2.91 Ghz, default is u)

instance	with u	CSDP sec.	with uv	CSDP sec.	# 0-1 vars	number of X(i,j,ip,jp) > x(i,j) with u	%	number of X(i,j,ip,jp) > x(i,j) with uv	%	number of X(i,j,ip,jp) < max(0,x.l(i,j)+x(ij,ip)-1)	%
16x7	-3687670	0.851	-648309	1.013	112	1611	13%	1257	10%	0	0%
20x10	-2218027	1.912	-1676906	2.634	200	10261	26%	9885	25%	0	0%
20-15-35	-4003849	3.974	-3931993	5.03	300	22351	25%	22337	25%	0	0%
20-15-55	-3438241	3.714	-3418839	5.442	300	13226	15%	13476	15%	0	0%
30-06-95	237147	1.435	353601	3.086	180	7999	25%	7529	23%	0	0%
30-07-75	-3384130	1.749	-2441652	3.805	210	11132	25%	11128	25%	0	0%
50-10-95	-14369254	12.2	-13468819	73.6	500	40088	16%	41286	17%	0	0%
moccia60x40	-153320314	1489	-153320271	1652	2400	1345008	23%	1344976	23%	0	0%

M (for Matrix) QCR

Considerons first the case of a quadratic problem without equality constraints.

$$
\text{(QP)} \quad \max \quad q(x) = x^{\mathrm{T}} Q\, x
$$

$$
\text{s.t.} \quad Ax \leq b
$$

$$
x \in \{0,1\}^n
$$

Rewrite $q(x) = x^{\mathrm{T}} Q\, x = x^{\mathrm{T}} (Q - M)\, x + x^{\mathrm{T}} M x$

where M is a matrix such that $Q - M \preceq 0$

and $M = \mathrm{Diag}(\rho) + P + N \in S_n$

with $P \in \mathrm{P}$ and $N \in \mathrm{N}$.

The following always hold for $x \in \{0,1\}^{n}$:

$$x_i x_j = \min(x_i, x_j) \text{ and } x_i x_j = \max(0, x_i + x_j - 1)$$

thus

$$q(x) = x^{T}[Q - Diag(\rho) - P - N]\,x + x^{T}[Diag(\rho) + P + N]x$$

$$= x^{T}[Q - Diag(\rho) - P - N]\,x + \rho^{T}x + \sum_{i,j}[P_{ij}x_i x_j + N_{ij}x_i x_j]$$

$$= x^{T}[Q - Diag(\rho) - P - N]\,x + \rho^{T}x + \sum_{i,j}[P_{ij}s_{ij} - N_{ij}t_{ij}]$$

where $s_{ij} = \min(x_i, x_j)$ and $t_{ij} = -\max(0, x_i + x_j - 1)$.

(QP) can be rewritten as (QP(ρ, P, N)):

$$
\begin{array}{ll}
\text{Max} & q(x) = x^{\mathrm{T}}[Q - Diag(\rho) - P - N]\,x + \rho^{\mathrm{T}} x + \sum_{i,j} [P_{ij} s_{ij} - N_{ij} t_{ij}] \\
\text{s.t.} & Ax \le b\ , \\
 & x \in \{0,1\}^n \\
 & s_{ij} = \min(x_i, x_j) \text{ and } t_{ij} = -\max(0, x_i + x_j - 1) \\
\text{where} & Q - Diag(\rho) - P - N \preceq 0,\ P \ge 0 \text{ and } N \le 0.
\end{array}
$$

$s_{ij} = \min(x_i, x_j)$ and $t_{ij} = -\max(0, x_i + x_j - 1)$ can be replaced by

$s_{ij} \le x_i$ and $s_{ij} \le x_j$ and $t_{ij} \le 0$ and $t_{ij} \le 1 - x_i - x_j$.

(QP) can finally be rewritten as

(QP(ρ,P,N)):

$$\text{Max} \quad q(x) = x^{\mathrm{T}}[Q - Diag(\rho) - P - N]\,x + \rho^{\mathrm{T}} x + \sum_{i,j}[P_{ij} s_{ij} - N_{ij} t_{ij}]$$

$$\text{s.t.} \quad Ax \le b$$

$$s_{ij} \le x_i, \quad s_{ij} \le x_j,$$

$$t_{ij} \le 0, \quad t_{ij} \le 1 - x_i - x_j.$$

$$x \in \{0,1\}^n$$

where $Q - Diag(\rho) - P - N \preceq 0$, $P \ge 0$ and $N \le 0$.

