FROM IMPRECISE N-WAY MODEL MATCHING TO PRECISE N-WAY MODEL MERGING

ECMFA@STAF 2019

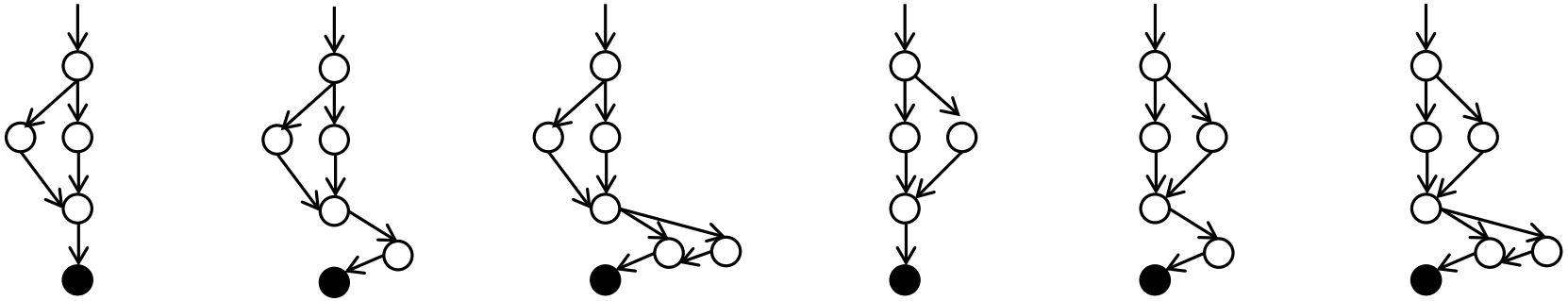
Eindhoven

Dennis Reuling, Malte Lochau, Udo Kelter

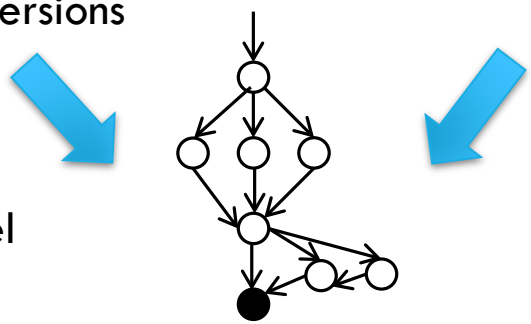
TECHNISCHE UNIVERSITÄT DARMSTADT

UNIVERSITÄT SIEGEN

N-WAY MODEL MERGING [RUBIN & CHECHIK, 2013]



