Computing Strong Bounds in Combinatorial Optimization

Hans D Mittelmann

School of Mathematical and Statistical Sciences
Arizona State University

Numerical Analysis and Optimization IV
Sultan Qaboos University
Muscat, Oman
January 2017
Outline

Computing lower bounds for QAPs
  The basic method and QAPLIB results
  Make method more efficient
  Solving a real-life problem from communications

Computing upper bounds for kissing numbers and binary codes
  High precision SDP bounds for kissing numbers
  High precision SDP bounds for binary codes
Outline

Computing lower bounds for QAPs
   The basic method and QAPLIB results
   Make method more efficient
   Solving a real-life problem from communications

Computing upper bounds for kissing numbers and binary codes
   High precision SDP bounds for kissing numbers
   High precision SDP bounds for binary codes
A QAP with four locations and facilities

Thickness of connections indicates level of flow

Optimal permutation: 2 1 4 3
A Mathematical Formulation of the Quadratic Assignment Problem

Mathematically, we can formulate the problem by defining two $n$ by $n$ matrices:

- a *flow matrix* $F$ whose $(i,j)$ element represents the flow between facilities $i$ and $j$,
- and a *distance matrix* $D$ whose $(i,j)$ element represents the distance between locations $i$ and $j$.

We represent an assignment by the vector $p$, which is a *permutation* of the numbers 1, 2, ..., $n$. $p(j)$ is the location to which facility $j$ is assigned.

With these definitions, the QAP can be written as

$$\min_{p \in \Pi} \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} d_{p(i)p(j)}$$
Computing Lower Bounds for QAPs via SDP relaxations and matrix splitting

Summary of results from


Work partly supported by AFOSR under grant FA9550-12-1-0153.
Quadratic Assignment Problem (QAP)

\[(QAP) \quad \min_{X \in \Pi} Tr(AXBX^T)\]

\(\Pi\): the set of permutation matrices.

Both \(A\) and \(B\) are symmetric with nonnegative elements

- First introduced by Koopmans and Beckmann [1957]
- Many applications from various fields: facility location, communication…[QAPLib]
- Hundreds of papers dedicated to QAPs as listed in a recent survey by Hahn et al. [2007]
Existing approaches for QAPs

- Heuristics [Hahn et al. 07]
  - Genetic algorithm, tabu search, simulated annealing

- Exact methods
  - Branch&bound, cutting planes [Pardalos et al. 97, Brixius and Anstreicher 01, Hahn et al. 01,02]
  - Needs to solve some relaxed problem in the process to get a lower bound
  - The efficacy of the relaxation model plays a crucial role
Cheap Relaxations of QAPs

- Cheap relaxations that can be solved quickly
  - GLB reformulation [Gilmore 62 and Lawler 63],
  - QP relaxation [Anstreicher and Brixius 01],
  - Spectral bound based on eigenvalues and projection [Hadley-Rendl-Wolkowicz 92]
  - Weak bounds have been observed, especially when n becomes large
  - Resulting in a huge number of nodes in a B&B approach
Expensive QAP Relaxations

- LP relaxation based on ILP reformulation [Adams and Sherali 86, 90, Hahn et al. 98, 01]
  - $z_{ijkl} = x_{ik} x_{jl}$ with extra constraints on $z$
- SDP relaxation based on matrix vectorization and Kronecker product [Zhao et al. 98, Rendl et al. 03]
  
  Let $x = \text{vector}(X)$ and apply the standard SDP relaxation to the matrix $xx^T$ with extra constraints on the matrix elements.

- Tight bound but involves intensive computation
  - Out of the question for QAP instances of size $n=50$
Efficacy VS Tightness

- **RLT**: reformul. based on lifting
- **ML**: matrix lifting

Diagram:
- Efficacy of the model
- GLB
- QP
- Eig
- Grey area
- ML
- RLT
- SDP
- Tightness of the bound
Motivation and Observation

- **Motivation:**
  - find cheap relaxations that yield strong bounds.

