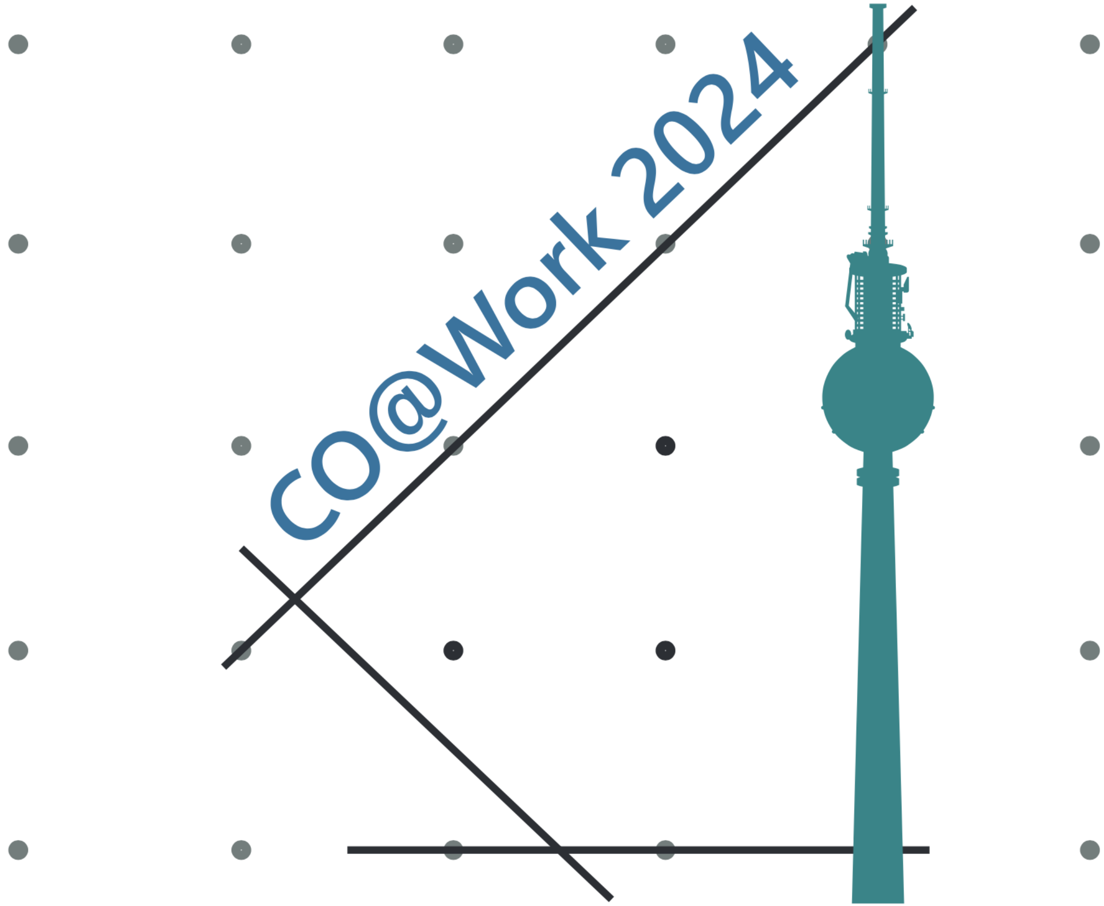
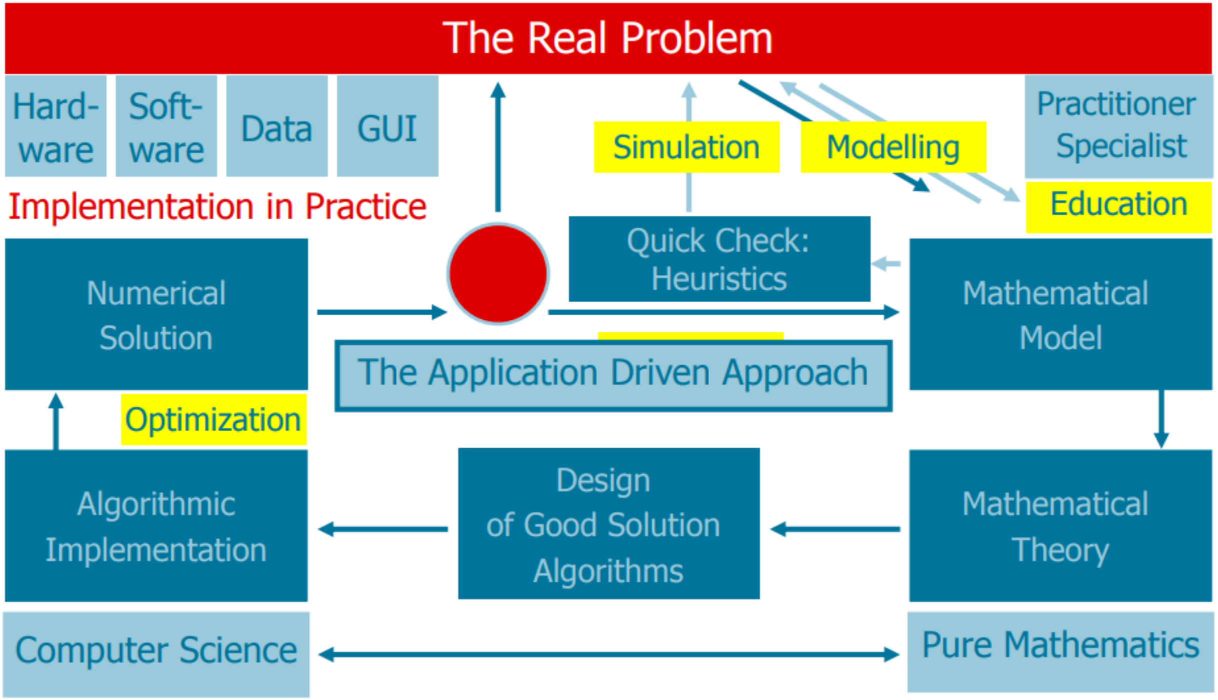
Aspects of MIP Modelling

Ambros Gleixner
HTW Berlin & ZIB



Grötschel's Problem Solving Cycle in Modern Applied Mathematics



- Modelling is the first and recurring step in solving a real-world problem.

Modelling as an art?

- “Art is a lie that makes us realize truth.”
  (Picasso, Cocteau, Camus, …)

- A mathematical model is also a lie that helps us to see particular aspects of a problem:
  - by selecting, ignoring, simplifying, emphasizing, weighting, …
- This is also a creative process.
- Like art, mathematical modelling is also a craft:
  - Can learn tools and skills: some of them today!

