Solving Practical Configuration Problems Using UML

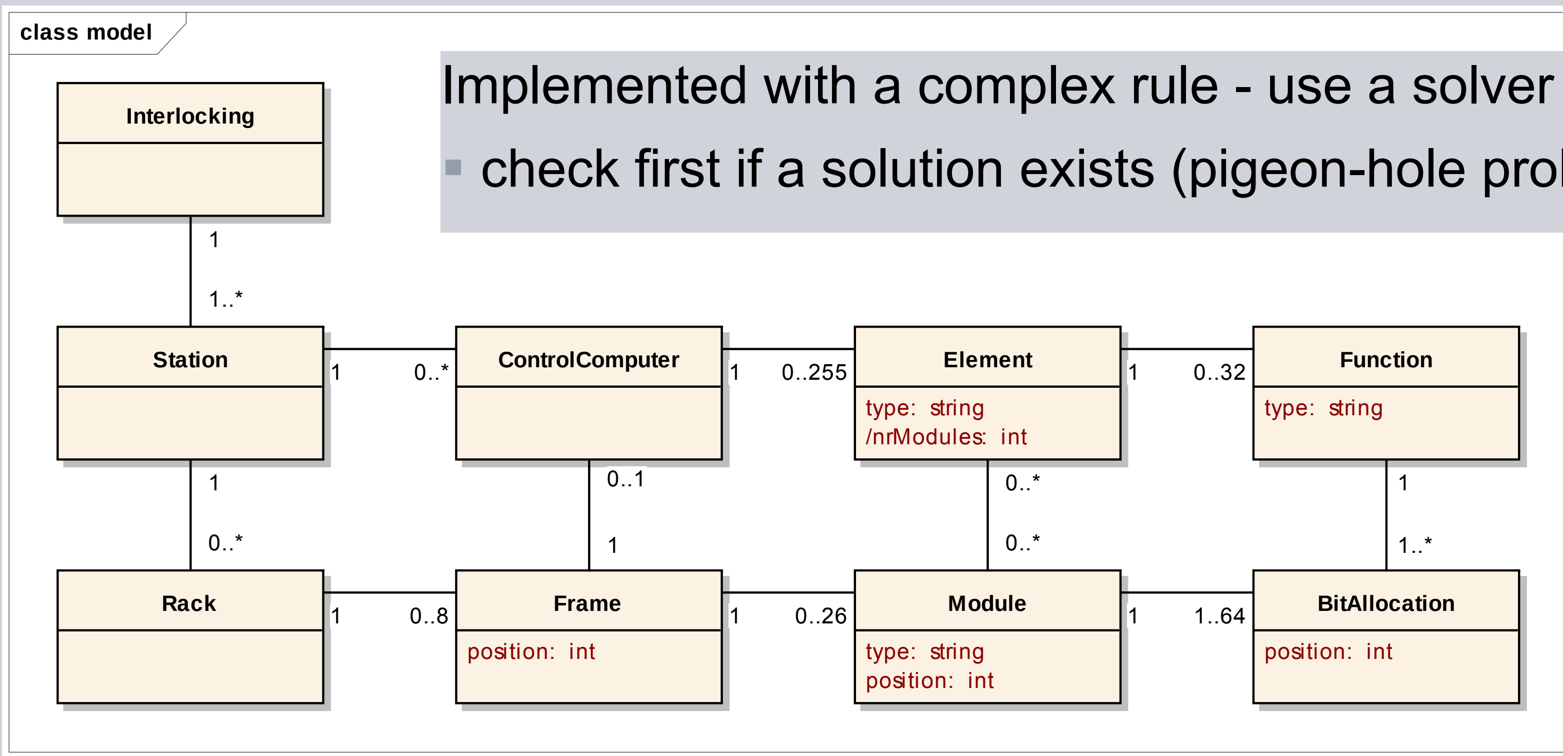
Andreas Falkner, Gottfried Schenner (Siemens AG Austria)
Gernot Salzer, Ingo Feinerer (University of Technology, Vienna)

ECAI 2008 Configuration Workshop

The initial problem

Hardware configuration for railway interlocking system

- given the elements and their functions, how much hardware is needed?
- actually create and configure hardware later (sizing problem)



Implemented with a complex rule - use a solver instead?