- **Observations:**
  - Most relaxations are based on the binary structure of the matrix elements, not the algebraic feature of the permutation matrix itself!
  - Specific QAPs arising from data mining have positive semidefinite matrices and large scale problems (n=1000s) have been solved based on SDP approaches

\[ B \succeq 0 \iff XBX^T \succeq 0 \]
Matrix Splitting

- What to do when $B$ is not PSD?
  - Split the matrix into two parts $OH, AH$

$$B = B^+ - B^-, \quad B^+, B^- \succeq 0.$$ 

Both $XB^+X^T$ and $XB^-X^T$ are positive semidefinite. Let $Y^+ = XB^+X^T, Y^- = XB^-X^T$, we have

$$Y^+, Y^- \succeq 0.$$
New SDP relaxations

Let $e$ be the all 1 vector and $\min(B)$ the minimum element of $B$. Using the properties of $X$, we derive the following relaxation

$$
\min \quad \text{Tr}(A(Y^+ - Y^-)) \\
\text{s.t.} \quad Y^+ e = XB^+ e, \quad Y^- e = XB^- e; \\
\text{diag}(Y^+) = X \text{diag}(B^+), \quad Y^+ \geq \min(B^+); \\
\text{diag}(Y^-) = X \text{diag}(B^-), \quad Y^- \geq \min(B^-); \\
Y^+ - XB^+ X^T \succeq 0, \quad Y^- - XB^- X^T \succeq 0; \\
X e = X^T e = e, \quad X \succeq 0.
$$
One Theorem

- **Theorem:** The lower bound provided by the new SDP relaxation is always tighter than the bound derived by SDP relaxation based on matrix lifting in [Ding and Wolkowicz 06].
  - As observed in Ding-Wolkowicz paper, such a bound is comparable to the strongest SDP bounds.
Improvement and simplification

- We could swap A and B to derive a more complex SDP relaxation;
- Symmetries can be explored to improve the model;
- We could simplify the SDP constraints to

\[ Y^+, Y^- \succeq 0. \]

- Leading to certain speedup in the solving process, while without much loss of tightness of the bound
New Splitting Schemes: I

**Definition:** We call the matrix splitting $B = B^+ - B^-$ an orthogonal splitting if $B^+$ and $B^-$ are orthogonal to each other

- Can be derived by using the singular value decomposition of $B$ directly
- Additional constraints can be added based on the orthogonality
CVX script for basic model

e = ones(n,1);
E = e*e';
I = eye(n);
[V,D] = eig(B);
Dp = max(D, zeros(n));
Dm = max(Dp - D, zeros(n));
Bp = V*Dp*V';
Bm = V*Dm*V';
Dp = sqrtm(Dp);
Dm = sqrtm(Dm);
Rp = V*Dp;
Rm = V*Dm;
cvx_begin
  variable X(n,n)
  variable Yp(n,n) symmetric
  variable Ym(n,n) symmetric
  variable Zp(n,n)
  variable Zm(n,n)
  minimize( trace(A*(Yp-Ym)) )
  subject to
    diag(Yp) == X*diag(Bp);
    diag(Ym) == X*diag(Bm);
    Yp*e == X*Bp*e;
    Ym*e == X*Bm*e;
    tril(Yp,-1) - tril(Ym,-1) >= min(min(B));
    tril(Yp,-1) >= min(min(tril(Bp,-1)));
    tril(Ym,-1) >= min(min(tril(Bm,-1)));
\texttt{\% norm(Yp + Ym,'fro') \leq \text{norm}(B,'fro');}
\texttt{Zp == X*Rp;}
\texttt{Zm == X*Rm; lambda_min([I,Zp';Zp,Yp]) \geq 0; lambda_min([I,Zm';Zm,Ym]) \geq 0;}
\texttt{\% lambda_min(Yp) >= 0; lambda_min(Ym) >= 0;}
\texttt{X \geq 0; sum(X) == 1; sum(X') == 1; cvx_end}
## Numerical Results: I