The continuous relaxation of (QP(ρ, P, N)) is ($\overline{\mathrm{QP}}$(ρ, P, N)) :

$$
\begin{array}{ll}
\text{Max} & q(x) = x^{\mathrm{T}}[Q - Diag(\rho) - P - N]\,x + \rho^{\mathrm{T}} x + \sum_{i,j}[P_{ij}s_{ij} - N_{ij}t_{ij}] \\
\text{s.t.} & Ax \le b \\
& s_{ij} \le x_i, \quad s_{ij} \le x_j, \\
& t_{ij} \le 0, \quad t_{ij} \le 1 - x_i - x_j, \\
& 0 \le x \le e
\end{array}
$$

and one wants to determine the values of ρ, P and N that will minimize its optimal value, which is an upper bound on v(QP):

$$
\begin{array}{lll}
\text{(UB)} & \text{Min} & \mathrm{v}(\overline{\mathrm{QP}}(\rho, P, N)) \\
& \text{s.t.} & Q - Diag(\rho) - P - N \preceq 0 \\
& & \rho \in \mathrm{R}^{\mathrm{n}},\ P \ge 0 \text{ and } N \le 0.
\end{array}
$$

(UB) can be solved using any SDP solver.

Extension to the case of equality constraints.

Assune that there is also a linear equality constraint

$$Fx = d,$$

then one can extend the analysis and convexify the problem as follows.

The extended SDP model is

duals

$$\max\ Q.X$$

s.t.	$X_{ii}=x_i,$	$i=1,\ldots,n$	(1)	ρ_i
	$X_{ij}\le x_i,\quad X_{ij}\le x_j,$	$i,j=1,\ldots,n$	(2)	$B_{ij}\ge 0,\ C_{ij}\ge 0$
	$X_{ij}\ge x_i+x_j-1,\ X_{ij}\ge 0,$	$i,j=1,\ldots,n$	(3)	$D_{ij}\ge 0,\ E_{ij}\ge 0$
	$Ax\le b$		(4)	
	$Fx=d$		(5)	
	$\Sigma_j F_{kj} X_{ij} = d_k x_i,$	$i,j=1,\ldots,n,\ k=1,\ldots,m,$	(6)	α_{ik}
	$0\le x\le e$		(7)	
	$\begin{bmatrix} 1 & x^{\mathrm{T}} \\ x & X \end{bmatrix} \succeq 0$		(8)	

Plateau

Hammer-Rubin

The convexified function will be

$$q_{P,N,u,v}(x) = x^{T}(Q\text{-}P\text{-}N)\,x + \sum_i \rho_i(x_i^2 - x_i) + \sum_{k,i} \alpha_{ki}\, x_i (F_k x - d_k)$$

$$+ \sum_{i,j} \boxed{(P_{ij} s_{ij} - N_{ij} t_{ij})}$$

Billionnet, Elloumi

or, in a cheaper version,

Ji, Zheng, Sun

$$q_{P,N,u,v}(x) = x^{T}(Q\text{-}P\text{-}N)\,x + \sum_i \rho_i(x_i^2 - x_i) + v\,\|Fx - d\|^2$$