- check first if a solution exists (pigeon-hole problem)

  Siemens IT Solutions and Services

Academic approach

Subset of the problem:

- UML classes and minimal number of instances
- associations with lower/upper bounds for cardinalities (multiplicities)

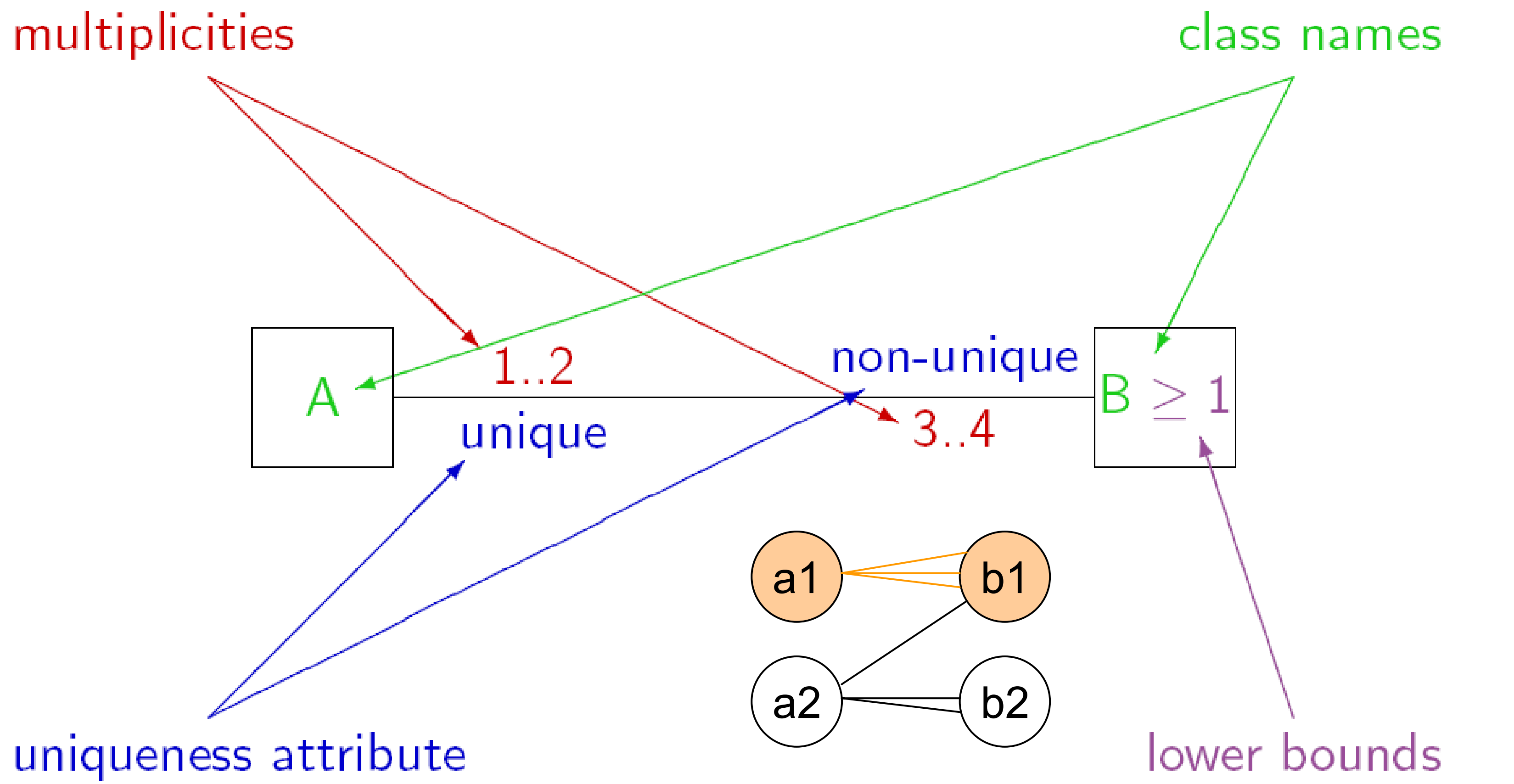
Salzer/Feinerer: paper at TASE 2007

- transform problem into linear Diophantine inequalities
- only 2 variables in inequalities: still NP-complete
- no upper bound of sum: polynomial
- algorithm with weighted directed graphs: O (inequalities * variables^2)
- existence of a solution, minimal solution: correspond to diagram