<table>
<thead>
<tr>
<th>Prob</th>
<th>QAP_{L-S1}</th>
<th>CPU</th>
<th>QAP_{S-L2}</th>
<th>CPU</th>
<th>Model-1</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nug25</td>
<td>18%</td>
<td>4665</td>
<td>3%</td>
<td>8914</td>
<td>11%</td>
<td>23s</td>
</tr>
<tr>
<td>Nug30</td>
<td>22%</td>
<td>11321</td>
<td>1%</td>
<td>26347</td>
<td>10%</td>
<td>72s</td>
</tr>
<tr>
<td>Tail30b</td>
<td>78%</td>
<td>12172</td>
<td>18%</td>
<td>50582</td>
<td>15%</td>
<td>161s</td>
</tr>
<tr>
<td>Tail35b</td>
<td>65%</td>
<td>24440</td>
<td>15%</td>
<td>141300</td>
<td>22%</td>
<td>322s</td>
</tr>
<tr>
<td>Tail40b</td>
<td>74%</td>
<td>43181</td>
<td>15%</td>
<td>330773</td>
<td>16%</td>
<td>763s</td>
</tr>
</tbody>
</table>

The relative gap is listed for comparison on tightness
### Numerical Results: 2

<table>
<thead>
<tr>
<th>Problem</th>
<th>Tail50b</th>
<th>Tail60b</th>
<th>Tail64C</th>
<th>Tail80b</th>
<th>Tail100b</th>
<th>Tail150b</th>
</tr>
</thead>
<tbody>
<tr>
<td>New gap</td>
<td>18.4%</td>
<td>22.2%</td>
<td>2.4%</td>
<td>18.4%</td>
<td>22%</td>
<td>13.6%</td>
</tr>
<tr>
<td>Old gap</td>
<td>91.2%</td>
<td>91.8%</td>
<td>51.7%</td>
<td>89.1%</td>
<td>86.3%</td>
<td>87.4%</td>
</tr>
</tbody>
</table>

All the above problems have been solved within 40 minutes. Strong bounds have been obtained for QAPs of size up to n=256.
From QAPLIB: **E.D. Taillard [Taillard:91,Taillard:94]**

<table>
<thead>
<tr>
<th>name</th>
<th>n</th>
<th>feas.sol. (method)</th>
<th>permutation/bound (method)</th>
<th>gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tai40b</td>
<td>40</td>
<td>637250948 (Ro-TS)</td>
<td>544404685 (SDRMS)</td>
<td>14.57%</td>
</tr>
<tr>
<td>Tai50a</td>
<td>50</td>
<td>4938796 (ITS)</td>
<td>4390920 (L&amp;P)</td>
<td>11.09%</td>
</tr>
<tr>
<td>Tai50b</td>
<td>50</td>
<td>458821517 (Ro-TS)</td>
<td>381474057 (SDRMS)</td>
<td>16.86%</td>
</tr>
<tr>
<td>Tai60a</td>
<td>60</td>
<td>7205962 (TS-2)</td>
<td>5555095 (GLB)</td>
<td>22.91%</td>
</tr>
<tr>
<td>Tai60b</td>
<td>60</td>
<td>608215054 (Ro-TS)</td>
<td>494776302 (SDRMS)</td>
<td>18.65%</td>
</tr>
<tr>
<td>Tai64c</td>
<td>64</td>
<td>1855928 (Ro-TS)</td>
<td>1812779 (SDRMS)</td>
<td>2.32%</td>
</tr>
<tr>
<td>Tai80a</td>
<td>80</td>
<td>13515450 (ITS)</td>
<td>10329674 (GLB)</td>
<td>23.57%</td>
</tr>
<tr>
<td>Tai80b</td>
<td>80</td>
<td>818415043 (Ro-TS)</td>
<td>683526345 (SDRMS)</td>
<td>16.48%</td>
</tr>
<tr>
<td>Tai100a</td>
<td>100</td>
<td>21052466 (ITS)</td>
<td>15824355 (GLB)</td>
<td>24.86%</td>
</tr>
<tr>
<td>Tai100b</td>
<td>100</td>
<td>1185996137 (Ro-TS)</td>
<td>961844607 (SDRMS)</td>
<td>18.90%</td>
</tr>
<tr>
<td>Tai150b</td>
<td>150</td>
<td>498896643 (GEN-3)</td>
<td>435738380 (SDRMS)</td>
<td>12.66%</td>
</tr>
<tr>
<td>Tai256c</td>
<td>256</td>
<td>44759294 (ANT)</td>
<td>43849646 (SDRMS)</td>
<td>2.03%</td>
</tr>
</tbody>
</table>
Outline