$$+ \sum_{i,j} (P_{ij} s_{ij} - N_{ij} t_{ij})$$

Guignard, GAMS

with $P = B+C$ and $N = -(D+E)$,

and the convexified model's constraints will be

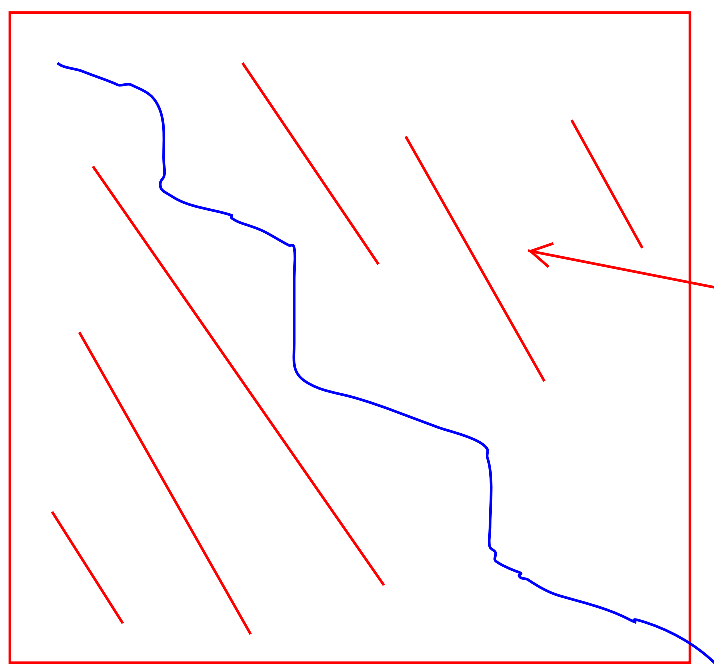
$$Ax \le b$$

$$Fx = d$$

$$\boxed{\begin{array}{ll} s_{ij} \le x_i, \;\; s_{ij} \le x_j, & i, j = 1, \ldots, n \\ t_{ij} \le 0, \;\; t_{ij} \le 1 - x_i - x_j, & i, j = 1, \ldots, n \end{array}}$$

$$x_{ij} \in \{0,1\}.$$

Ji, Zheng, Sun



entire matrix

diagonal

Ji, Zheng, Sun, 2011

Table 2: Comparison results for (QKP) with continuous relaxation ($m = 1$)

n	($\mathrm{DPR_d}$)	($\mathrm{QKP}(\bar{\rho})$)			($\mathrm{TUB_d}$)	($\mathrm{QKP}(\rho^*, P^*, N^*)$)		
	time	time	nodes	rel.error(%)	time	time	nodes	rel.error(%)
60	1.00	1800.00	71350	5.33	5.00	822.67	4784	1.00
70	1.60	1800.00	68077	8.40	6.60	1694.20	7643	1.20
80	1.78	1800.00	76147	21.67	9.00	1659.00	7395	2.63
90	2.00	1800.00	35954	6.00	12.00	1800.00	4930	2.00
100	2.37	1800.00	39933	31.67	13.00	1800.00	4427	5.33

G,L,P

Table 1: Comparison results for (QKP) with continuous relaxation

Instance	($\mathrm{DPR_d}$)	($\mathrm{QKP}(\bar{\rho})$)			($\mathrm{TUB_d}$)	($\mathrm{QKP}(\rho^*, P^*, N^*)$)		
	time	time	nodes	rel.error(%)	time	time	nodes	rel.error(%)
100_100_1	3	3600	39187	12.94	39	3433	5382	0
100_100_2	3	3600	40160	4.82	41	3600	5003	1.36
100_100_3	2	3600	39192	1.20	43	1824	2489	0
100_100_4	3	3600	48886	17.15	37	3600	5237	4.56
100_100_5	3	3600	39526	1.04	42	884	1211	0
100_100_6	2	3600	45209	22.69	35	3600	5118	1.48
100_100_7	5	3600	70218	49.39	38	3600	5059	12.48
100_100_8	2	3600	52620	25.04	36	3600	6538	1.90
100_100_9	6	3600	38654	1.18	44	3600	4931	0.63
100_100_10	3	3600	36432	3.47	39	3600	4861	1.08
100_75_1	4	16	101	0	44	3	0	0
100_75_2	3	3600	52415	13.32	37	3600	6586	4.02
100_75_3	4	3600	19704	19.19	42	3600	2499	2.82
100_75_4	2	3600	24415	20.74	43	3600	2426	5.09
100_75_5	2	3600	77056	35.02	36	3600	4299	7.53
100_75_6	3	3600	13380	3.08	40	2097	1149	0
100_75_7	3	3600	20513	9.91	36	3600	2048	2.12
100_75_8	2	3600	66529	53.50	40	3600	5824	25.33
100_75_9	3	3600	21753	11.14	41	3600	2000	3.85
100_75_10	3	3600	36064	3.50	38	3600	3595	0.40

The experiments were performed on a collection of QKP instances provided by Billionnet and Soutif (`http://cedric.cnam.fr/~soutif/QKP/`).

The comparison results between the two reformulations ($\mathrm{QKP}(\bar{\rho})$) and ($\mathrm{QKP}(\rho^*, \mathrm{P}^*, \mathrm{N}^*)$), when using MIQP solver in CPLEX 12.1 with continuous relaxation, are summarized in Tables 1-2, where

- Columns 2 and 6 of the tables are the CPU time for computing the parameters by solving the SDP problems ($\mathrm{DPR_d}$) and ($\mathrm{TUB_d}$), respectively;
- Columns 3-5 and Columns 7-9 of the tables record the average CPU time, average number of nodes explored and the average relative errors of the two reformulations.