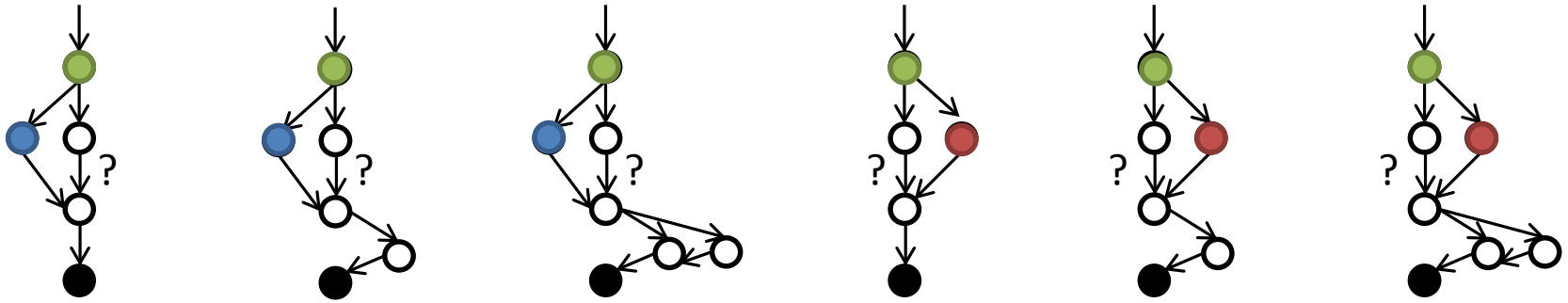
- N input model variants/versions



- one merged output model

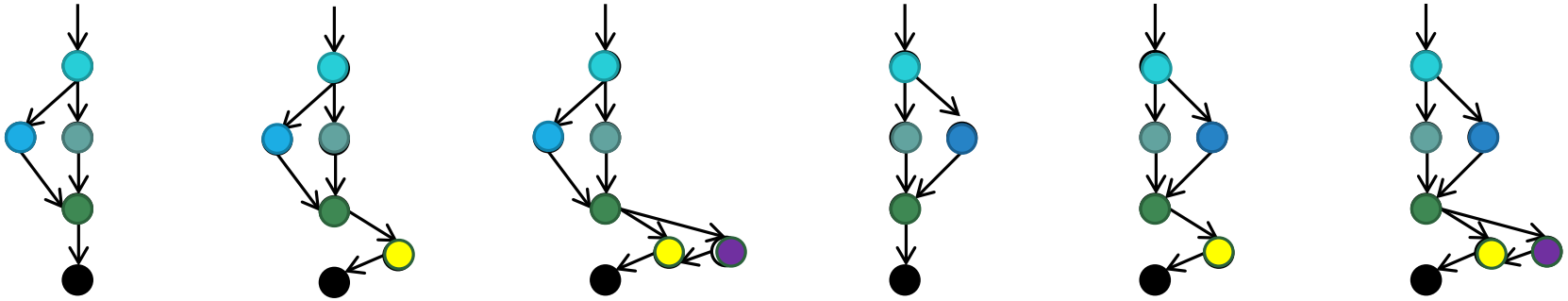
- Integration of N concurrent model variants/versions into one merged model
- Key technique for managing *software variability*
- 3 Steps: *Compare*, *Match*, *Merge*

COMPARE