Computing lower bounds for QAPs
  The basic method and QAPLIB results
  Make method more efficient
  Solving a real-life problem from communications

Computing upper bounds for kissing numbers and binary codes
  High precision SDP bounds for kissing numbers
  High precision SDP bounds for binary codes
Desirable Improvements of Our Approach
there are four major ones

- Reduce memory consumption
- Cut CPU time for very large problems
- Make bounds guaranteed lower bounds
- Solve large, real-life problem

All issues were addressed successfully.
Reduce memory consumption
w/o sacrificing quality of bounds

Reason for the large memory consumption:

- Solution of the SDP relaxations via interior point methods that employ direct numerical algebra (Cholesky decomposition)
- Problems are not necessarily sparse
Reduce memory consumption

go iterative!

Key paper for both resource issues:


Code available from K.C. Toh

This needed to be combined with CVX in place of SDPT3/SeDuMi
Cut CPU time for very large problems

go iterative!

The Newton-cg method is also faster, requiring fewer and cheaper iterations compared with the IPM

Example to be given with real-life problem in next section
Compute guaranteed bounds
in case SDP solved incompletely

A key paper is here:


- Both rigorous lower and upper bounds computed
- Employs interval-arithmetic ideas
- Minor post-processing effort
Outline

Computing lower bounds for QAPs
  The basic method and QAPLIB results
  Make method more efficient
  Solving a real-life problem from communications

Computing upper bounds for kissing numbers and binary codes
  High precision SDP bounds for kissing numbers
  High precision SDP bounds for binary codes
A real-life problem from communications
optimal index assignment

- In communication systems index assignment is the problem of labeling source codewords by binary integer numbers (channel codewords)

- For a source code of fixed integer rate $n$, there are
  $$\frac{2^n!}{2^n \times n!} = \frac{(2^n-1)!}{n!}$$
  distinct index assignments

- In the presence of channel errors the overall system performance does depend on the index assignment

- Channel-optimized index assignment of source codewords is the simplest way of improving the system error resilience
A real-life problem from communications

optimal index assignment

- Source codewords that are distant from one another in code space should be indexed by binary numbers of large Hamming distances.

- A basic element of a signal compression and communication system is the quantizer $Q$, either scalar or vector.

- We focus on index assignment of vector quantizers (VQ) for their superior source coding performance.

- A vector quantizer $Q : \mathbb{R}^d \rightarrow \{c_1, c_2, \cdots, c_N\}$ maps a continuous source vector $x \in \mathbb{R}^d$ to a codeword $c_j \in \mathbb{R}^d$ in the VQ codebook $C = \{c_1, c_2, \cdots, c_N\}$ by the nearest neighbor rule.

- The index $i$ rather than the codeword $c_j$ itself is transmitted via the channel. Upon receiving $i$ correctly, the VQ decoder can reconstruct $x$ from $c_i$ by inverse quantizer mapping $Q^{-1}$. 
A real-life problem from communications

optimal index assignment

- Typically, the size $N$ of the codebook $\mathcal{C}$ is made an integer power of two, $N = 2^n$ so that the codeword index $i$ is a binary number of $n$ bits. An index assignment of $\mathcal{C}$ is a bijection map $\pi : \mathcal{C} \leftrightarrow \{0, 1\}^n$.