G,L,P

Table 2: Comparison results for (QKP) with continuous relaxation

Instance	$(\mathrm{DPR_d})$	$(\mathrm{QKP}(\bar{\rho}))$			$(\mathrm{TUB_d})$	$(\mathrm{QKP}(\rho^*, P^*, N^*))$		
	time	time	nodes	rel.error(%)	time	time	nodes	rel.error(%)
100_50_1	5	3600	37773	5.39	37	2545	3064	0
100_50_2	3	3600	18774	3.08	39	3600	2134	0.85
100_50_3	3	3600	29594	19.77	36	3600	2674	3.70
100_50_4	3	3600	19298	2.45	39	2133	1304	0
100_50_5	3	3600	23087	11.49	36	3600	2080	1.47
100_50_6	3	3600	31420	45.55	36	3600	2638	13.88
100_50_7	3	3600	25417	14.83	37	3600	2121	1.28
100_50_8	3	3600	22136	12.17	37	3600	2176	2.04
100_50_9	3	3600	22426	10.01	39	3600	2156	3.18
100_50_10	3	3600	32801	5.64	37	3600	2673	1.65
200_100_1	16	3600	3965	1.81	456	3600	497	0.90
200_100_2	15	3600	5054	24.19	412	3600	578	2.16
200_100_3	14	3600	7091	116.26	432	3600	700	55.73
200_100_4	13	3600	6691	103.90	379	3600	599	8.01
200_100_5	13	3600	3869	6.32	434	3600	532	2.42
200_100_6	14	3600	7050	102.40	386	3600	600	38.48
200_100_7	13	3600	3776	73.9	487	3600	497	1.89
200_100_8	14	3600	3616	6.23	461	3600	500	1.06
200_100_9	13	3600	3926	10.10	443	3600	506	3.60
200_100_10	13	3600	4711	23.25	406	3600	565	5.38
300_50_1	46	3600	2583	10.08	1488	3600	384	3.23
300_50_2	49	3600	1987	117.78	1493	3600	190	42.67
300_50_3	47	3600	1269	8.63	1520	3600	140	1.36
300_50_4	45	3600	1675	8.31	1635	3600	221	2.12
300_50_5	54	3600	1679	3.99	1625	3600	204	1.43

Table 6: Comparison results for (k-QKP) with continuous relaxation ($m = 1,\ K = \lfloor \frac{n}{8} \rfloor$)

n	(k-DPR$_\mathrm{d}$)	(k-QKP($\bar{\rho}, \bar{\alpha}$))			(k-TUB$_\mathrm{d}$)	(k-QKP($\rho^*, \alpha^*, P^*, N^*$))		
	time	time	nodes	rel.error(%)	time	time	nodes	rel.error(%)
60	1.00	1800.00	69836	5.84	6.00	1800.00	18800	4.17
70	2.00	1800.00	53224	8.90	7.20	1800.00	14551	7.64
80	2.00	1800.00	44465	13.05	9.40	1800.00	12746	12.11
90	2.20	1800.00	38035	13.52	12.00	1800.00	9397	11.21
100	3.00	1800.00	29276	15.74	15.20	1800.00	7569	12.86

A convex form of the quadratic assignment problem, DJ White, *EJOR* 65 407-416, 1993)

For 0-1 variables :

$x_i x_j \; = 1/2(x_i + x_j)^2 - 1/2\, x_i^2 - 1/2\, x_j^2 \; = 1/2(x_i + x_j)^2 - 1/2\, x_i - 1/2\, x_j$

