A new SDP relaxation for the Quadratic Assignment Problem

Introducing Cut Pseudo Bases

Maximilian John, Andreas Karrenbauer

16.05.2016

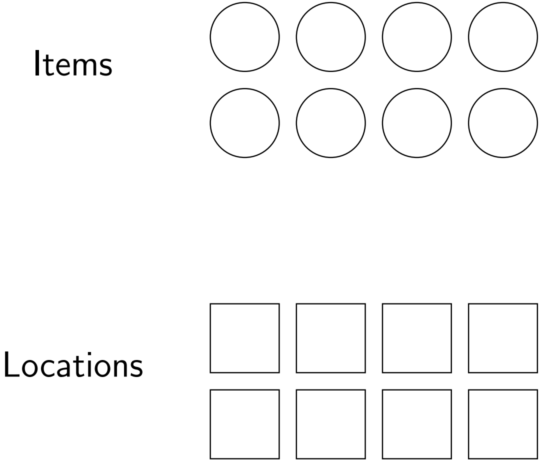
max planck institut informatik

SAARLAND UNIVERSITY
GRADUATE SCHOOL OF COMPUTER SCIENCE

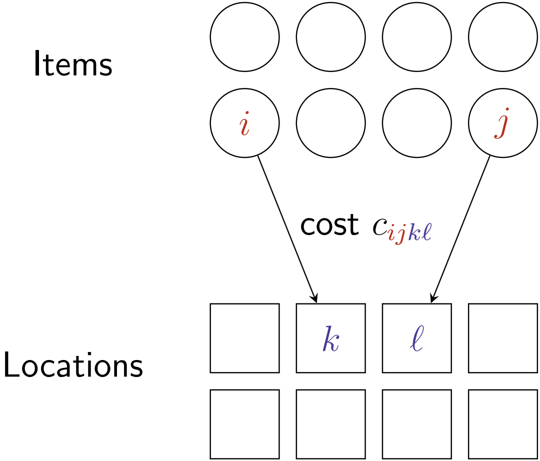
Max Planck Center for visual computing and communication

The quadratic assignment problem
Applications

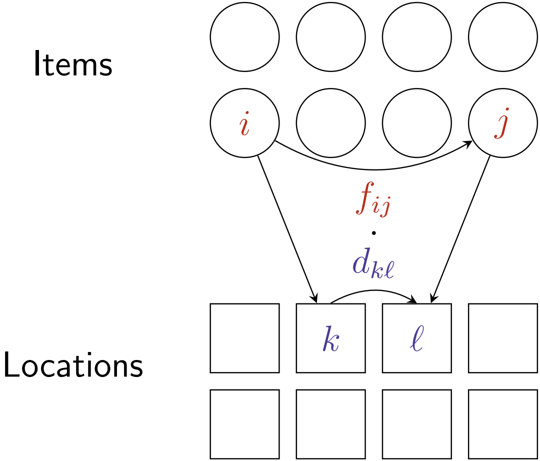
The quadratic assignment problem



The quadratic assignment problem



The quadratic assignment problem



Formal definition

As integer program

$$
\begin{aligned}
\text{minimize} \quad & \sum_{ijk\ell} c_{ijk\ell} x_{ik} x_{j\ell} \\
\text{subject to} \quad & \sum_{i} x_{ik} = 1 \\
& \sum_{k} x_{ik} = 1 \\
& x_{ik} \in \{0, 1\}
\end{aligned}
$$

- In many cases: $c_{ijk\ell} = f_{ij} d_{k\ell}$
- F - flow (probability) matrix
- D - distance matrix

Various relaxations

Various relaxations

- Kaufman and Broeckx
  - \+ Very fast to compute and solve
  - – Bad lower bounds
- Reformulation-linearization techniques (RLT)
  - \+ Good lower bounds
  - – $\mathcal{O}(n^4)$ variables, hard to solve
- Various LP relaxations (Gilmore-Lawler, Padberg-Rinaldi, ...)
- Various SDP relaxations (Rendl et al., ...)