Agenda

- Building a model
- 0/1 vs general integer, assignment formulations
- TSP example
- Combinatorial Constraints
- Indicator Constraints
- How not to do it
- Concluding remarks

Building a model

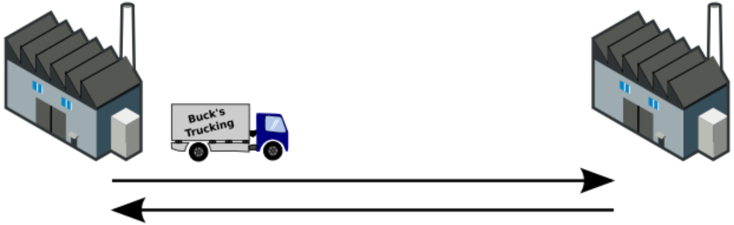
Ambros Gleixner
HTW Berlin & ZIB

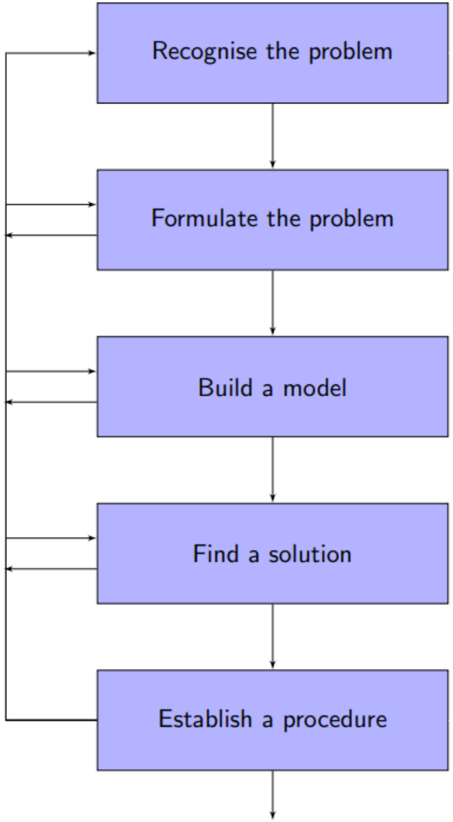
What is modelling?

- Describing a particular situation using a collection of logical and mathematical relationships.
- In Optimization:
  - An objective function is used to evaluate alternative solutions.
  - Constraints define the alternative solutions that are feasible for the situation under consideration.
- Why do we build models?
  - To capture essential aspects
  - Reality is too detail: simplify
  - To evaluate what-if scenarios
  - Experimentation or simulation might not be possible
  - …

The model building process

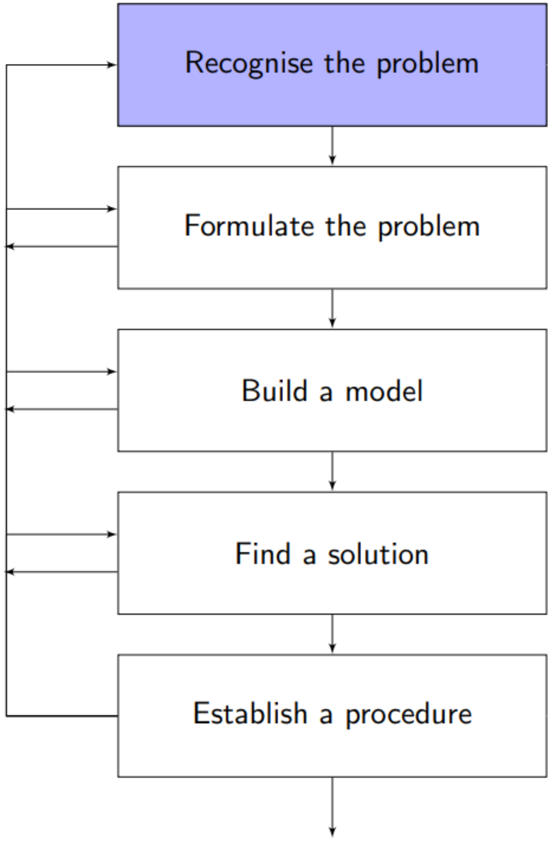
- Think about what is important to the situation and the problem considered
- An abstraction of the complete problem
- Simplification of the problem can yield tractable problems and interpretable solutions





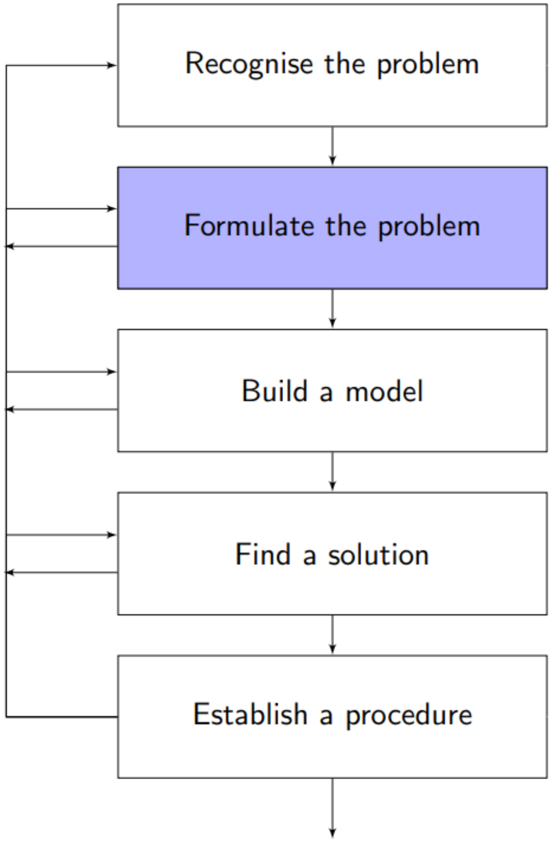
The model building process

- Identify the problem
  - Can be abstract or real world
  - Single out a concrete question



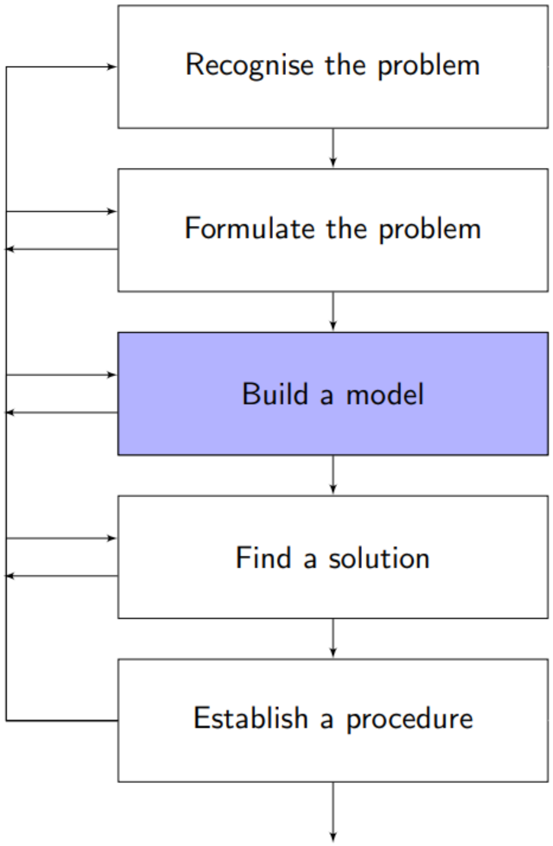
The model building process

- Identify the important features
  - Define this problem in mathematical and logical notation
  - Never forget that your model is an abstraction of reality