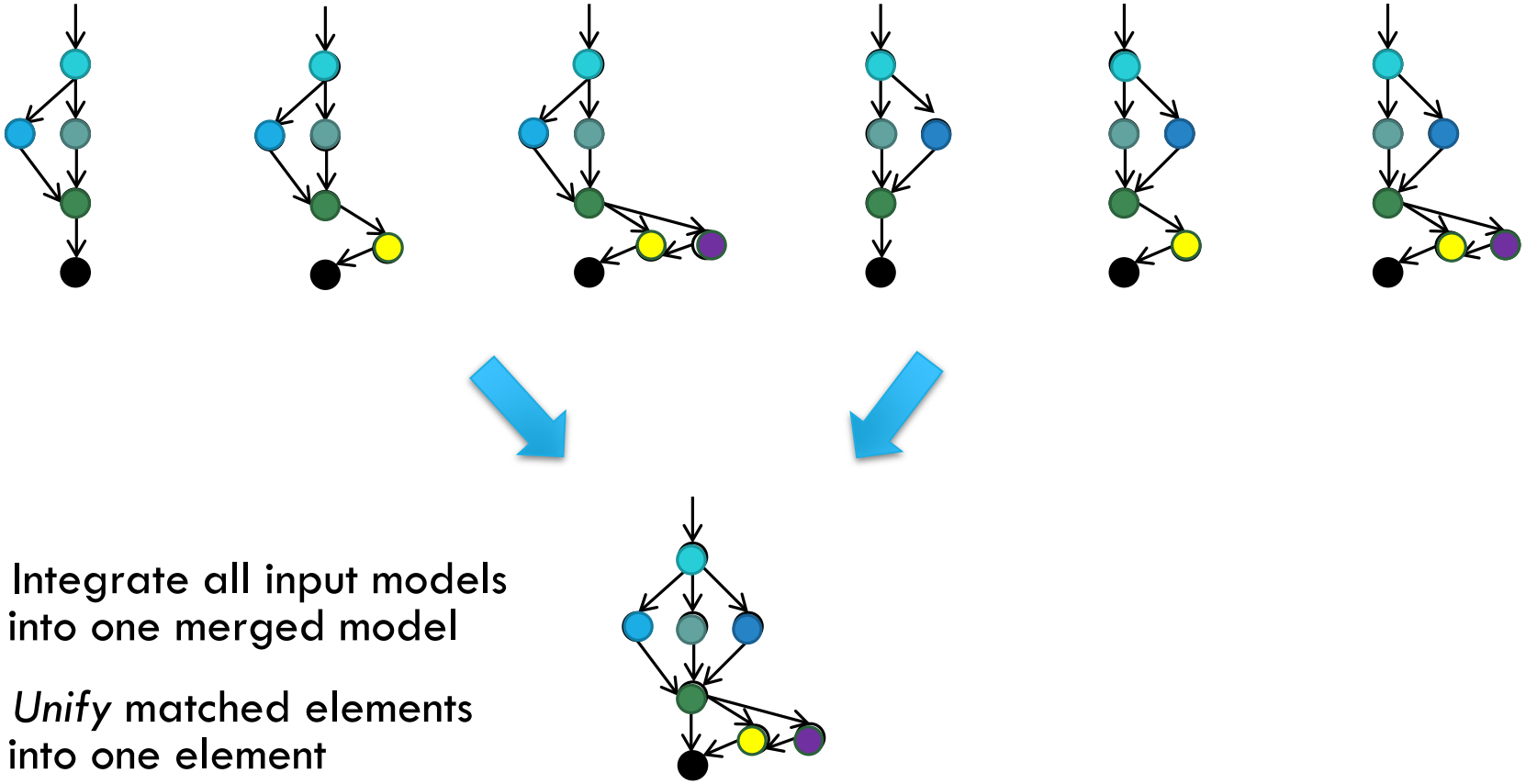
- Set of N-tuples $t = (e_1, e_2, \ldots, e_k) \in T, 1 \leq k \leq N$:
  all possible *groupings* of model elements from different input models
- Function *compare* : $T \rightarrow [0,1]$:
  $compare(t') > compare(t'')$ if elements grouped in t' are *more similar* than elements grouped in t''