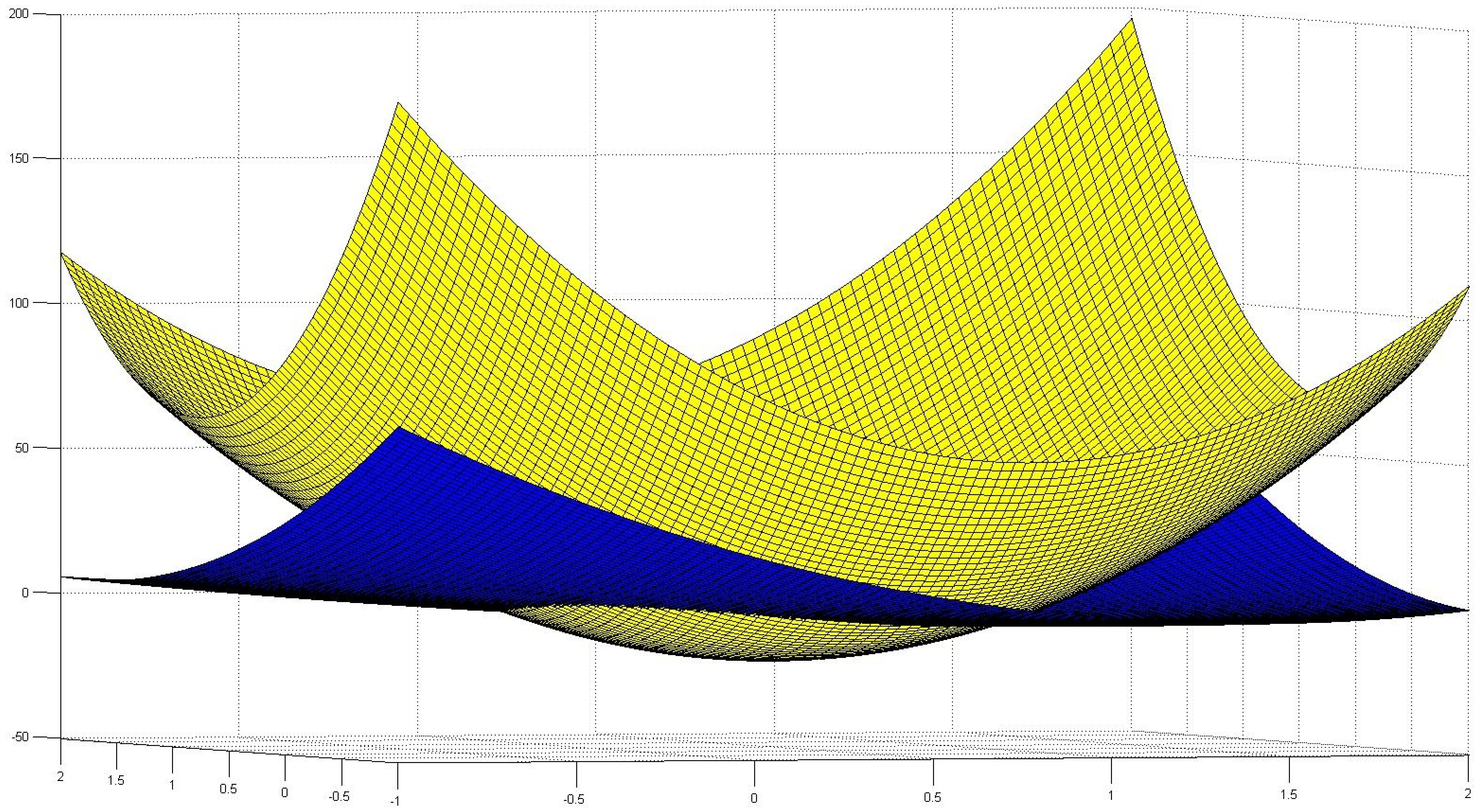
thus one can render the objective function convex.

Bound quality (all negative!):

	DJWhite	cplex 12.4	u	uv	# 0-1 vars
16x7	-11,949,628	-3,882,635	-3,687,670	-648,099	112
20x10	-13,133,687	-2,908,816	-2,218,027	-1,676,910	200
27x21	-68,190,327	-11,643,384	-9,204,537	---	567

FLATTENING A CONVEX FUNCTiON OF 0-1 VARIABLES

Combining QCR and CHR for convex quadratic pure 0-1 programming problems with linear constraints, A. Ahlatcioglu, M. Bussieck, M.Esen, M. Guignard, J.-H. Jagla, A. Meeraus, *Annals of OR* 199(1):33-49, 2012


200
150
100
50
0
-50
2
1.5
1
0.5
0
-0.5
-1
-0.5
0
0.5
1
1.5
2

An Other Bounding Approach:

The CHR (Convex Hull Relaxation) Algorithm

(For Pseudoconvex MINLP Problems With Linear Constraints)

Based on Primal Relaxation For Nonlinear Integer Problems

(Guignard 1994)

For a nonlinear integer programming problem

(P) $\quad \text{Min}_x \; \{ f(x) \mid Ax = b, Cx \le d, x \in X \}$

with a nonlinear (pseudo-)convex function f(x), one can construct a **primal relaxation**

(PR) $\quad \mathbf{Min_x \; \{ f(x) \mid Ax = b, \; x \in Co\{ x \in X \mid Cx \le d \}\}.}$

The constraint set is a polyhedron, and the objective function is still (pseudo-)convex.