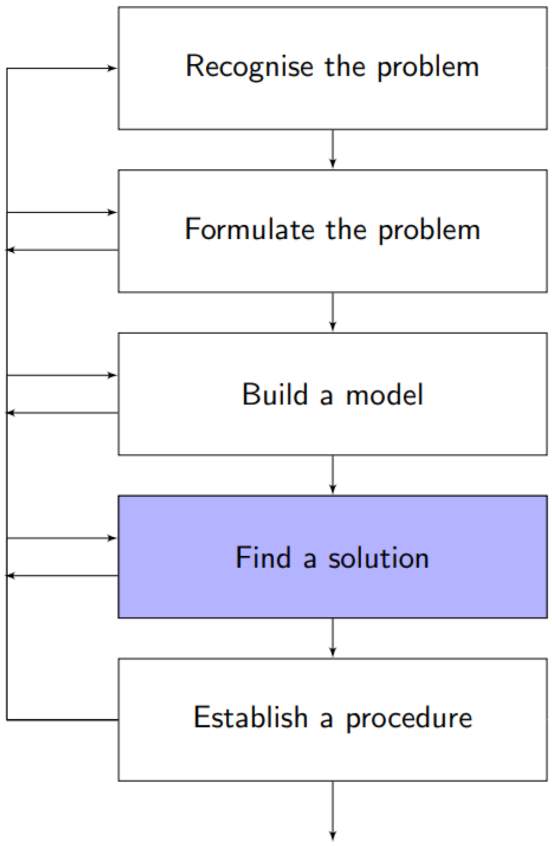


The model building process

- Transfer the mathematical problem formulation to a model
  - Make use of available modelling tools
    - direct coding via API or modeling language
  - Think about alternative formulations

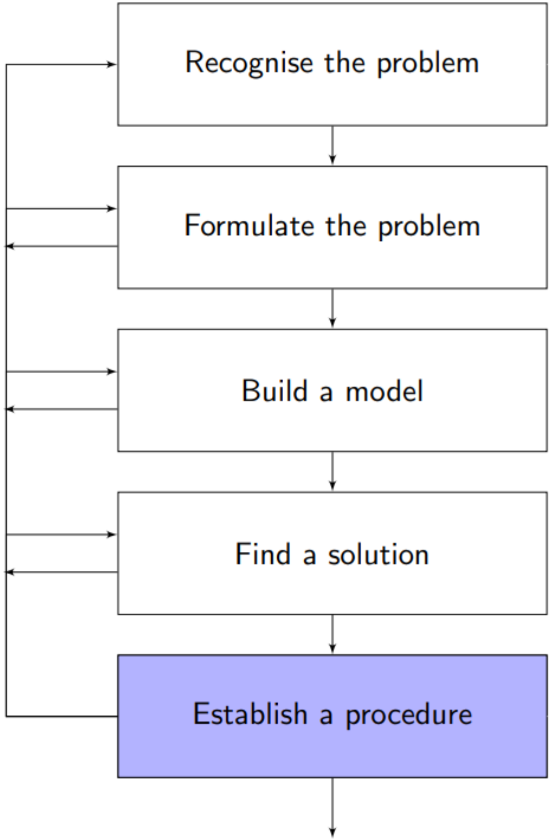


The model building process



- Employ tools to solve the mathematical model
  - In our case, typically a MIP solver
  - Check validity of solution in practice

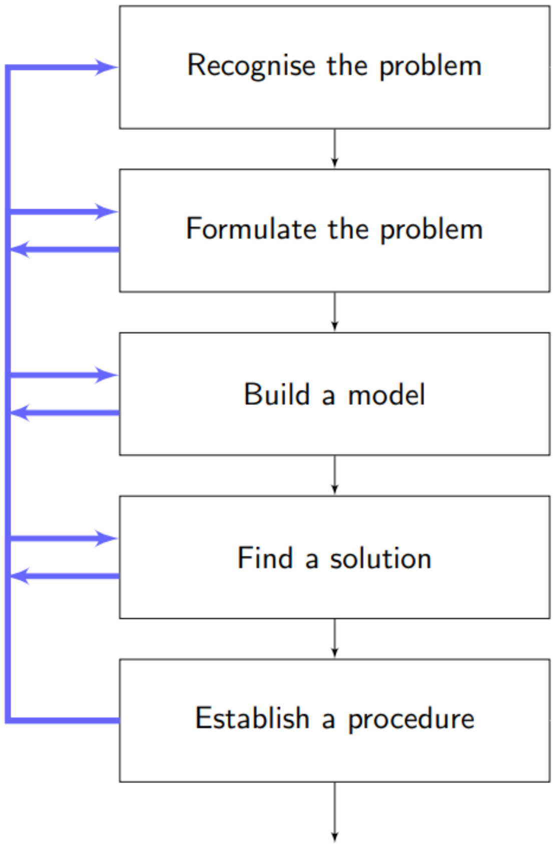
The model building process



- Deployment:
  - Design a procedure to implement the solution
  - This is where a lot of OR projects fail!

The model building CYCLE

- FEEDBACK is crucial
  - Each stage helps refine the previous stages
  - The modelling process aids the understanding of the problem.
  - The problem understanding develops and the solution approach becomes clearer.



Example: Knapsack

A burglar has a knapsack with 15kg capacity and breaks into a house with the following items: 1kg worth 2000€, 1kg worth 1000€, 2kg worth 2000 €, 4kg worth 10000€, 12kg worth 4000 €