- If in the event of a transmission error an index $\pi(c_i)$ is received as $\pi(c_j)$, an input vector $x$ such that $w_i = Q(x)$ will be reconstructed as $w_j$, incurring an extra channel distortion $d(c_i, c_j)$ that does depend on index assignment $\pi$.

- Let $P(j|i)$ be the probability of transmitting index $i$ but receiving index $j$, and $P(c_i)$ be the prior probability of the codeword $c_i$. 
A real-life problem from communications

optimal index assignment

- Given an index assignment \( \pi \), the expected channel distortion is

\[
\bar{d}_\pi = \sum_{i=1}^{N} P(c_i) \sum_{j=1}^{N} P(\pi(w_j)|\pi(c_i))d(c_i,c_j)
\]

- Adopting the common probability model of binary symmetric channel (BSC), we have

\[
P(\pi(w_j)|\pi(c_i)) = (1-p)^n- h(\pi(w_j),\pi(c_i)) p^{h(\pi(w_j),\pi(c_i))}
\]

\(p\) the BSC crossover probability, and \(h(\cdot, \cdot)\) Hamming distance.

- To minimize the expected channel distortion \(\bar{d}_\pi\) one needs to find

an index assignment defined by the objective function

\[
\pi_* = \arg \min_{\pi} \sum_{i=1}^{N} P(c_i) \sum_{j=1}^{N} (1-p)^{n-h(\pi(w_j),\pi(c_i))} p^{h(\pi(w_j),\pi(c_i))} d(c_i,c_j).
\]
A real-life problem from communications

optimal index assignment

For convenience, we rewrite in matrix form. Let

- \( W = \text{diag}(P(c_1), P(c_2), \cdots, P(c_N)) \) be the diagonal matrix consisting of prior probabilities of the VQ codewords,

- \( B = \{(1 - p)^{n-h(i,j)} p^{h(i,j)} \}_{1 \leq i \leq N, 1 \leq j \leq N} \) be the symmetric matrix whose elements \( B(i, j) \) are the codeword transition probabilities \( P(\pi(w_j)|\pi(c_i)) \) due to BSC bit errors of probability \( p \),

- \( D = \{d(c_i, c_j)\}_{1 \leq i \leq N, 1 \leq j \leq N} \) be the symmetric distance matrix between pairs of codewords, and

- \( X \) be the \( N \times N \) permutation matrix to specify \( \pi \).
A real-life problem from communications
optimal index assignment

Then,

\[ d_{\pi} = \sum_{i=1}^{N} P(c_i) \sum_{j=1}^{N} \{XBX^T\}_{i,j} d(c_i, c_j) \]

\[ = \text{trace} \left( WXBX^T D \right) \]

\[ = \text{trace} \left( DWXBX^T \right) \]

Symmetrizing

\[ \tilde{D} = DW + D^T W^T. \]

we finally have

\[ d_{\pi} = \frac{1}{2} \text{trace} \left( \tilde{D} XBX^T \right) \]

A quadratic assignment problem!
A real-life problem from communications

optimal index assignment

- Due to the wide use of VQ in image coding, we present a case study on image VQ index assignment. A training set of 18 natural images is used to design 16-dimensional vector quantizers of various fixed integer rates $n$.