MATCH



- Subset $M \subseteq T$ is a *match* if each model element from each input model occurs in exactly one N-tuple $t \in M$
- Match M is *minimal* (*optimally precise*) if for every match $M' \subseteq T$ it holds that $\left(\sum_{t \in M} compare(t)\right) \geq \left(\sum_{t' \in M'} compare(t')\right)$

MERGE

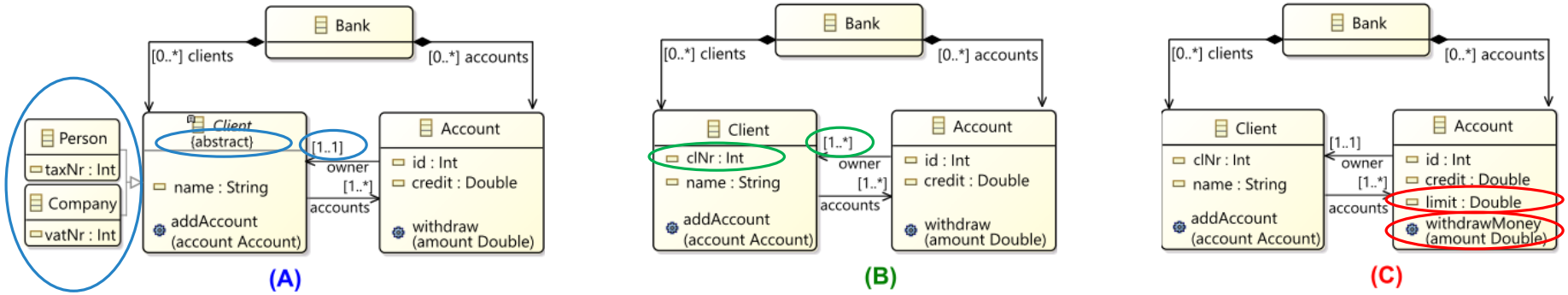


- Integrate all input models into one merged model
- *Unify* matched elements into one element