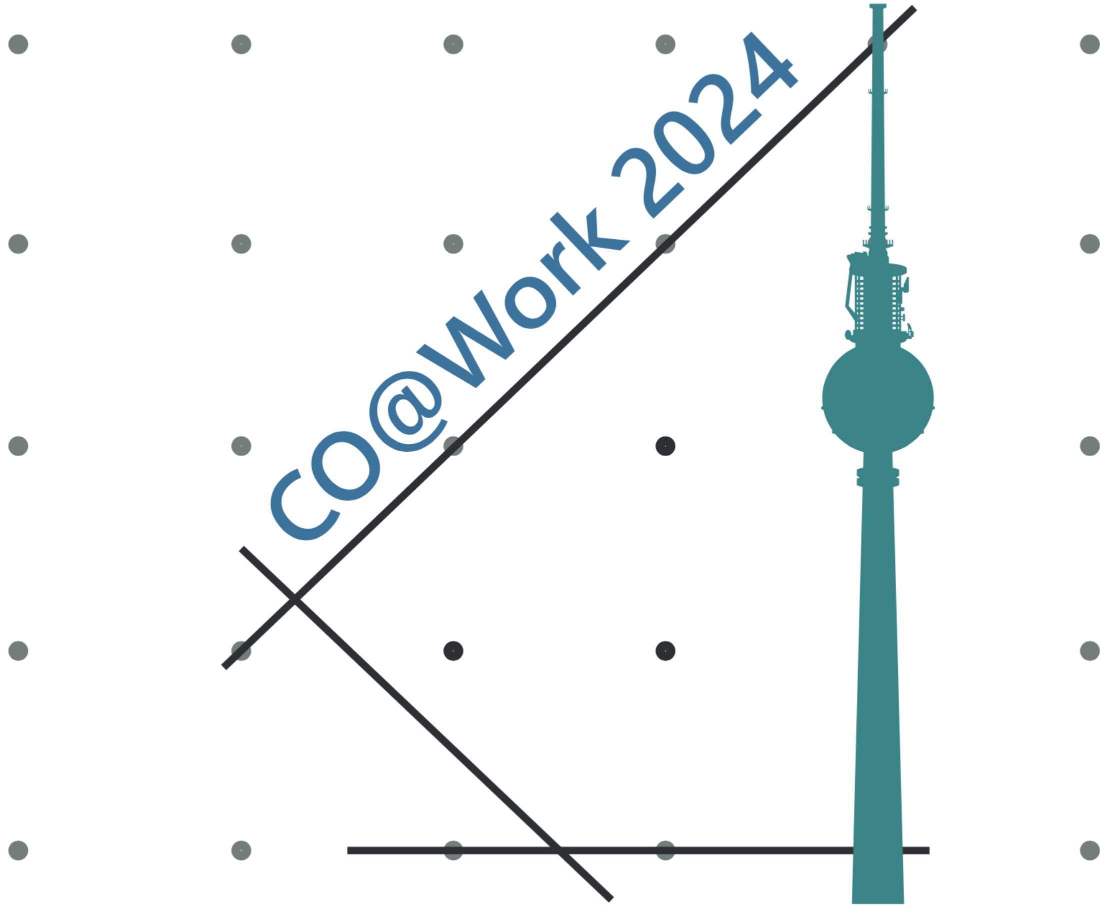
- What are the variables? What are the constraints? What is the objective?
  - Variables: $x_i \in \{0,1\}$: Do I take item i?
  - Constraint: Must not exceed capacity: $x_1 + x_2 + 2x_3 + 4x_4 + 12x_5 \leq 15$
  - Objective: Maximize revenue: $\max 2x_1 + x_2 + 2x_3 + 10x_4 + 4x_5$
- Final model:
  - $\max 2x_1 + x_2 + 2x_3 + 10x_4 + 4x_5$
    s.t. $x_1 + x_2 + 2x_3 + 4x_4 + 12x_5 \leq 15$
    $x_i \in \{0,1\}$

General integers and other variable types

Ambros Gleixner
HTW Berlin & ZIB

Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

ZIB
ZUSE INSTITUTE BERLIN



Binaries or integer variables?

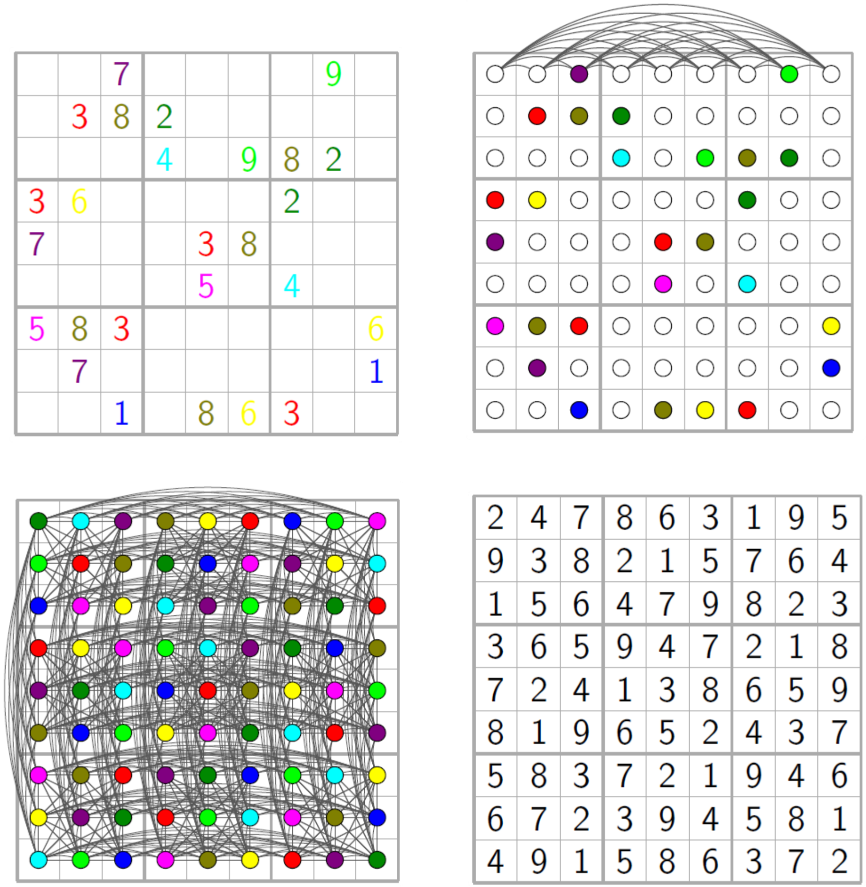
- Rule of thumb
  - Use general integers whenever they represent actual quantities and ordering is important
  - Whenever integers represent just „some different values“, use binaries instead
- Example: Sudoku

8			6			9		5
				2		3	1	
		7	3	1	8		6	
2	4						7	3
		2	7	9		1		
5				8			3	6
		3						

Naive approach:
Use 81 integer variables $1 \leq y_i \leq 9$
And then...?

Sudoku as graph coloring

- Each number corresponds to a color, each cell to a vertex.
  - 9 binaries per cell: x_{ijk}
  - Exactly one color: $\sum_k x_{ijk} = 1 \ \forall i,j$
- Edges when two cells must not have the same color/number, e.g., $x_{1,1,1} + x_{1,2,1} \leq 1$
  - Can do better and add clique equations: $\sum_j x_{ijk} = 1 \ \forall i,k$
- Sudoku corresponds to the question: Is there a feasible 9-coloring of a partially colored graph with 27 9-cliques?