- We consider the general case (multiple bit errors) or we assume the BSC channel crossover probability $p$ to be sufficiently small and consider only one bit errors. This simplifies the codeword transition probability expression to

$$P(\pi(w_j)|\pi(c_i)) = (1 - p)^{n-1} p$$

and consequently,

$$B = (1 - p)^{n-1} pA$$

where $A$ is the adjacency matrix of the $n$-dimensional hypercube.
Table: Different lower and upper bounds

### One bit errors

<table>
<thead>
<tr>
<th>n</th>
<th>PB</th>
<th>GLB</th>
<th>SDP</th>
<th>ILS</th>
<th>gap</th>
<th>NBC</th>
<th>gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>102156</td>
<td>84784</td>
<td>304551</td>
<td>358984</td>
<td>0.71 dB</td>
<td>470147</td>
<td>1.88 dB</td>
</tr>
<tr>
<td>6</td>
<td>42807</td>
<td>58617</td>
<td>289883</td>
<td>349337</td>
<td>1.06 dB</td>
<td>530994</td>
<td>2.62 dB</td>
</tr>
<tr>
<td>7</td>
<td>&lt; 0</td>
<td>45503</td>
<td>242657</td>
<td>334360</td>
<td>1.39 dB</td>
<td>592534</td>
<td>3.89 dB</td>
</tr>
<tr>
<td>8</td>
<td>&lt; 0</td>
<td>43942</td>
<td>199959</td>
<td>294756</td>
<td>1.69 dB</td>
<td>650936</td>
<td>5.14 dB</td>
</tr>
<tr>
<td>9</td>
<td>&lt; 0</td>
<td>38156</td>
<td>193271</td>
<td>291314</td>
<td>1.78 dB</td>
<td>719776</td>
<td>5.71 dB</td>
</tr>
</tbody>
</table>

### Multiple bit errors

<table>
<thead>
<tr>
<th>n</th>
<th>PB</th>
<th>GLB</th>
<th>SDP</th>
<th>ILS</th>
<th>gap</th>
<th>NBC</th>
<th>gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1146</td>
<td>884</td>
<td>2891</td>
<td>3608</td>
<td>0.96 dB</td>
<td>4685</td>
<td>2.10 dB</td>
</tr>
<tr>
<td>6</td>
<td>614</td>
<td>819</td>
<td>2834</td>
<td>3734</td>
<td>1.20 dB</td>
<td>5273</td>
<td>2.70 dB</td>
</tr>
<tr>
<td>7</td>
<td>&lt; 0</td>
<td>484</td>
<td>2412</td>
<td>3347</td>
<td>1.42 dB</td>
<td>5871</td>
<td>3.86 dB</td>
</tr>
<tr>
<td>8</td>
<td>&lt; 0</td>
<td>465</td>
<td>2020</td>
<td>2984</td>
<td>1.69 dB</td>
<td>6424</td>
<td>5.02 dB</td>
</tr>
<tr>
<td>9</td>
<td>&lt; 0</td>
<td>416</td>
<td>1931</td>
<td>2937</td>
<td>1.82 dB</td>
<td>7072</td>
<td>5.64 dB</td>
</tr>
</tbody>
</table>
Outline

Computing lower bounds for QAPs
   The basic method and QAPLIB results
   Make method more efficient
   Solving a real-life problem from communications

Computing upper bounds for kissing numbers and binary codes
   High precision SDP bounds for kissing numbers
   High precision SDP bounds for binary codes
Kissing Number Problem

In geometry, the *kissing number* $\tau_n$ is the maximum number of spheres of radius 1 that can simultaneously touch the unit sphere $S^{n-1}$ in $n$-dimensional Euclidean space.

1694  12 or 13?  
Sir Isaac Newton  
David Gregory
What is known in low dimensions?

- $\tau_1 = 2$, $\tau_2 = 6$ is trivial.

- $\tau_3 = 12$, Schütte and van der Waerden (1953).

- $\tau_8 = 240$, $\tau_{24} = 196560$, Odlyzko, Sloane, and Levenshtein (1979).

- $\tau_4 = 24$, Musin (2003).

---

**Goal:**
Find good upper bounds using semidefinite programming.
Which SDP bounds have been found?

- Previous best upper bounds from 1979-2007
- In 2008 paper by C. Bachoc and F. Vallentin with moderately improved bounds up to dimension 10.
- Use of regular (double precision) computation prevented better results.
- Use of multiple precision SDP solvers (SDPA/CSDP)
- Computations tedious/tricky
Which SDP needs to be solved?