Dissertation of Feinerer:

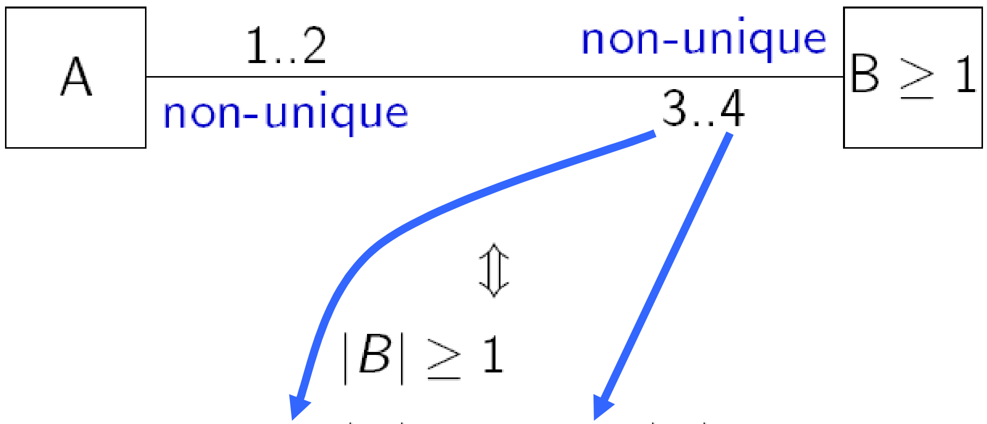
- won Austrian INiTS Award 2007 for innovative business ideas

Specification with UML



unique = "count objects" (default) non-unique = "count links"