Assignment structure

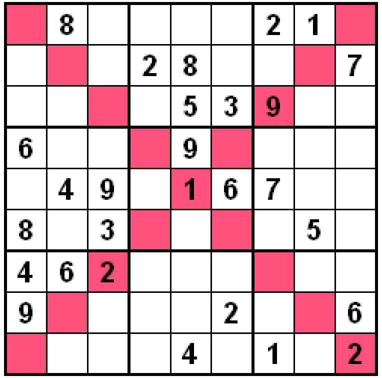
$$
\begin{aligned}
\text{maximize} \quad & 0 \\
\text{subject to} \quad & \sum_{v=1}^{9} x_{vrc} = 1 \text{ for } r, c \in [1, 9] \\
& \sum_{r=1}^{9} x_{vrc} = 1 \text{ for } v, c \in [1, 9] \\
& \sum_{c=1}^{9} x_{vrc} = 1 \text{ for } v, r \in [1, 9] \\
& \sum_{r=3p-2}^{3p} \sum_{c=3q-2}^{3q} x_{vrc} = 1 \text{ for } v \in [1, 9] \text{ and } p, q \in [1, 3]
\end{aligned}
$$

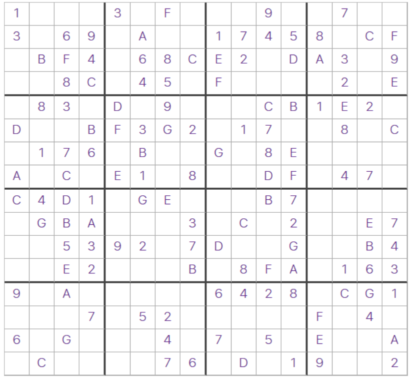
8			6			9		5
				2		3	1	
		7	3	1	8		6	
2	4						7	3
		2	7	9		1		
5				8			3	6
		3						

- Important concept: Assignment structure
  - Assignment problem: Given costs c_{ij} for assigning object i to person j
  - $\min \sum_{i,j} c_{ij} x_{ij}$
    s.t. $\sum_i x_{ij} = 1$
    $\sum_j x_{ij} = 1$
  - Easy, but a common substructure in other problems

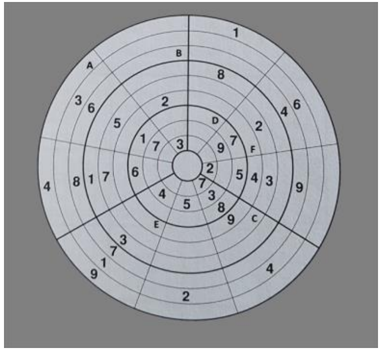
Many variants, similar models

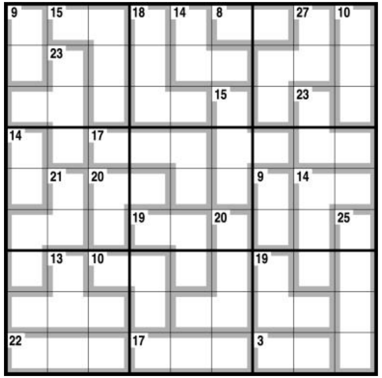
- X-Sudoku
- 16x16-Sudoku
- 3D-Sudoku

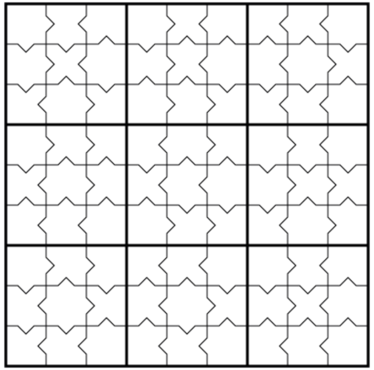




- Ensaimada
- Killer-Sudoku
- Comparison-Sudoku







CO@Work 2024

Many choices in modelling

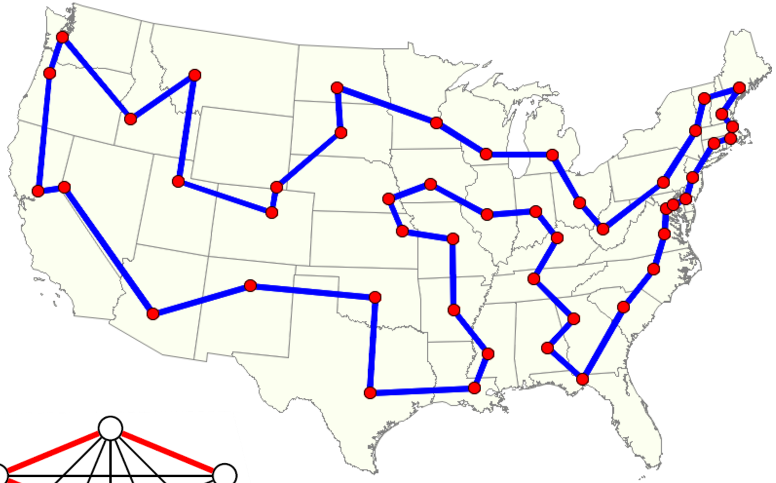
Ambros Gleixner
HTW Berlin & ZIB

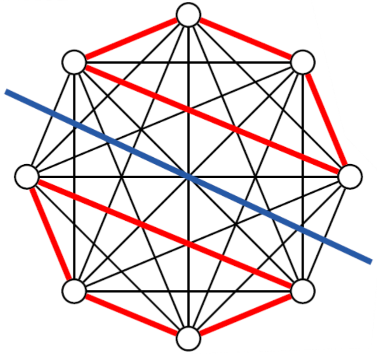
Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

ZIB
ZUSE INSTITUTE BERLIN

TSP – the most famous optimization problem?

- TSP: Given a complete graph $G = (V, E)$ and distances c_{ij} for all $(i, j) \in E$:
  - Find a Hamiltonian cycle (tour) of minimum length.
- Classical MIP formulation:
  - $\min \sum c_e y_e$
    s.t. $\sum_{e \in \delta(v)} y_e = 2$ for all $v \in V$
    $\sum_{e \in \delta(S)} y_e \geq 2$ for all $S \subseteq V, S \neq \emptyset$
    $y_e \in \{0,1\}$
- Highly efficient special pupose codes
  - Concorde





TSP – Miller Tucker Zemlin formulation (1960)

- Consider G as directed graph with arcs (ij) and (ji) between all $i, j \in V$. Use variables
  - y_{ij} whether (ij) is part of the tour
  - u_i for the number of nodes visited before i
- Model:
  - $$\begin{aligned} &\min \textstyle\sum c_{ij} y_{ij} \\ \text{s.t. } &\textstyle\sum_{(i,j)\in\delta^-(j)} y_{ij} = 1 \quad \text{for all } j \in V \\ &\textstyle\sum_{(i,j)\in\delta^+(i)} y_{ij} = 1 \quad \text{for all i} \in V \\ &u_1 = 0 \\ &u_i - u_j + (n-1)y_{ij} \le n-2 \text{ for all } (i,j) \in A, j \neq 1 \\ &1 \le u_i \le n-1 \text{ for all } i \in \tilde{V} \coloneqq V \setminus \{1\} \\ &y_{ij} \in \{0,1\} \\ &u_i \in \mathbb{Z}_{\ge 0} \end{aligned}$$

TSP – Vyve Wolsey formulation (2006)