- kissing number is stability number of infinite graph
- stability number is bounded by Lovasz theta number
- Lovasz theta number is solution of SDP
- graph $\Gamma(S^{n-1}, (0, \pi/3))$ on vertex set $S^{n-1} = \{x \in \mathbb{R}^n : x \cdot x = 1\}$
- edges when angular distance $< \pi/3$, inner product $> 1/2$
- bounds from this SDP strengthened using symmetries and Lasserre hierarchy

\[
v'(\Gamma(S^{n-1}, (0, \pi/3))) = \inf \left\{ \lambda : \begin{array}{l}
K \in \mathcal{C}(S^{n-1} \times S^{n-1})_{\geq 0}, \\
K(x, x) = \lambda - 1, \text{ for all } x \in S^{n-1}, \\
K(x, y) \leq -1, \text{ for all } x, y \in S^{n-1} \\
\text{with } x \cdot y \leq 1/2 \end{array} \right\},
\]

- $\mathcal{C}(S^{n-1} \times S^{n-1})_{\geq 0}$ cone of positive definite Hilbert-Schmidt kernels
Our improved bounds as recorded in Wikipedia

<table>
<thead>
<tr>
<th>n</th>
<th>lower</th>
<th>old bd</th>
<th>year</th>
<th>our bd</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>40</td>
<td>45</td>
<td>2008</td>
<td>44</td>
</tr>
<tr>
<td>7</td>
<td>126</td>
<td>135</td>
<td>2008</td>
<td>134</td>
</tr>
<tr>
<td>9</td>
<td>306</td>
<td>366</td>
<td>2008</td>
<td>364</td>
</tr>
<tr>
<td>10</td>
<td>500</td>
<td>567</td>
<td>2008</td>
<td>554</td>
</tr>
<tr>
<td>11</td>
<td>582</td>
<td>915</td>
<td>1979</td>
<td>870</td>
</tr>
<tr>
<td>12</td>
<td>840</td>
<td>1416</td>
<td>1979</td>
<td>1357</td>
</tr>
<tr>
<td>13</td>
<td>1130</td>
<td>2233</td>
<td>1979</td>
<td>2069</td>
</tr>
<tr>
<td>14</td>
<td>1582</td>
<td>3492</td>
<td>1979</td>
<td>3183</td>
</tr>
<tr>
<td>15</td>
<td>2564</td>
<td>5431</td>
<td>1979</td>
<td>4866</td>
</tr>
</tbody>
</table>

==============================================
Our improved bounds as recorded in Wikipedia

<table>
<thead>
<tr>
<th>n</th>
<th>lower</th>
<th>old bd</th>
<th>year</th>
<th>our bd</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>4320</td>
<td>8312</td>
<td>2007</td>
<td>7355</td>
</tr>
<tr>
<td>17</td>
<td>5346</td>
<td>12210</td>
<td>2007</td>
<td>11072</td>
</tr>
<tr>
<td>18</td>
<td>7398</td>
<td>17877</td>
<td>1979</td>
<td>16572</td>
</tr>
<tr>
<td>19</td>
<td>10668</td>
<td>25900</td>
<td>1994</td>
<td>24812</td>
</tr>
<tr>
<td>20</td>
<td>17400</td>
<td>37974</td>
<td>1979</td>
<td>36764</td>
</tr>
<tr>
<td>21</td>
<td>27720</td>
<td>56851</td>
<td>1994</td>
<td>54584</td>
</tr>
<tr>
<td>22</td>
<td>49896</td>
<td>86537</td>
<td>1979</td>
<td>82340</td>
</tr>
<tr>
<td>23</td>
<td>93150</td>
<td>128095</td>
<td>1994</td>
<td>124416</td>
</tr>
</tbody>
</table>

---------------------------------------------------------------------------------------------------
From Wikipedia