Transformation to linear inequalities



$$\Updownarrow$$

$$|B| \geq 1$$

$$3 \cdot |A| \leq n \leq 4 \cdot |A|$$

$$1 \cdot |B| \leq n \leq 2 \cdot |B|$$

$$(n \ \dots \ \text{number of links})$$

$$\Updownarrow$$

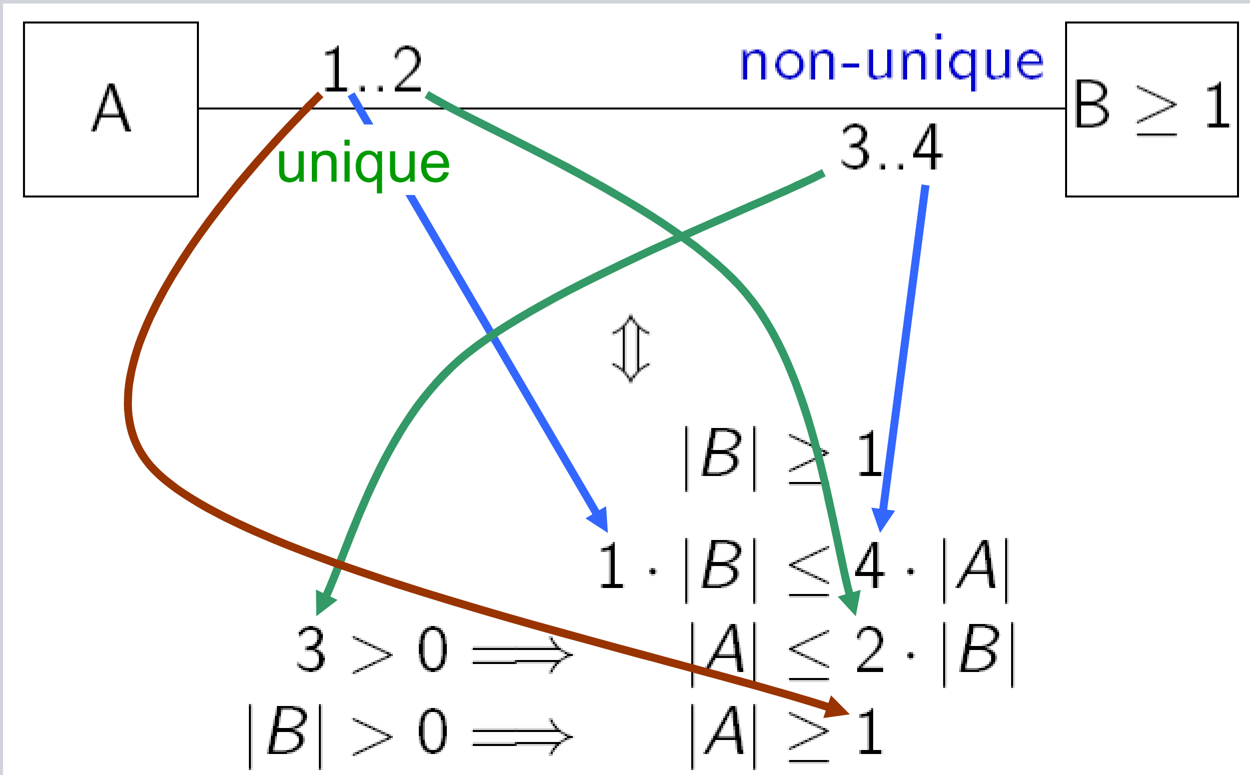
$$|B| \geq 1$$

$$1 \cdot |B| \leq 4 \cdot |A|$$

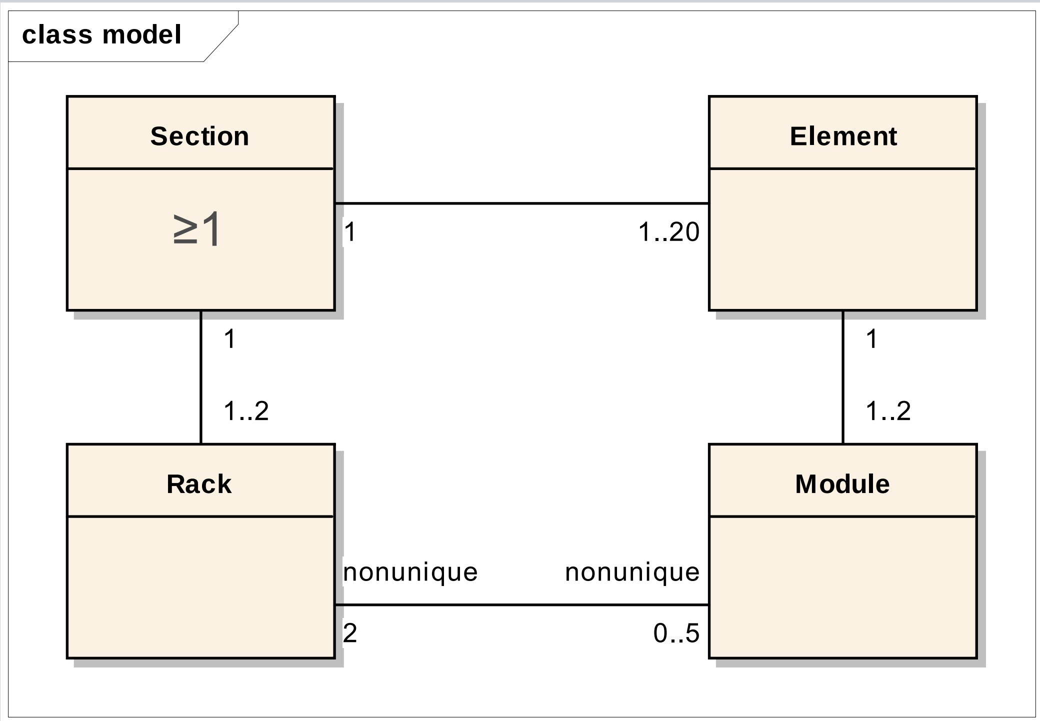
$$3 \cdot |A| \leq 2 \cdot |B|$$

- lower and upper bounds for the number of links (i.e. size of the relation)
- similar considerations for unique and mixed cases
- inequalities are complete and correct (solutions of inequalities correspond to valid object diagrams)