Convex hull relaxation (CHR) for NLIP problems

(Albornoz 1998) (Ahlatcioglu, Guignard 2007)

For a nonlinear integer programming problem

(P) $\quad \text{Min}_x \quad \{ f(x) \mid x \in X \}$

with a nonlinear convex function f(x), and X su ch that Co(X) is a polyhedron.

One can construct a **primal relaxation**

(PR) $\quad \text{Min}_x \; \{ f(x) \mid x \in Co(X) \}$.

The constraint set is a polyhedron, and the objective function is still the same.

Special case: The CHR relaxation for linear constraints

(NLIP) $\quad$ **min {f(x) | $x \in Y$, Ax = b}**

where f(x) is a nonlinear convex function of $x \in R^n$,

A is an $m \times n$ constraint matrix, b is a resource vector in R^m ,

Y is a subset of R^n specifying integrality restrictions on x.

The **Convex Hull Relaxation** of (NLIP) is

(CHR) $\quad$ **min** $\quad$ **{f(x) | $x \in Co\{x \in Y \mid Ax = b\}$}.**

This relaxation is a primal relaxation, in the x-space. It is actually a relaxation that does not "relax" any true constraint.

(CHR) looks difficult because of the implicit formulation of the convex hull.

However one can solve the problem exactly by decomposing it into

a sub-problem and

a master problem,

(same as in Michelon and Maculan (1992) for the linear case of Lagrangean relaxation).

Possible methods:

Frank & Wolfe (1956),

Von Hohenbalken's Simplicial Decomposition (1977)

Hearn et al. 's Restricted SD (1987)

We use *restricted* simplicial decomposition.

To guarantee a global optimal solution of (CHR), several conditions must be satisfied:

(i) the feasible region must be compact and convex,

(ii) the objective function must be (pseudo-)convex, and

(iii) the constraints must be linear.

We solve a sequence of LIP subproblems and NLP master problems.

LOWER BOUND:

Simplicial Decomposition finds an optimal solution x* to (CHR).

This provides a lower bound on v(NLIP):

$$LB_{CHR} = f(x^*).$$

UPPPER BOUND

At each iteration k of the subproblem, an extreme point, y^{*k}, of the convex hull is found. It is an integer feasible point of (NLIP).

Each point y^{*k} yields an Upper Bound (UB) on the optimal value of (NLIP), and the best upper bound on v(NLIP) is

$$UB_{CHR} = \min f(y^{*1}), f(y^{*2}),\ldots, f(y^{*k})\}.$$

USED AS A HEURISTIC FOR NONCONVEX PROBLEMS:

multistart. Use different initial linearization points.