Our challenge

- Trade-off between efficiency and quality of lower bounds

Standard branch and bound

Branching on assignment variables ($x_{ik} \in \{0, 1\}$)

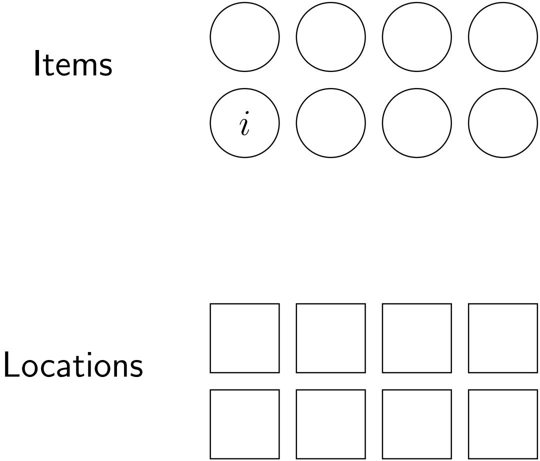
- $x_{ik} = 1$
  - Fix item i to location k
  - Strong decision
- $x_{ik} = 0$
  - Keep away item i from location k
  - Very weak decision

$\Rightarrow$ Highly imbalanced branching tree

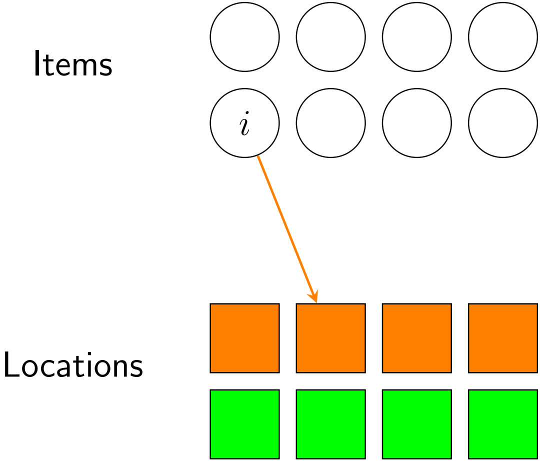
New approach

Branch on cuts

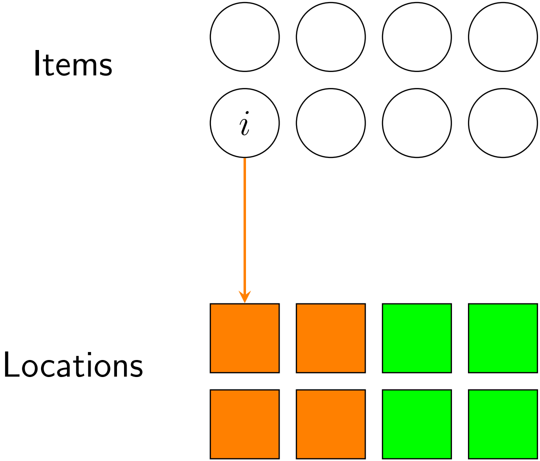
Assign items to cuts



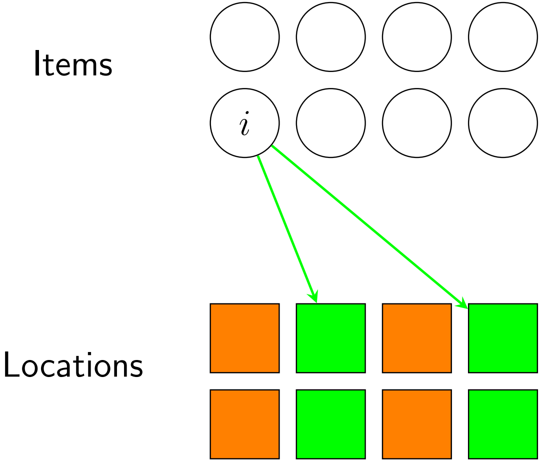
Assign items to cuts



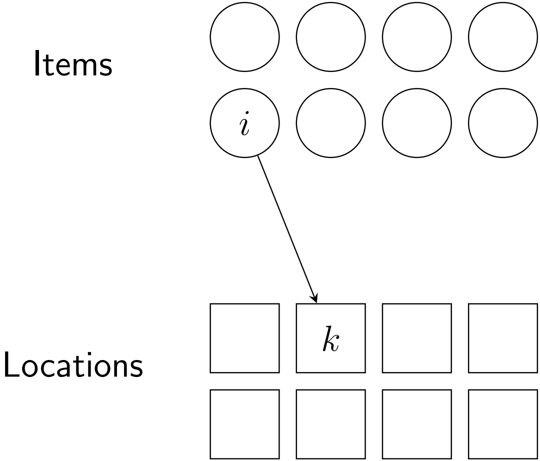
Assign items to cuts



Assign items to cuts



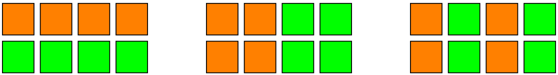
Assign items to cuts



Introduce cuts

- Bipartition the keyboard into cuts such that
    - all cuts are balanced
    - all locations (singletons) are linear combinations of cuts
    - the number of cuts is minimal

$\Rightarrow$ *Cut pseudo-base*



Pseudo-base examples

Binary decomposition cuts