Theoretic example



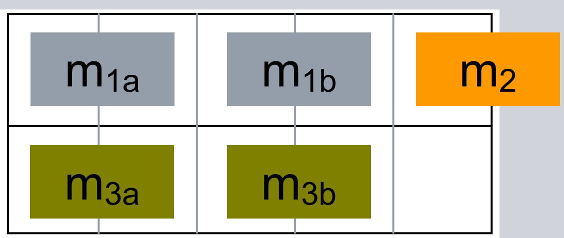
Practical example



given 3 elements

- e1 with 2 modules
- e2 with 1 module
- e3 with 2 modules

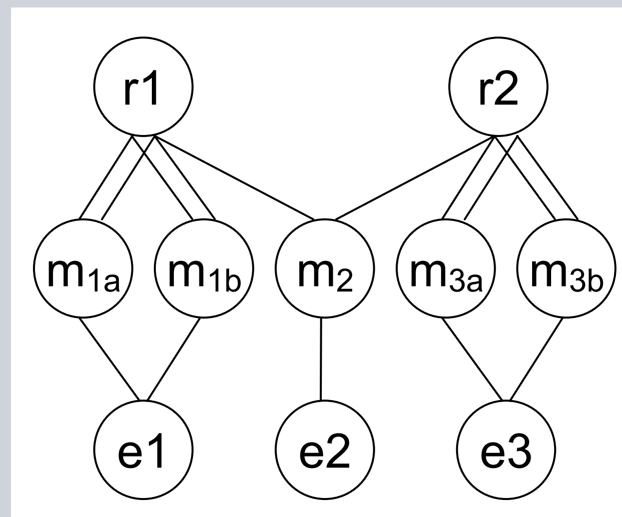
needs 2 racks ???