CHALLENGES

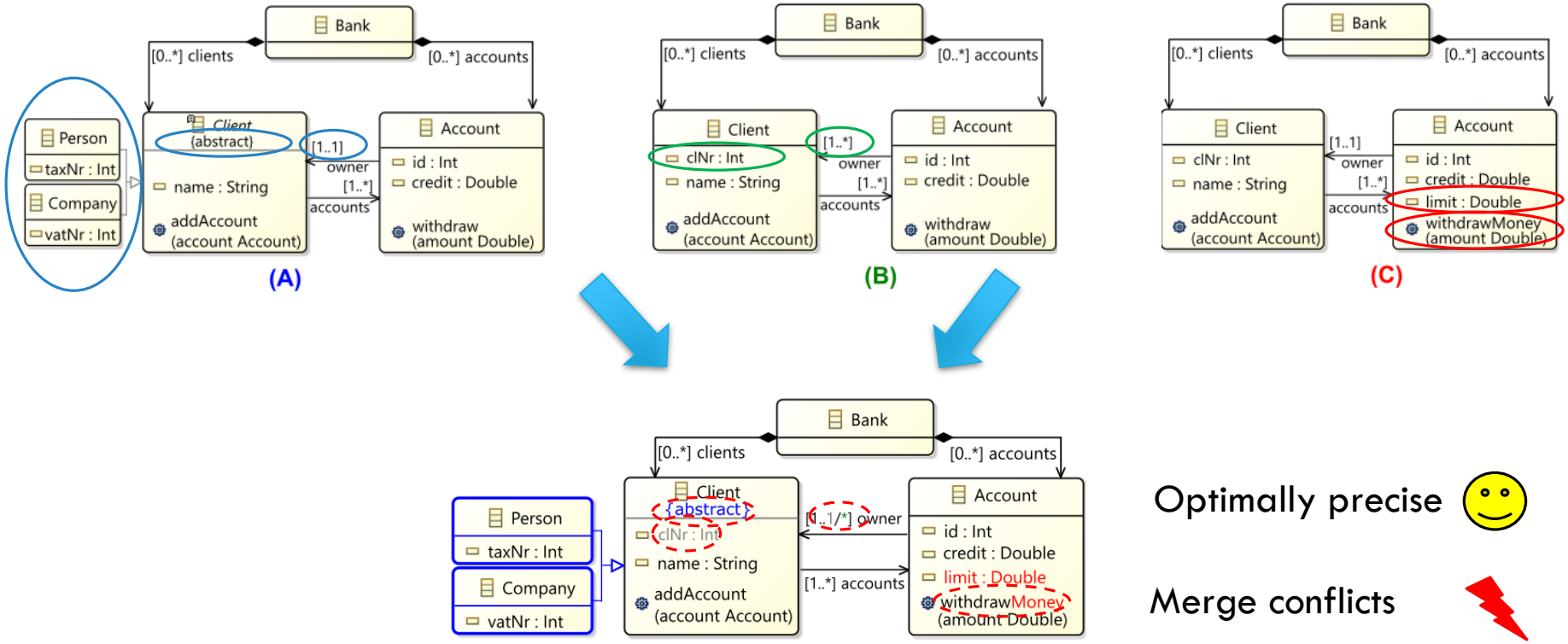
- *Compare*: Similarity metrics must be sufficiently precise to ensure precise matches (yet being computationally tractable)
  [Holthusen et al. 2014], [Martinez et al., 2015], [Strüber et al., 2016], [Wille et al. 2017], [Reuling et al., 2019]
- *Match*: Finding optimally precise matches is NP-hard
  [Rubin & Chechik, 2013]
- *Merge:* Unify-merge infeasible for realistic modeling languages, whereas realistic merge-operators often decrease precision and/or require manual conflict resolution
  [Wieland et al., 2012], [Debreceni et al., 2016]

EXAMPLE: CLASS DIAGRAMS



- *Name-based matching:* only match elements with identical types and names
- *Structure-based matching:* take context information into account (e.g., similarity of source/target/nesting elements)
- *Identity-based matching:* utilize internal element identifiers to match similar elements even after changes of properties (e.g., renamings)