- Now, interpret TSP as fixed charge network design problem
  - delivering one unit of flow from source node to each other node
  - For each city $l \in V$, define neighborhood $l \in V_l \subseteq V$ (typically k nearest nodes)
  - Introduce variables w_{ij}^l for flow to city l on arc (i,j)

$$\min \sum c_{ij} y_{ij}$$

s.t. $\sum_{(i,j)\in\delta^-(j)} y_{ij} = 1$ for all $j \in V$

$\sum_{(i,j)\in\delta^+(i)} y_{ij} = 1$ for all $i \in V$

$u_1 = 0$

$u_i - u_j + (n-1)y_{ij} \leq n-2$ for all $(i,j) \in A, j \neq 1$

$1 \leq u_i \leq n-1$ for all $i \in \tilde{V}$

$y_{ij} \in \{0,1\}$, $u_i \in \mathbb{Z}_{\geq 0}$

$\sum_{(i,l)\in\delta^-(l)} w_{il}^l - \sum_{(l,i)\in\delta^+(l)} w_{li}^l = 1$ for all $l \in \tilde{V}$

$\sum_{(i,j)\in\delta^-(j)} w_{ij}^l - \sum_{(i,j)\in\delta^+(i)} w_{ji}^l = 0$ for all $l \in \tilde{V}, j \in V_l, j \neq l$

$0 \leq w_{ij}^l \leq y_{ij}$ for all $l \in \tilde{V}, (i,j) \in A(V_l) \cup \delta^-(V_l)$

Many different variations (Achterberg et al 2008)

- Vyve-Wolsey strengthens Miller-Tucker-Zemlin: It gives a better LP bound
- In the following, we consider redundant changes, that do not change the LP bound (or the set of integer optimal solutions)
- Yet, they can have a huge effect on solver performance

- Relaxations (that remove redundant constraints):
  - Remove upper bounds on u_i
  - Equality in the flow conservation constraints can be ommited:

$$\sum_{(i,l)\in\delta^-(l)} w_{il}^l - \sum_{(i,l)\in\delta^+(l)} w_{li}^l \geq 1$$

$$\sum_{(i,j)\in\delta^-(j)} w_{ij}^l - \sum_{(i,j)\in\delta^+(j)} w_{ji}^l \geq 0$$

Many different variations II

- Additional restrictions, some of which, the solver can figure out itself
  - E.g., fixing w_{ij}^{l} with $(i,j) \in \delta^{+}(V_l)$ to zero
- Others lead to huge reductions (e.g., changing flow constraints to inequality)
- 64 cases, which fall into three clusters w.r.t. problem size (minor variations still occur)
  - Some solve in seconds, others not in a day

Conclusions from (Achterberg et al. 2008)

- “very important to precisely state the model formulation when reporting computational results. Otherwise reproducibility will be difficult to achieve.”
- “the intractability of a specific formulation of a model using a specific solver does not necessary imply that the model in general is intractable.”
- “might be useful to identify solving strategies that are likely to be independent of the specific formulation.”
  - “One candidate in this regard clearly is the branching strategy. The strategy used in SCIP, while considerably slower in the best case than CPLEX, has a much smaller dependency on the formulation.”
  - “As a first step, we have proposed the notion of implicit integer variables”

Logical constraints

Ambros Gleixner
HTW Berlin & ZIB

Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences
ZIB ZUSE INSTITUTE BERLIN

CO@Work 2024

Logical Constraints

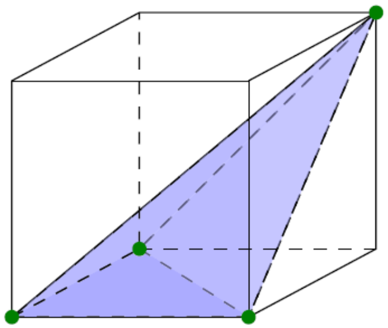
- For binary resultant r and operators x_i
  - $r = \text{AND}(x_1, \ldots, x_n)$: $r = 1 \Leftrightarrow x_1 = \cdots = x_n = 1$
  - $r = \text{OR}(x_1, \ldots, x_n)$: $r = 1 \Leftrightarrow \exists i : x_i = 1$
  - $r = \text{XOR}(x_1, \ldots, x_n)$: $r = 1 \Leftrightarrow |i : x_i = 1|$ is odd

- Relatively common constructs in modelling
- Easy to linearize
  - Or are they?

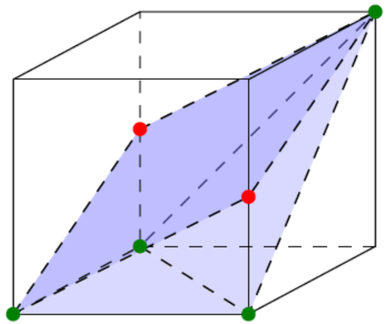
Advantages

- Convenient for modelling
  - Supported by many languages and solver interfaces
  - Together with MIN, MAX, ABS constraints
- Solver might decide dynamically how to linearize them
- Solver might use constraint specific presolving techniques
  - $r = \mathrm{AND}(x, y, z),\ z = \mathrm{AND}(a, b) \Rightarrow r = \mathrm{AND}(x, y, a, b)$
- Higher-level formulation might give additional structural insights that can be exploited

Linearizing AND constraints



- $\sum_{i=1}^{n} x_i - r \leq n - 1$
- $x_i - r \geq 0$ for all $i \in \{1, \dots, n\}$
- Strong relaxation
  - n+1 linear constraints
  - Only integral vertices (green)