For CDAP: from 2 real crossdocks

For GQAP: with original data from Lee and Ma, and Moccia

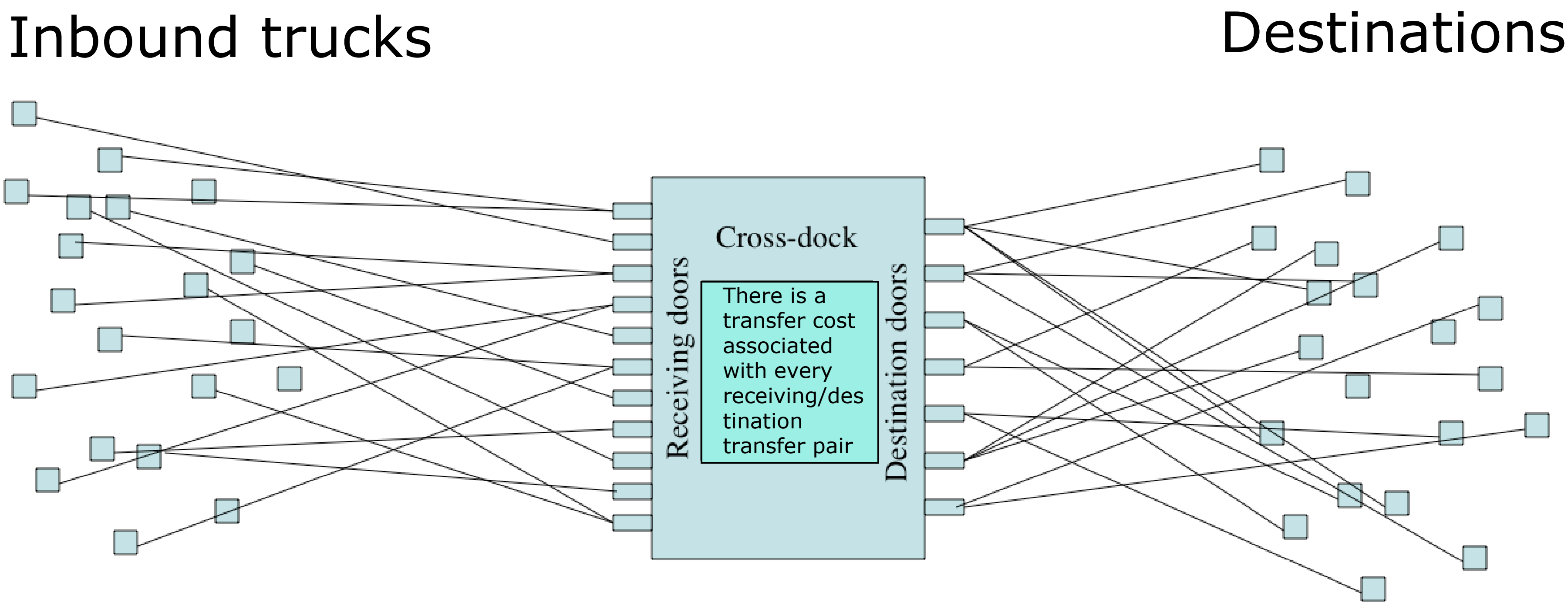
For QAP: QAPLib problems

Overall results: very fast. Average gap for known solutions less than 1%.

Example: Nonconvex GQAP

Instance	# 0-1 vars	OV	% gap	CH heuristic	Time (MIP, NLP, total)
pb16x7	112	2809870	0	2809870	9,2,11
pb-20-15-35	300	1471896	0	1471896	5,2,7
pb-20-15-55	300	1723638	0	1723638	9,6,15
pb-20-15-75	300	1953188	0	1953188	11,3,14
pb-30-06-95	180	5160920	0	5160920	11,6,17
pb-30-07-75	210	4383923	0	4383923	17,4,21
pb-30-08-55	240	3501695	0	3501695	3,2,5
pb-30-10-65	300	3620959	0	3620959	17,5,23
pb-30-20-35	600	3379359	0	3379359	16,6,22
pb-30-20-55	600	3593105	0	3593105	15,5,20
pb-30-20-75	600	4050938	0	4050938	11,3,14
pb-30-20-95	600	5710645	0.28	5726530	219,2,222
pb-35-15-35	525	4456670	0	4456670	16,8,23
pb-35-15-55	525	4639128	0	4639128	14,8,22
pb-35-15-75	525	6301723	0	6301723	16,9,26
pb-35-15-95	525	6670264	0	6670264	334,13,347

Application: Crossdocking



Formulation of the CDAP

Parameters

M - number of suppliers.
N - number of customers.
I - number of strip doors.
J - number of stack doors.
f_{mn} - the number of trips required by the material handling equipment to move items originating from supplier m to the stack door where freight destined for customer n is being consolidated.

d_{ij} - distance between strip door i and stack door j.

Decision Variables

$x_{mi} = 1$ if supplier m is assigned to strip door i, $x_{mi} = 0$ otherwise.
$y_{nj} = 1$ if customer n is assigned to stack door j, $y_{nj} = 0$ otherwise.

Formulation of the CDAP

$$\min \left\{ \text{cost} = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{m=1}^{M} \sum_{n=1}^{N} d_{ij} f_{mn} x_{mi} y_{nj} \right\}$$