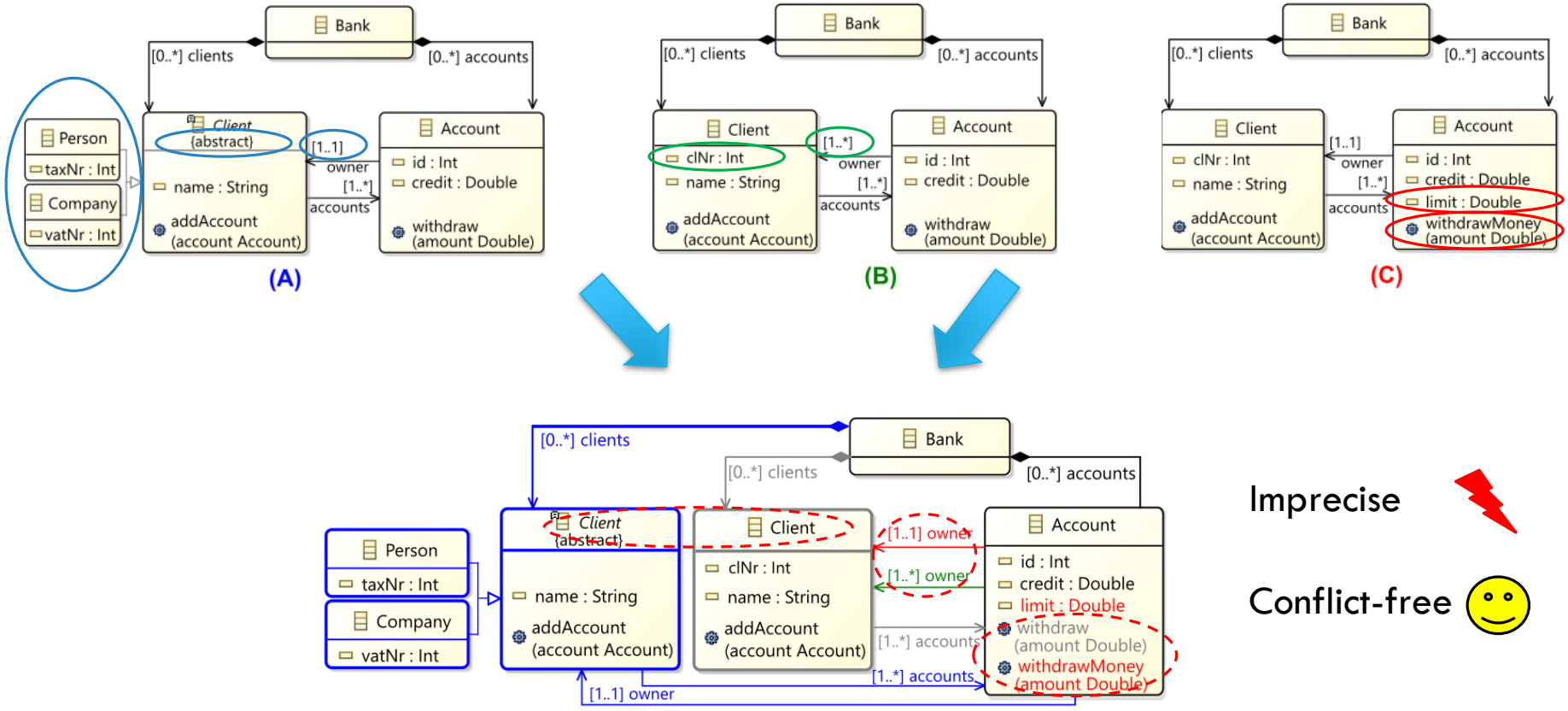
MERGE PRECISION AND MERGE CONFLICTS



Optimally precise

Merge conflicts

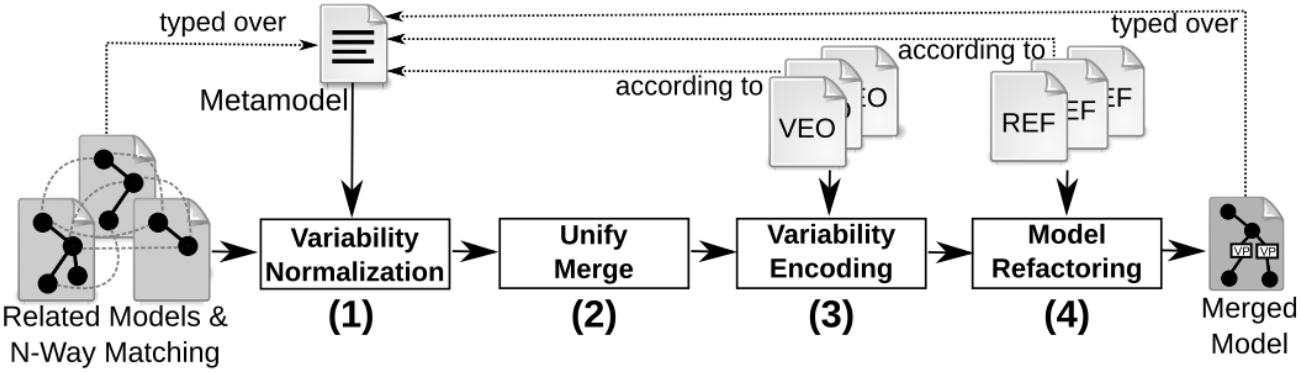
MERGE PRECISION AND MERGE CONFLICTS



Imprecise

Conflict-free

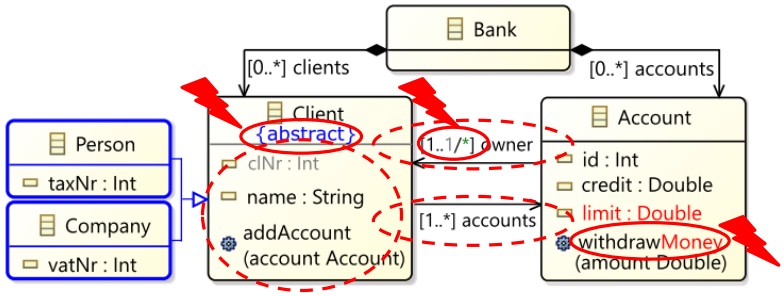
OUR APPROACH



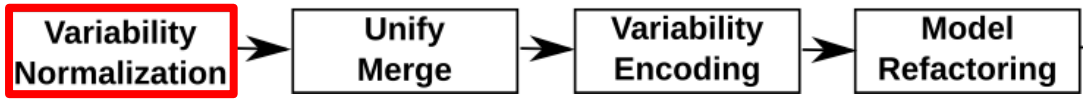
Combine *variability-encoding* operators and *model-refactoring operators* to improve merge precision of (arbitrarily) imprecise input matches

- Variability-encoding operators embed variability meta-information into the merged model for automated conflict resolution
- Model-refactoring operators are applied to merged model to identify and unify further similarity among initially unmatched elements