- Idea: Enumerate $n = 2^B$ locations from 0 to $n-1$
- Each of the B bits defines a cut ($\mathbf{0}$-side and $\mathbf{1}$-side)
- Formally: Introduce $z_i^b \in \{0, 1\}$
  - $\Rightarrow$ indicating the side of i in cut b
- $x_{ik} = 1 \Leftrightarrow \forall b : \quad z_i^b = k[b], \;\; k[b] = b$-th bit of k

000	**0**01	**0**10	**0**11
100	**1**01	**1**10	**1**11

0**0**0	0**0**1	0**1**0	0**1**1
1**0**0	1**0**1	1**1**0	1**1**1

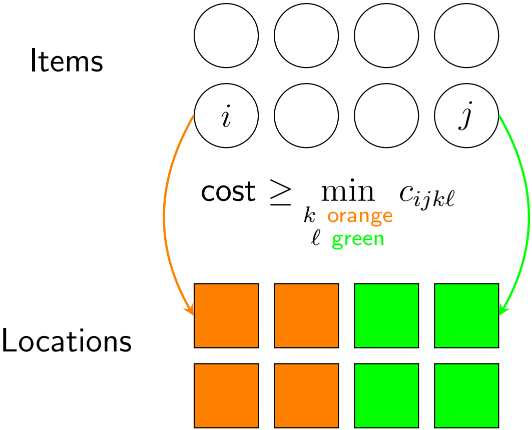
00**0**	00**1**	01**0**	01**1**
10**0**	10**1**	11**0**	11**1**

Goal: Derive lower bounds for the QAP

Approach

- Introduce cut variables
- Estimate cost for each pair of cut variables
- Eliminate assignment variables
- Obtain a quadratic formulation with $n\log_2(n)$ variables

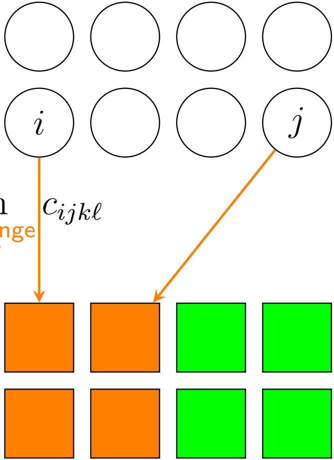
$\Rightarrow$ Relax new formulation to an SDP


Items
i
j
cost $\geq \min_{\substack{k \text{ orange} \\ \ell \text{ green}}} c_{ijk\ell}$
Locations

Items

$$\text{cost} \geq \min_{\substack{k,\ell\ \textcolor{orange}{\text{orange}} \\ k \neq \ell}} c_{ijk\ell}$$