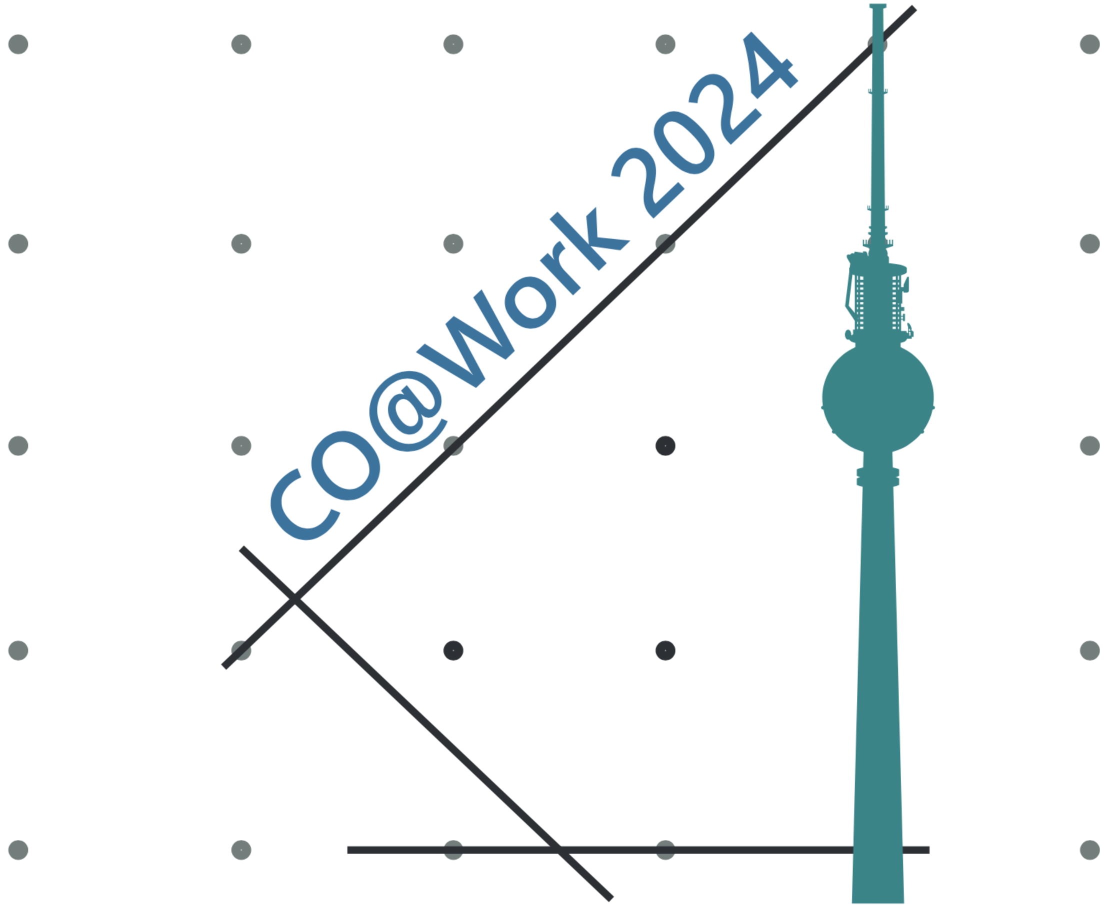


- $\sum_{i=1}^{n} x_i - r \leq n - 1$
- $\sum_{i=1}^{n} x_i - nr \geq 0$
- Weak relaxation
  - 2 linear constraints
  - Contains fractional vertices (red)

Domain propagation

- There are four propagation rules for AND constraints
  1. $r = 1 \rightarrow x_i = 1$ for all i
  2. $x_i = 1$ for all $i \rightarrow r = 1$
  3. $\exists i\colon x_i = 0 \rightarrow r = 0$
  4. $r = 0 \text{ and } x_i = 1 \forall i \in \{1, \dots, n\} \backslash \{j\} \rightarrow x_j = 0$
- Do we lose information by using a linearization?
  - Rule 1 is enforced by $\sum_{i=1}^{n} x_i - nr \geq 0$ or by all $x_i - r \geq 0$
  - Rule 2 is enforced by $\sum_{i=1}^{n} x_i - r \leq n - 1$
  - Rule 3 is enforced by $\sum_{i=1}^{n} x_i - nr \geq 0$ or by $x_i - r \geq 0$
  - Rule 4 is enforced by $\sum_{i=1}^{n} x_i - r \leq n - 1$
- Not necessarily, but can slow down the LP.

Indicator constraints



Ambros Gleixner
HTW Berlin & ZIB

Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

ZUSE INSTITUTE BERLIN

Indicator Constraints

- Model If-then relations
- Most simple form: If (x==1) then y==0 with x binary, y continuous
  - Often written as $x \rightarrow y = 0$
  - x is called the indicator variable
- General form: Indicator for constraints $x_0 \rightarrow a^T x \leq b$
  - Constraint is enforced when $x_0 = 1$, relaxed otherwise
- Used to model that subsets of constraints have to hold
- Or for adding penalty terms when certain constraints do not hold
- The same indicator variable can be used in different indicator constraints to model different scenarios

Indicator Constraints: big-M formulation

- Take indicator constraints $y \rightarrow a^T x \leq b$
- Linearize as $a^T x \leq b + (1 - y)\mathrm{M}$
  - Requires careful choice of M
  - E.g. $\mathrm{M} = \max(a^T x - b)$, but with user knowledge, much smaller values might be feasible
  - Too small M might lead to solutions being cut off
- Propagation, theoretically the same, but numerically it might be different...
  - Big-M formulations are known to be numerically cumbersome
  - For $y \leq 1000000x,\ x \in \{0,1\}$, the solution $x = 0.0000001, y = 1$ is feasible (and „integer“)
- Indicator formulation often solve slower because information is not present in the LP relaxation
- Choose your big-Ms wisely!!! Try both variants, indicators and big-Ms

Some caveats

Ambros Gleixner
HTW Berlin & ZIB

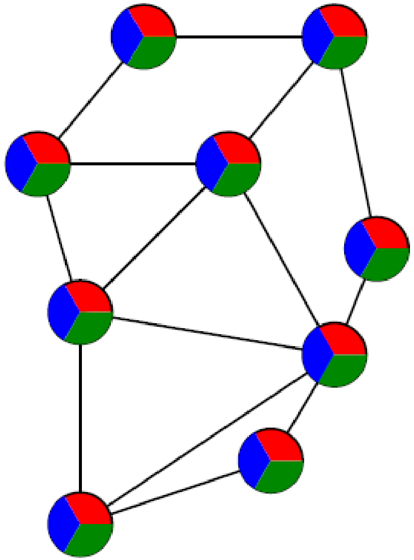
CO@Work 2024

Making assumptions what structure a MIP solver can recognize

- MIP solvers are great at making deductions from single constraints or pairs of constraints
  - Less so from a specific combinatorial structure that is implictly captured in hundreds of constraints
  - The solver will, e.g., detect a single-layer network flow structure, but not necessarily further layers or dependencies.
- You as a modeler are the „structure“ expert, try passing information to the solver

Destroying structure that is there

- Often, hard to prove optimality for symmetric models
  - If possible, choose a non-symmetric formulation
- MIP solvers employ sophisticated methods to handle symmetries
- Breaking them by hand („I fixed a few of the decisions“) might do more harm than good
  - Also, it increases the risk of making a non-obvious error