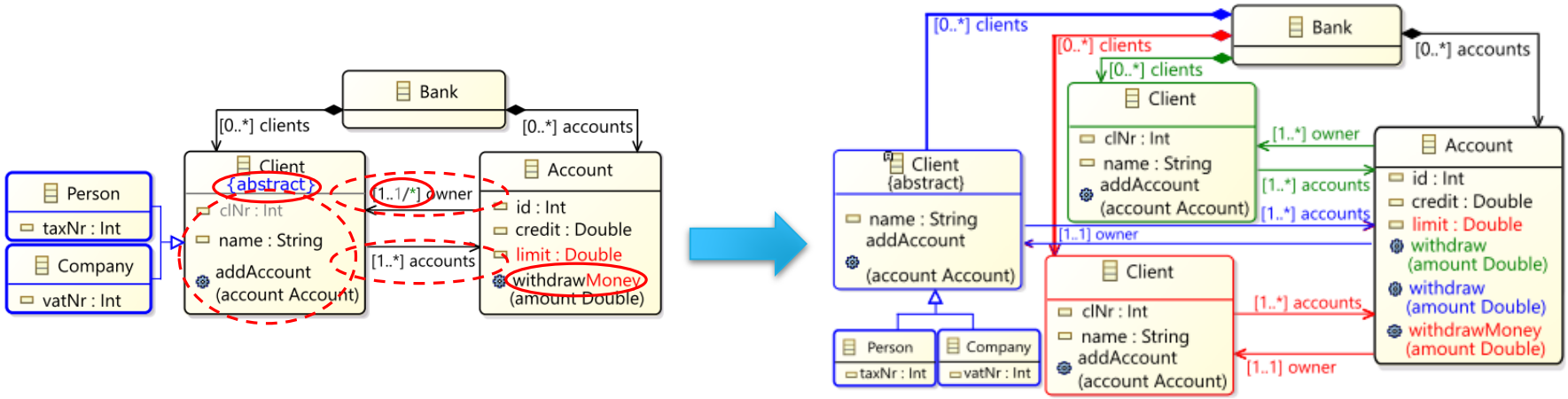
VARIABILITY NORMALIZATION



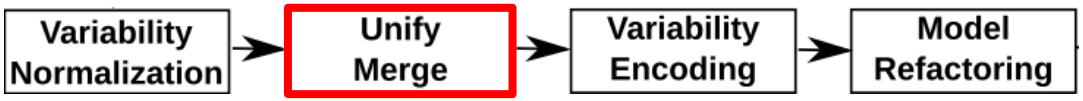
- *Normalization rules* reduce precision of input match M until unify-merge yields a merged model that conforms to the meta model
- Split all conflicting tuples $t \in M$ into $|t|$ singleton tuples and recursively split further tuples related to t



UNIFY MERGE

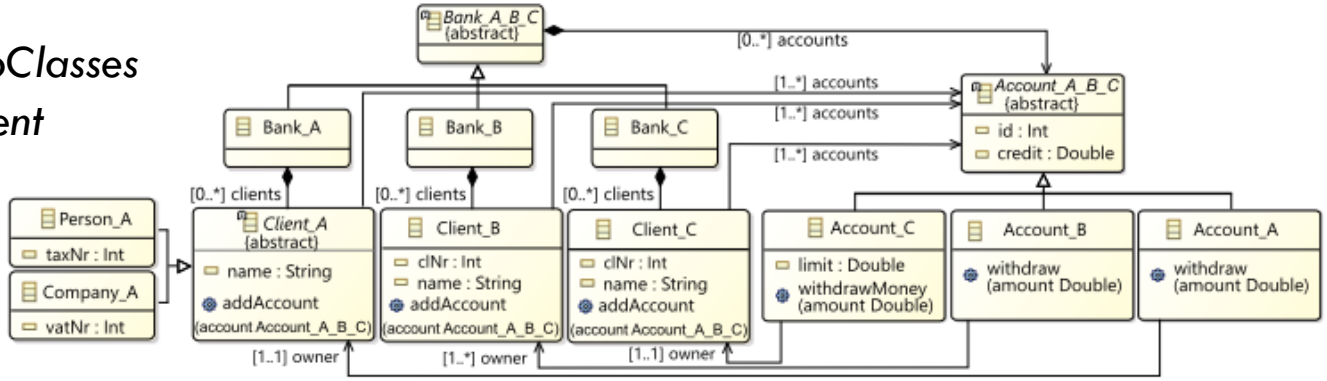


- Apply *unify-merge operator* to the normalized match M'
- Tag all tuples $t \in M'$ with *variability meta-information*