Locations



Introduce cut variables

Reminder

$x_{ik} = 1 \Leftrightarrow \forall b: \quad z_i^b = k[b], \quad k[b] =$ b-th bit of k

Insert a factor 1

$$\sum_{i,j} [z_i z_j + z_i(1 - z_j) + (1 - z_i)z_j + (1 - z_i)(1 - z_j)] \sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell}$$

Introduce cut variables

Reminder

$x_{ik} = 1 \Leftrightarrow \forall b: \quad z_i^b = k[b], \quad k[b] =$ b-th bit of k

Insert a factor 1

$$\sum_{i,j} [z_i z_j + z_i(1 - z_j) + (1 - z_i)z_j + (1 - z_i)(1 - z_j)] \sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell}$$
$$\geq \sum_{i,j} z_i z_j \quad \cdot \min\{\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(11)}\}$$

Introduce cut variables

Reminder

$x_{ik} = 1 \Leftrightarrow \forall b : \quad z_i^b = k[b], \quad k[b] =$ b-th bit of k

Insert a factor 1

$$\sum_{i,j} [z_i z_j + z_i(1 - z_j) + (1 - z_i) z_j + (1 - z_i)(1 - z_j)] \sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell}$$
$$\geq \sum_{i,j} z_i z_j \cdot \min\{\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(11)}\}$$
$$+ \sum_{i,j} z_i(1 - z_j) \cdot \min\{\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(10)}\}$$

Introduce cut variables

Reminder

$x_{ik} = 1 \Leftrightarrow \forall b: \quad z_i^b = k[b], \quad k[b] =$ b-th bit of k

Insert a factor 1

$$\sum_{i,j}[z_i z_j + z_i(1-z_j) + (1-z_i)z_j + (1-z_i)(1-z_j)] \sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell}$$
$$\geq \textstyle\sum_{i,j} z_i z_j \qquad \cdot \min\{\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(11)}\}$$
$$+ \textstyle\sum_{i,j} z_i(1-z_j) \quad \cdot \min\{\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(10)}\}$$
$$+ \textstyle\sum_{i,j} (1-z_i)z_j \quad \cdot \min\{\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(01)}\}$$

Introduce cut variables

Reminder

$x_{ik} = 1 \Leftrightarrow \forall b: \quad z_i^b = k[b], \quad k[b] = $ b-th bit of k

Insert a factor 1

$$\sum_{i,j} [z_i z_j + z_i(1-z_j) + (1-z_i)z_j + (1-z_i)(1-z_j)] \sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell}$$

$$\begin{aligned}
&\geq \textstyle\sum_{i,j} z_i z_j && \cdot \min\{\textstyle\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(11)}\} \\
&+ \textstyle\sum_{i,j} z_i(1-z_j) && \cdot \min\{\textstyle\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(10)}\} \\
&+ \textstyle\sum_{i,j} (1-z_i)z_j && \cdot \min\{\textstyle\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(01)}\} \\
&+ \textstyle\sum_{i,j} (1-z_i)(1-z_j) && \cdot \min\{\textstyle\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(00)}\}
\end{aligned}$$

Eliminate assignment variables

Exemplary case: $z_i(1 - z_j)$

$$z_i(1 - z_j) \cdot \min\{\textstyle\sum_{k,\ell} c_{ijk\ell} x_{ik} x_{j\ell} : x \in \Pi_{ij}^{(10)}\}$$

- Solve the minimization problem
  $\Rightarrow$ Solution is minimum cost over the given cut
- Remaining problem contains z variables only ($n \log_2(n)$ many)
- Remark: Branching tightens the formulation further!