$2\ |Module| \leq 5\ |Rack|$

$|Module| = 5$

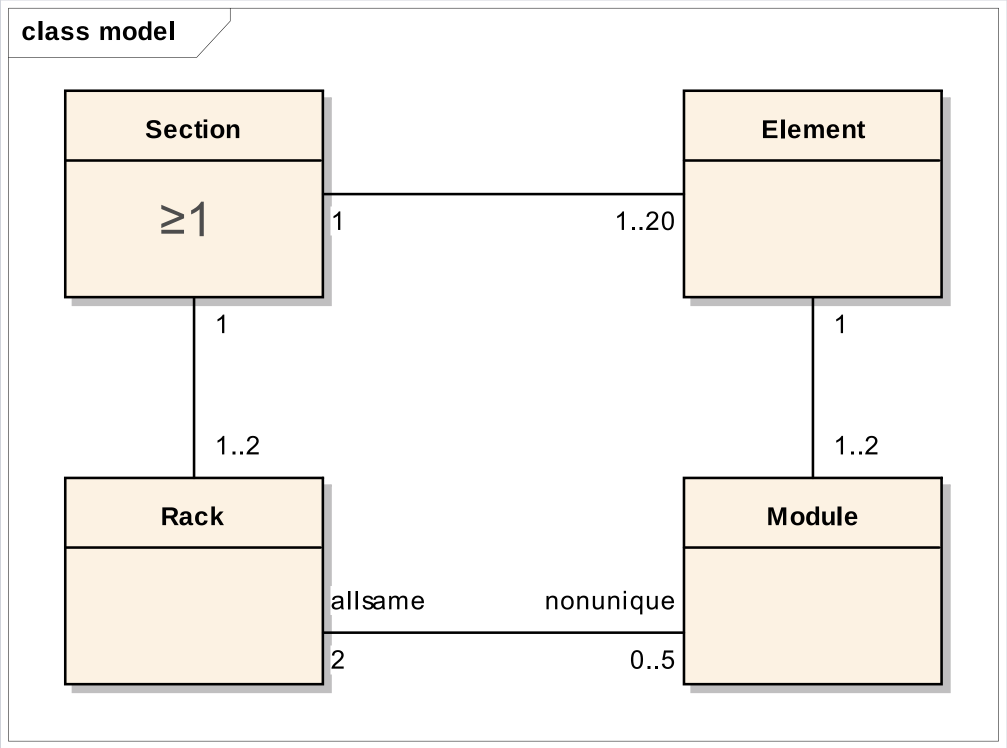
$\rightarrow\ 2 \leq |Rack|$



Equality constraints

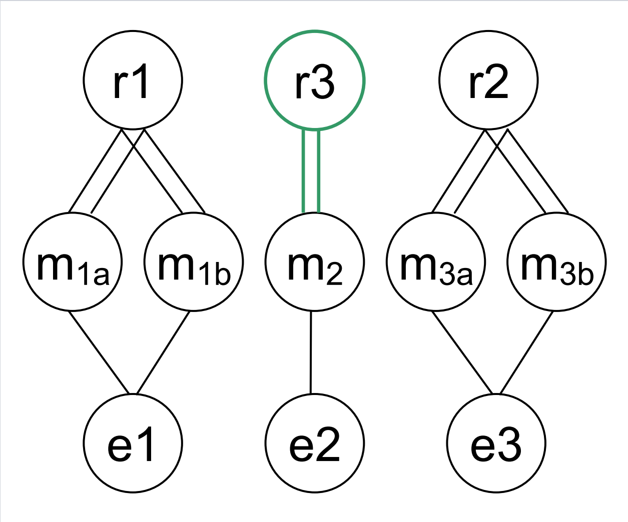
Additional constraints other than for multiplicities

- e.g. "all slots occupied by a module must belong to the same rack"



Extend UML diagram

- introduce tag "all-same"



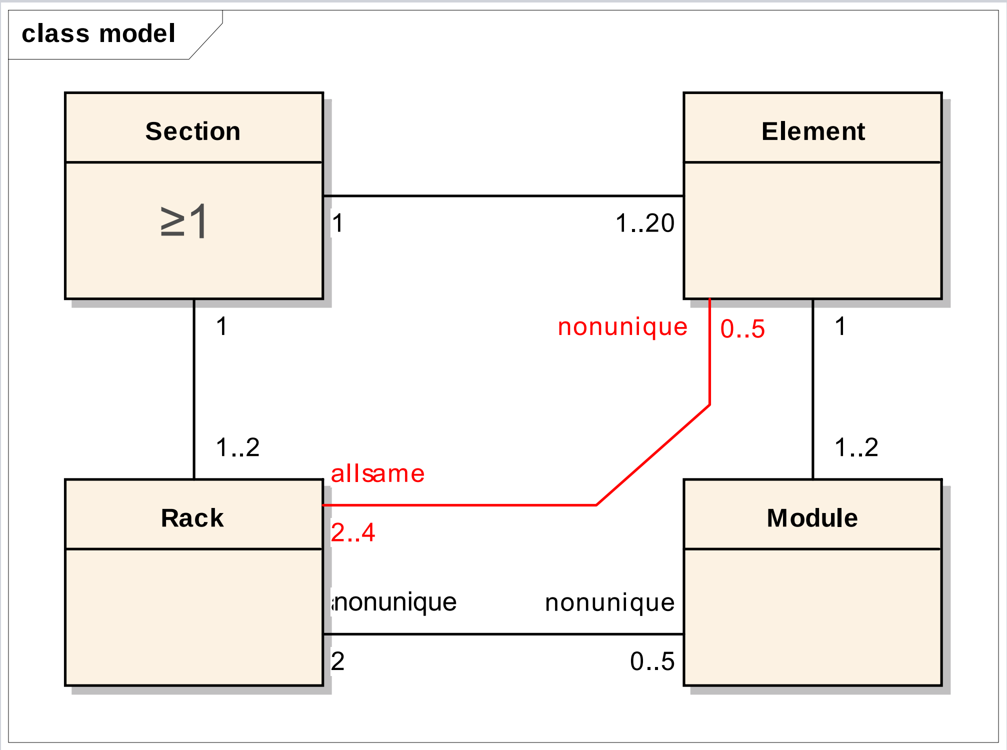
 July 2008 Siemens IT Solutions and Services

