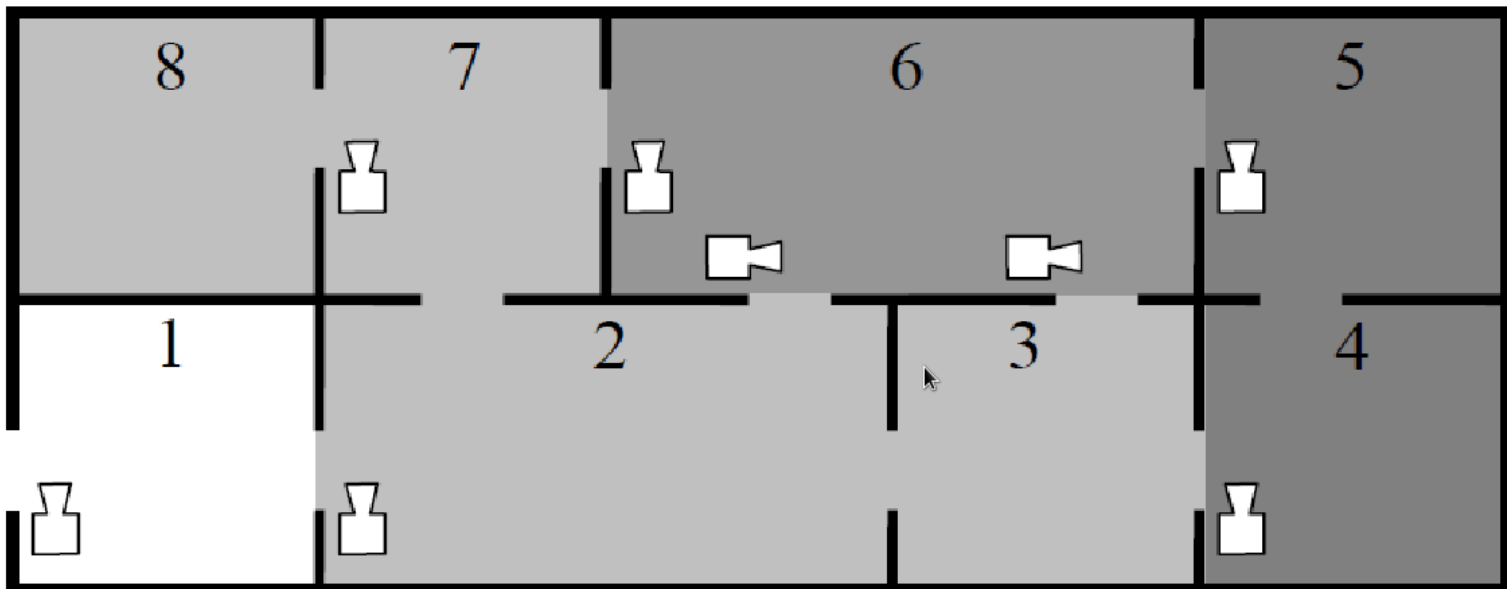
Partner Units Revisited

Klagenfurt, Oxford, Potsdam

2012 Oxford Configuration Workshop

13.01.2012

The Museum Scenario



The Partner Units Problem (PUP)

IN: Sensors on doors, rooms grouped into zones

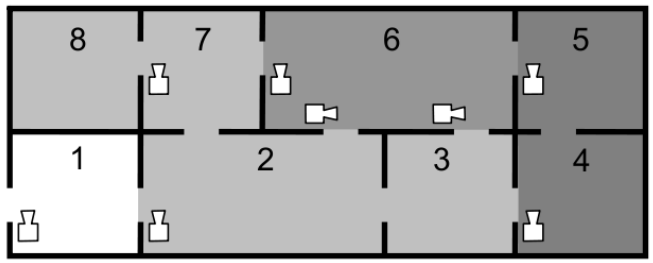
OUT: Assign sensors and zones to control units respecting:

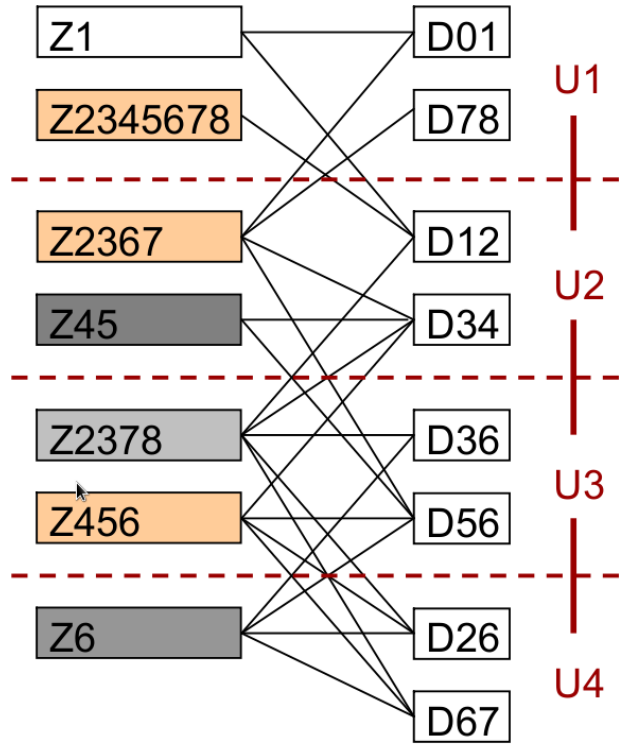
(1) Limited unit capacity
(say *UnitCap* zones and sensors each)

(2) Sensor connected to zone?
Sensor and zone on the *same or adjacent* unit
(aka partner unit)

(3) Limited connections between units
(say max *InterUnitCap* neighbours)

Museum Scenario plus Solution





The Partner Units Problem Formalized

Given a bipartite graph

Find a partitioning respecting *UnitCap* and *InterUnitCap* constraints

Reasoning

Find optimal solutions:

- Minimize the number of units
- Don’t care about the number of connections

General Problem Properties

Regular Graphs and PUP Solutions

- *k*-Regular Graph: All nodes have degree *k*
- *InterUnitCap*-regular solutions are most general solutions (if they exist)

Number of k-regular graphs (M. Meringer)

Vertices	Degree 3	Degree 4	Degree 5
10	19	59	60
11	0	265	0
12	85	1544	7848
13	0	10778	0
14	509	88168	3459383

- ☹ All can be forced
- ☺ $\mathit{InterUnitCap} = 2$? There is only one 2-regular graph

Bounds on $|\mathcal{U}|$

Lower: $|\mathcal{U}| \geq \lceil \frac{\max(|\mathcal{V}_s|, |\mathcal{V}_z|)}{\mathit{UnitCap}} \rceil$

Upper: $|\mathcal{V}_s| + |\mathcal{V}_z| \geq |\mathcal{U}|$

Upper, where $\mathit{InterUnitCap} = 2, \mathit{UnitCap} > 1$: $\max(|\mathcal{V}_s|, |\mathcal{V}_z|) \geq |\mathcal{U}|$

All instances “in the wild” solvable at lower bound

Complexity

Mostly unknown, but:

- PUP is BINPACKING if $\mathit{InterUnitCap} \in \{0, 1\}$
- PUP is polynomial if $\mathit{InterUnitCap} = 2$ and *UnitCap* is fixed

Encodings and Algorithms

Old Methods (CPAIOR 2011)

- $|\mathcal{U}| \times |\mathcal{U}|$ matrix for inter-unit-connections
- Fixed cyclic layout if *InterUnitCap* $= 2$
- Counting constraints up to *InterUnitCap* and *UnitCap*
- As answer set-, integer- and constraint program and in SAT
- Implementation of polynomial DECPUP algorithm

Empirically good:

- SAT, ASP (i.e. CDCL) for *InterUnitCap* > 2
- CSP for *InterUnitCap* $= 2$

Search Strategies

(1) Iterative deepening on the number of units (lb to ub)
(2) Search minimal solution using ub units
(3) Try (1) with lb, then (2) with ub units

(1) allows to assume a fixed cyclic solution layout for $\mathit{InterUnitCap} = 2$