Self-tightening framework

Exemplary case: $z_i(1-z_j)$

$$z_i(1-z_j)\cdot\min\{\textstyle\sum_{k,\ell} c_{ijk\ell}x_{ik}x_{j\ell} : x \in \Pi_{ij}^{(10)}\}$$

- Assume $c_{ij\tilde{k}\tilde{\ell}}$ is minimizing cost term
- But: Branching does not allow to assign $i \to \tilde{k}$ or $j \to \tilde{\ell}$

$\Rightarrow$ Our estimation is too weak

- Take the minimum *feasible* cost over the given cut

Objective functions

Including a full cut pseudo base

- Maximum over all cuts
  - Pro: Maximum never less than average
  - Contra: Additional auxiliary variables seemed to harm the solver's stability
- Average over all cuts
  - Lower bound is valid for all cuts
  - $\Rightarrow$ Also valid for the average over all cuts
  - Advantage: No auxiliary variables needed
  - No numerical issues

Build the SDP

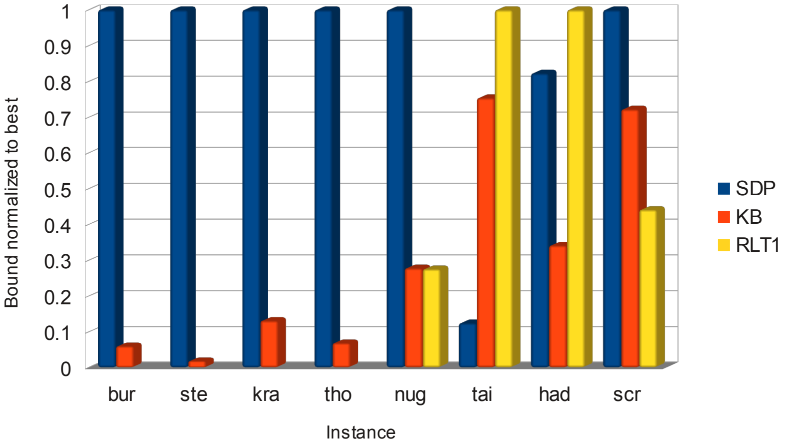
- Transform $z_i^b \in \{0,1\}$ to $y_b^i \in \{-1,1\}$
- Introduce $Y = yy^T$ and relax it to $Y \succeq 0$
- Encode QAP constraints in terms of Y
  - Cuts are balanced $\leftarrow$ Row sum of Y is 0
  - Assignment is injective $\leftarrow Y_{ij}^b$ not identical for each b
- Encode branching decisions as SDP constraints
  - Take an item i as reference item
  - Branching is done relative to i
  - $Y_{ij} = 1 \Leftrightarrow y_i \cdot y_j = 1$

    $\Rightarrow$ i and j are in the same cut

Evaluation

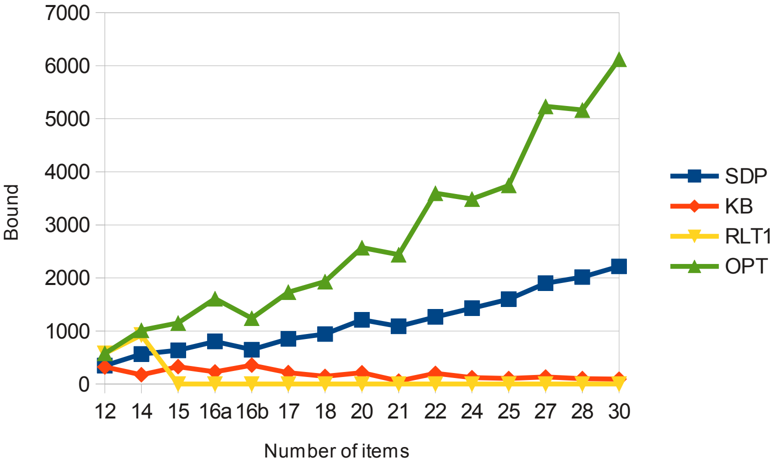
Keyboard problem instances (1h computation)

48 Intel (R) Xeon (R) E5-2680 2.50GHz cores, 258 GB RAM, single-threaded



Evaluation

Graph problems (1h computation)



Future work

1. Regarding the framework
   - Develop primal heuristics
     - Randomized rounding
     - Primal LP heuristics
     - ...
2. Regarding the applications
   - Extend the application space
     - QAP applications in image processing
     - Other quadratic programs (like quadratic set cover, ...)