SIEMENS

Equality constraints (2)

Additional constraints other than for multiplicities

- e.g. "all slots occupied by a module must belong to the same rack"



Extend UML diagram

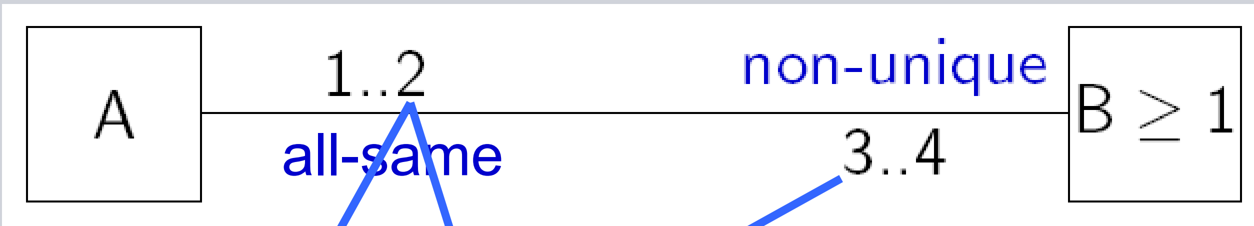
- introduce tag "all-same"

Inequalities:

- $|Module| \leq 2 \cdot |Rack|$
- (max. 5/2 = 2 modules fit in a rack)
- links = $2 \cdot |Module|$
- (each module requires exactly 2 links to a rack)

- e.g. "all modules of an element must be placed in the same rack"

Additional inequalities



Inequalities:

$2 > 0 \rightarrow \lceil 3/2 \rceil \cdot |A| \leq |B|$
$1 > 0 \rightarrow |A| \leq \lfloor 4/1 \rfloor \cdot |B|$

simplify to:

$2 \cdot |A| \leq |B| \leq 4 \cdot |A|$

Interpretation:

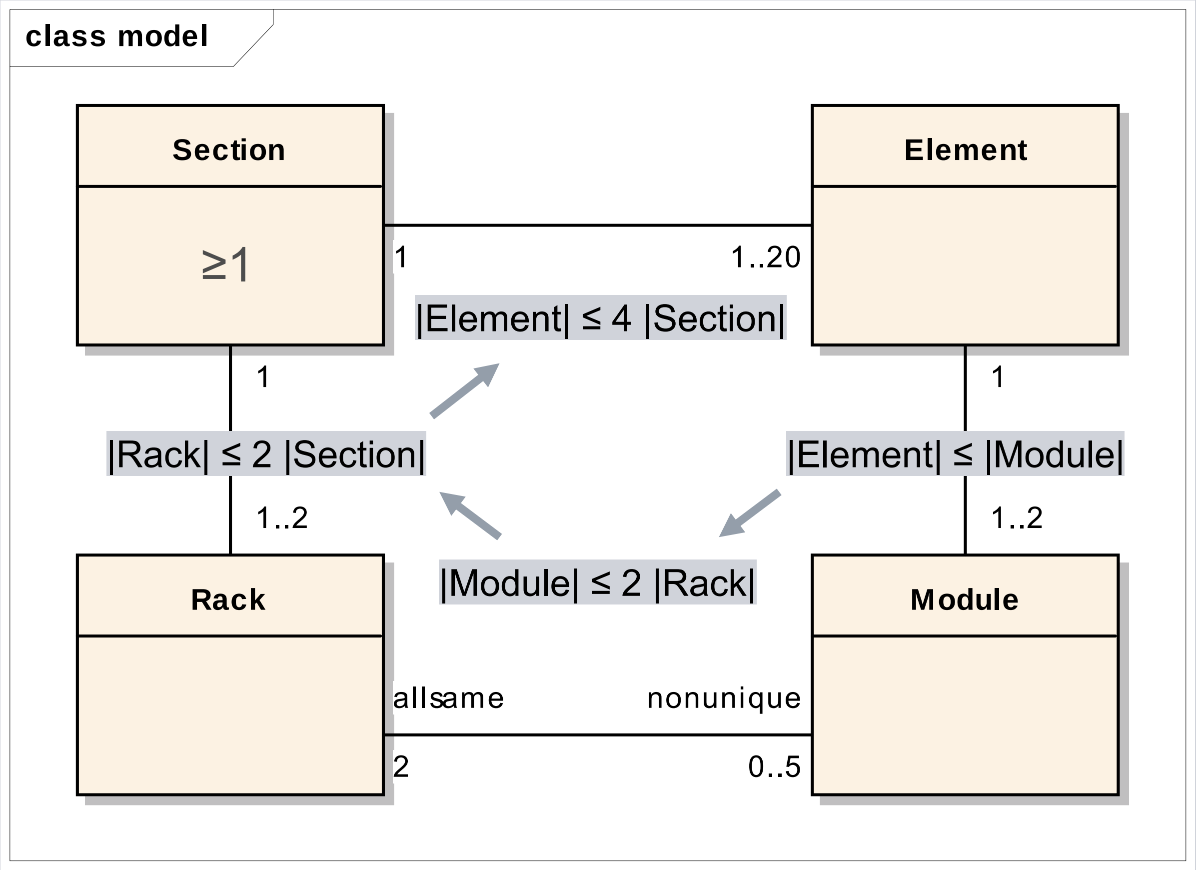
- "Slots" to be filled: 3..4
- Size in slots: 1..2
- 2 B (size 1 and 2) in an A
- up to 4 B of size 1