$$s.t.$$

$$\sum_{i=1}^{I} x_{mi} = 1 \text{ for all } m$$

each inbound truck is assigned one strip door

$$\sum_{m=1}^{M} s_m x_{mi} \le S_i \text{ for all } i$$

capacity of strip door not exceeded

$$\sum_{j=1}^{J} y_{nj} = 1 \text{ for all } n$$

each destination is assigned one stack door

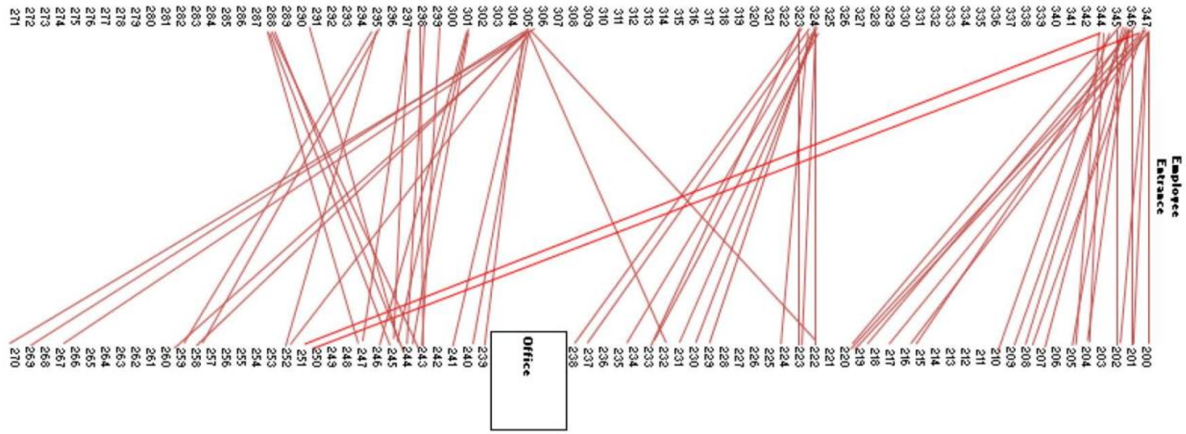
$$\sum_{n=1}^{N} r_n y_{nj} \le R_j \text{ for all } j$$

capacity of stack door not exceeded

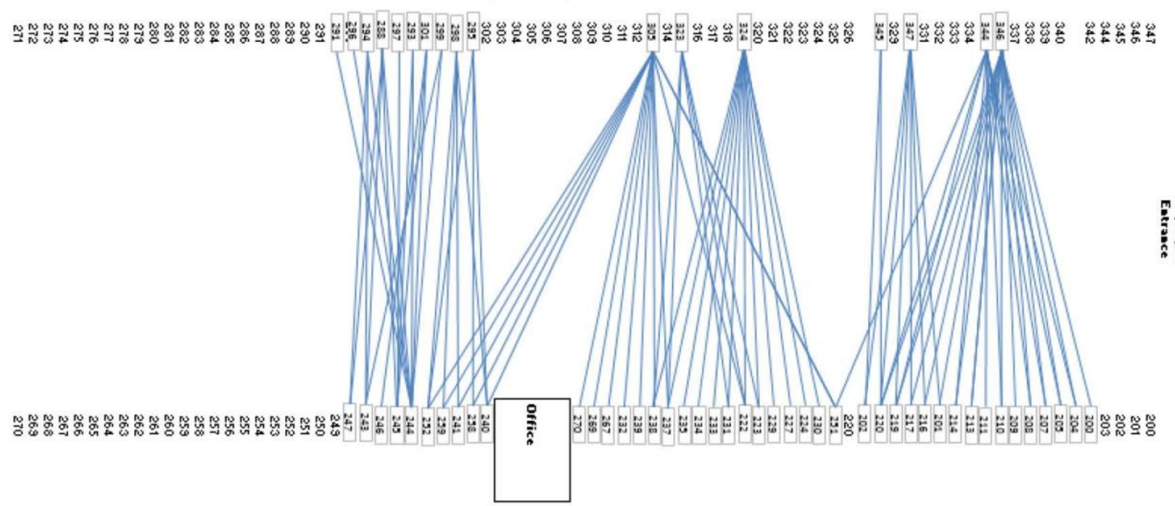
$$x_{mi}, y_{nj} \text{ are either 0 or 1}$$

CDAP DEMONSTRATION

Current operations



Modified operations



- For January 2012, the total daily cost could be reduced by utilizing the CDAP algorithm to assign the doors. By extrapolation on average, the CDAP cost is 11% lower than the Base Case cost, which translates to $10,000 of cost savings per month.

* The algorithm is very fast. It takes less than 50 seconds on a laptop to to solve a practical instance with 56 incoming trucks, 16 destinations, 41 inbound and 16 outbound doors. It can be used real time in the crossdock.

Thank You!