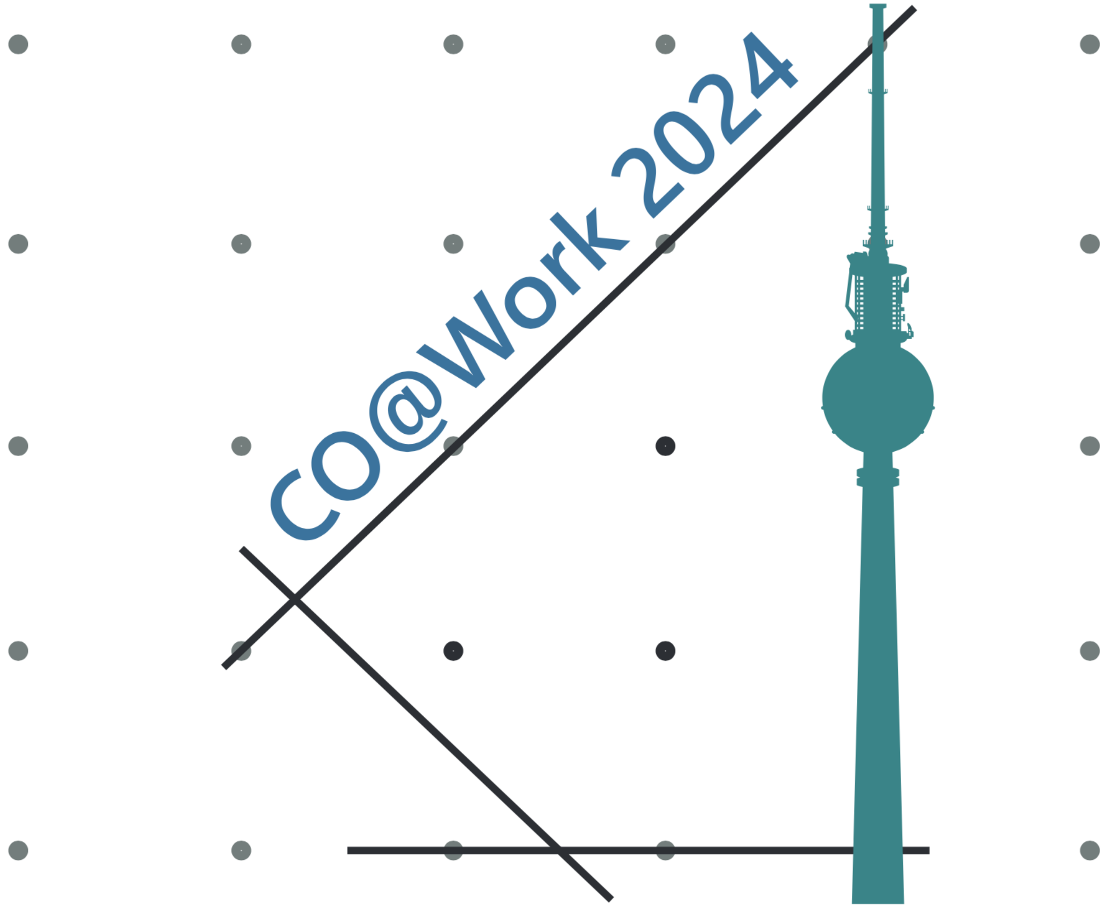


Some conclusions

Ambros Gleixner
HTW Berlin & ZIB

Hochschule für Technik
und Wirtschaft Berlin
University of Applied Sciences

ZIB
ZUSE INSTITUTE BERLIN



What does a good model look like?

- Compact is not always better
  - There are huge models (>1M variables) that solve in seconds and small (<50 variables) that do not solve in days
- Ideally, the LP optimum should be close to the integer optimum (tight formulation)
- Small number of fractionals in the LP solution is a plus
- Fixing variables should have an impact on other variables (not too many degrees of freedom)
- Keep numerics under control: not too large span of coefficients
  - Not more than six orders of magnitudes for single row/column
  - Not more than nine over the whole model
- Try to avoid big-M formulations

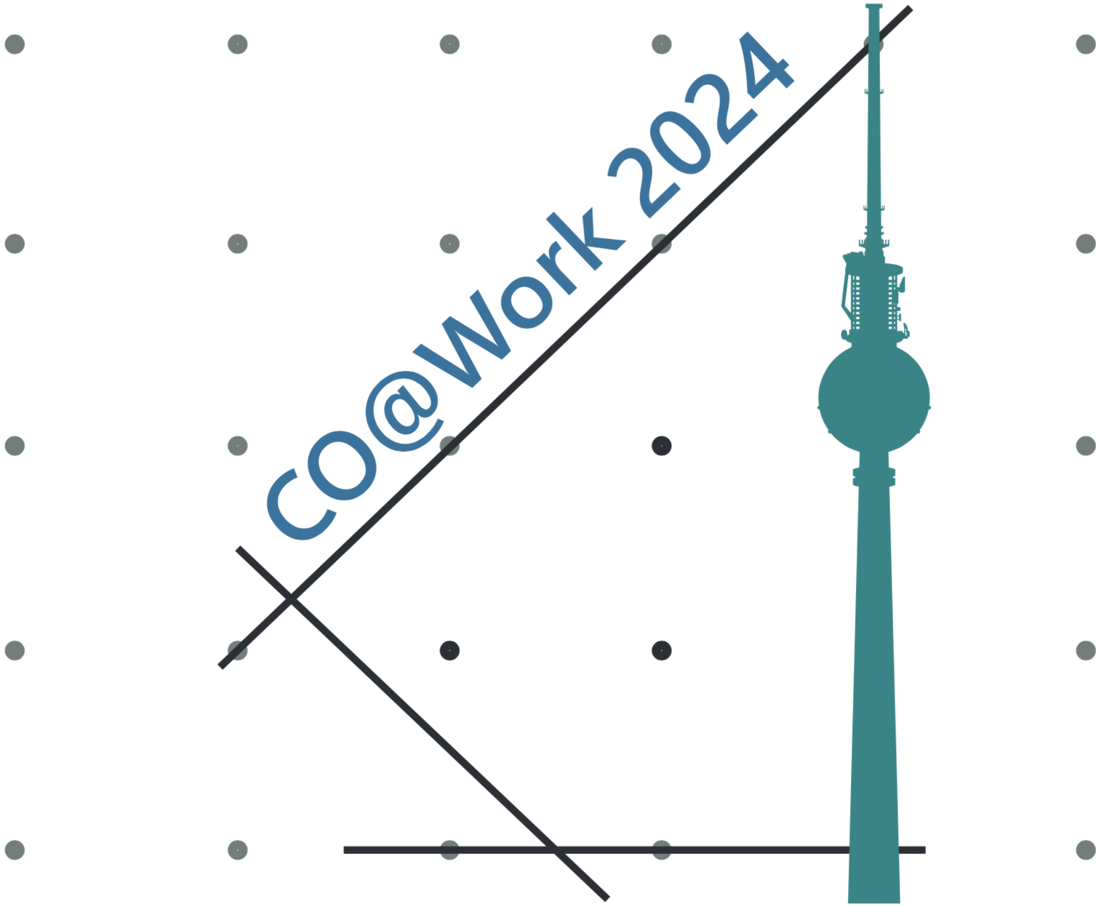
Things to keep in mind

- The first modelling attempt often is infeasible or unbounded
  - MIP solvers are typically super fast in detecting those „trivial“ errors
- The „second“ attempt often produces unsatisfying solutions
  - Might violate some implicit constraints that were forgotten in the model
- MIP solvers prefer extremal solutions
  - Customers often do not
  - Most often, there are alternative optima
  - Or solutions almost as good that might fulfill some robustness considerations

Things to keep in mind

- Try to stress-test your model
  - Do the solutions for corner cases make sense?
  - Always ensure your solutions are at least two-opt.
- A solution is only always optimal (or [in]feasible) w.r.t. your model
  - ...and the data that was fed into your model
  - Slight violations might still be tolerable in practice
  - Often enough solving to exact optimality is not required (e.g., due to inaccurate data)
- Be prepared for a pushback

Thank you! Questions?



Ambros Gleixner
HTW Berlin & ZIB

Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

ZIB
ZUSE INSTITUTE BERLIN