- Similar inequalities for other combinations (all-same/unique, etc.)
- Those additional inequalities do not corrupt the existing approach (computational properties, finding and mapping of solutions)

Reasoning about cardinalities

By-product of checking the consistency of inequalities:

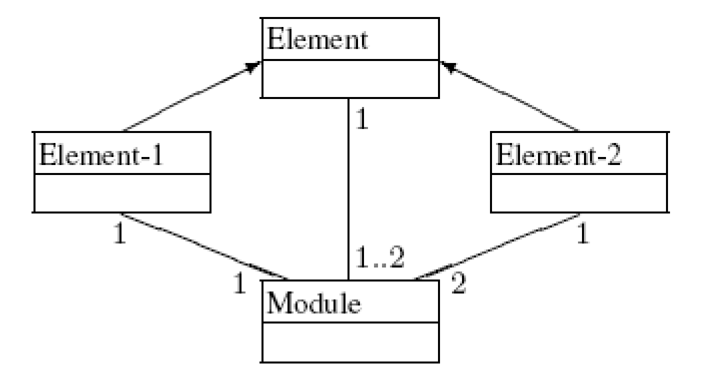
- Find stronger relationships (restricted cardinalities)
- for Section to Element: 1..4
- instead of 1..20



Type-specific cardinalities

Different multiplicities depending on the type of an element

Idea: use sub-classes



Additional equation:

$$1 \cdot |Element\text{-}1| + 2 \cdot |Element\text{-}2| = 1 \cdot |Module|$$

Open issue:

- extend existing algorithm
- computational complexity?

Remaining issues

Use the approach for the completion of partial configurations

- e.g. "how many more elements will fit into a section?"
- up to now: static analysis, finding (minimal) solutions from scratch

Make use of derived associations

- composed by a path over several classes, e.g. Element - Module - Rack
- used for coping with additional non-numeric constraints like "an element and the rack for it must belong to the same section"

Extend the approach for handling more types of constraints

- e.g. for numerical attributes or with additional diagram extensions
- if some constraints cannot be incorporated: generate-and-test solutions

Integration into configurators used in practice

- usability, performance
- up to now: academic prototypes

Contact

Thank you for your attention!

Andreas Falkner
Siemens AG Austria
Siemens IT Solutions and Services
PSE CVD IDB6
Erdberger Lände 26
A-1030 Wien, Austria
Phone: +43 51707 35932
E-Mail: andreas.a.falkner@siemens.com

Gernot Salzer
Technische Universität Wien
Computer Science Department
Theory and Logic Group
Argentinierstraße 8
A-1040 Wien, Austria
Phone: +43 1 58801 18541
E-Mail: salzer@logic.at