Lower and our upper bounds

![Graph showing the relationship between Kissing Number and Dimension.](image-url)
Computing lower bounds for QAPs
   The basic method and QAPLIB results
   Make method more efficient
   Solving a real-life problem from communications

Computing upper bounds for kissing numbers and binary codes
   High precision SDP bounds for kissing numbers
   High precision SDP bounds for binary codes
High precision SDP bounds for binary codes

D. C. Gijswijt, H. D. Mittelmann, and A. Schrijver,
*Semidefinite code bounds based on quadruple distances*,

- $A(n,d)$ is maximum number of binary words of length $n$, any two having Hamming distance at least $d$
- Classical *Delsarte* bound yields huge SDP which can be reduced to small LP
- In 2005 *Schrijver* generalized to SDPs of sets of size at most 3
- They can be reduced to small SDPs with block-diagonalization
- New work generalizes to *quadruples* of words. Reduced SDPs are still large
- They are ill-conditioned and require high precision
<table>
<thead>
<tr>
<th>( n )</th>
<th>( d )</th>
<th>known lower bound</th>
<th>known upper bound</th>
<th>new upper bound</th>
<th>( A_4(n, d) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>6</td>
<td>512</td>
<td>680</td>
<td>673</td>
<td>673.005</td>
</tr>
<tr>
<td>19</td>
<td>6</td>
<td>1024</td>
<td>1280</td>
<td>1237</td>
<td>1237.939</td>
</tr>
<tr>
<td>20</td>
<td>6</td>
<td>2048</td>
<td>2372</td>
<td>2279</td>
<td>2279.758</td>
</tr>
<tr>
<td>23</td>
<td>6</td>
<td>8192</td>
<td>13766</td>
<td>13674</td>
<td>13674.962</td>
</tr>
<tr>
<td>19</td>
<td>8</td>
<td>128</td>
<td>142</td>
<td>135</td>
<td>135.710</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>256</td>
<td>274</td>
<td>256</td>
<td>256.000</td>
</tr>
<tr>
<td>25</td>
<td>8</td>
<td>4096</td>
<td>5477</td>
<td>5421</td>
<td>5421.499</td>
</tr>
<tr>
<td>26</td>
<td>8</td>
<td>4096</td>
<td>9672</td>
<td>9275</td>
<td>9275.544</td>
</tr>
<tr>
<td>27</td>
<td>8</td>
<td>8192</td>
<td>17768</td>
<td>17099</td>
<td>17099.644</td>
</tr>
<tr>
<td>21</td>
<td>10</td>
<td>42</td>
<td>48</td>
<td>47</td>
<td>47.007</td>
</tr>
<tr>
<td>22</td>
<td>10</td>
<td>64</td>
<td>87</td>
<td>84</td>
<td>84.421</td>
</tr>
<tr>
<td>24</td>
<td>10</td>
<td>128</td>
<td>280</td>
<td>268</td>
<td>268.812</td>
</tr>
<tr>
<td>25</td>
<td>10</td>
<td>192</td>
<td>503</td>
<td>466</td>
<td>466.809</td>
</tr>
<tr>
<td>26</td>
<td>10</td>
<td>384</td>
<td>886</td>
<td>836</td>
<td>836.669</td>
</tr>
<tr>
<td>27</td>
<td>10</td>
<td>512</td>
<td>1764</td>
<td>1585</td>
<td>1585.071</td>
</tr>
<tr>
<td>28</td>
<td>10</td>
<td>1024</td>
<td>3170</td>
<td>2817</td>
<td>2817.313</td>
</tr>
<tr>
<td>25</td>
<td>12</td>
<td>52</td>
<td>56</td>
<td>55</td>
<td>55.595</td>
</tr>
<tr>
<td>26</td>
<td>12</td>
<td>64</td>
<td>98</td>
<td>96</td>
<td>96.892</td>
</tr>
</tbody>
</table>
THE END
Thank you for your attention

Questions or Remarks?