Maximum Joint Vertex Degree

- No vertex can have more than $(\mathit{InterUnitCap} + 1) * \mathit{UnitCap}$ neighbours
- Throughout search the sensors (or zones) on a k-clique of units cannot have more than $[k * ((\mathit{InterUnitCap} + 2) - k)] * \mathit{UnitCap}$ neighbours

Some Ideas of Symmetry Breaking

- Vertex v_1 goes on unit U_1,
  Vertex v_2 goes on unit U_1 or U_2,
  Vertex v_3 goes on unit U_1 or U_2 or U_3, etc.
- If v_i is on U_j, $v_k, k < i$ have to be on $U_l, l \leq j$
- Lexicographically order inter-unit connection matrix

Variable Orderings

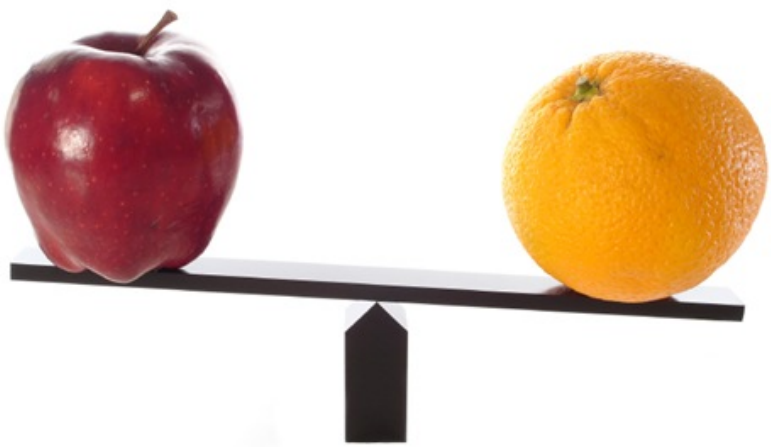
For search in CSP and symmetry breaking

Families:

- `adj`: Each variable has an assigned predecessor
- `deg`: Mininum or maximum degree

plus `random` and cross breeds

Experimental Results



Conflict Driven Clause Learning Solvers

- Work on ground Boolean problem representation
- Typically only one rule of inference (unit clause)
- Add conflict clauses during search
- Variables are selected heuristically

We use POTASSCO, LINGELING, MINISAT

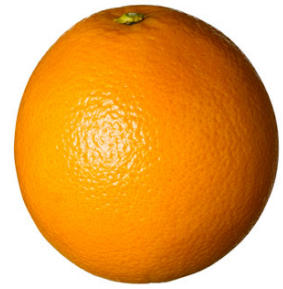
Key CDCL Findings

- Built-in support for “atmost” constraints beats counting encodings
- Symmetry breaking via `maxdegadj` and `maxadjdeg` really good
- Folding the matrix helps ASP and hurts SAT

CLASP from POTASSCO beats MINISAT and LINGELING

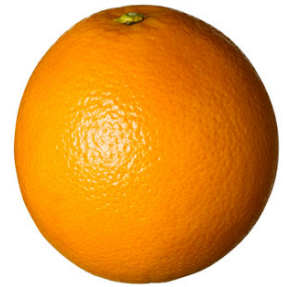
Constraint Solvers

- Work on non-ground problem representation
- Different propagators for different constraints
- Variable and value selection is programmed by hand



We use ECL^{i}PS^e and MINION

Key Findings for Constraint Solvers



- Non-decomposing `adj` variable orderings rule for *InterUnitCap* $= 2$
- On the same model MINION is much faster than ECL^{i}PS^e
- Need custom propagator for inter-unit matrix for *InterUnitCap* > 2
- `maxadjdeg` and `maxdegadj` again did best
- Don't assign inter-unit connections early

Lazy Clause Generation

- Works on CSP with propagators and programmed search
- Incrementally translates propagations to SAT clauses
- SAT solver inside does the clause learning
- CSP solver can do backjumping
- No custom propagators yet

We use FDX from MINIZINC

General Findings

- Constraint solvers rule for $\mathit{InterUnitCap} = 2$
- CDCL solvers rule for $\mathit{InterUnitCap} > 2$
- Big speedups, still can't tackle all industrial instances

Thank you!