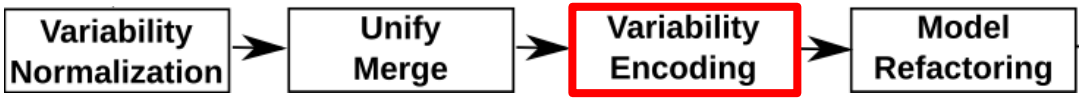


VARIABILITY ENCODING

- *PushPropertiesToSubClasses*
- *RenameNamedElement*

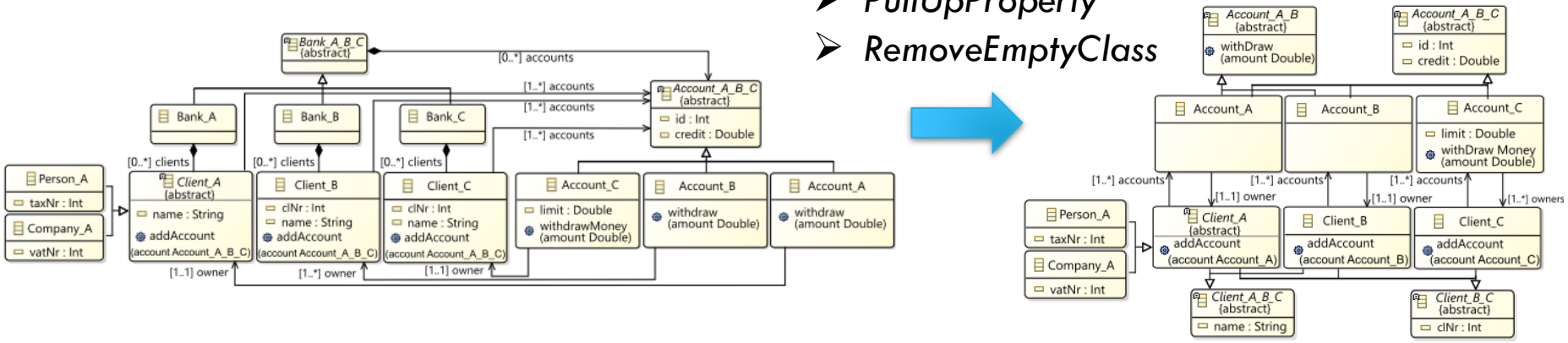


- *Variability-encoding operators* (VEO) integrate variability meta-information into the unify-merged model
- Class Diagrams: use class inheritance to push similar properties into sub classes and apply class renaming to mark variant-specific elements

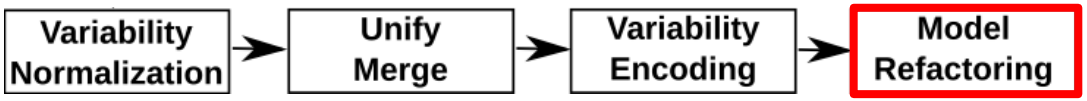


MODEL REFACTORING

- *ExtractSuperClass*
- *PullUpProperty*
- *RemoveEmptyClass*



- Default *model-refactoring operators* (REF) perform semantic-preserving model transformations
- Identify and eliminate duplicated elements by employing language-specific reuse constructs to further improve merge precision



EXPERIMENTS

- **RQ1:** How precise is the merged model resulting from our metholodolgy as compared to arbitrarily (im-)precise input matchings?
- **RQ2:** Does our approach scale to large input models with arbitrarily (im-)precise input matchings?

			#Elements			
System	Oracle	#Models	Class	Attr	Assoc	Method
PPU	Optimal	13	418	309	428	232
BCMS	Approx.	15	948	635	184	1675
ArgoUML	Optimal	7	12270	22172	4851	89566

- Oracles for precision:

(1) *optimal*: identity-based matching of generated/extracted artifacts
(2) *approximated*: provided manually by developers

$$Prec(M) = \frac{TP}{TP + FP}$$

EFFECTIVENESS

System	Matching Approach	Matching Precision	# Class Members MAT	# Class Members UNI	# Class Members REF	Merging Precision UNI	Merging Precision REF
PPU	ID	**1.00**	69	125	79	0.55	0.87
	SIM	0.67	93	323	79	0.21	**0.87**
	EMFC	0.15	451	451	79	0.15	**0.87**
	OR50	0.48	171	171	91	0.40	**0.76**
	Empty	0.00	686	686	662	0.10	**0.11**
bCMS	ID	1.00	604	2018	339	0.30	**1.78**
	SIM	0.51	1023	1245	332	0.48	**1.81**
	EMFC	0.29	428	2058	341	0.29	**1.77**
	OR50	0.39	2019	2019	339	0.29	**1.78**
	Empty	0.00	2310	2310	2310	0.26	**0.26**
ArgoUML	ID	**1.00**	16917	16952	16301	0.98	0.99

- Precision of (near) optimally precise input matches is preserved
- Precision of imprecise input matches is improved up to a factor of 1.81

EFFICIENCY

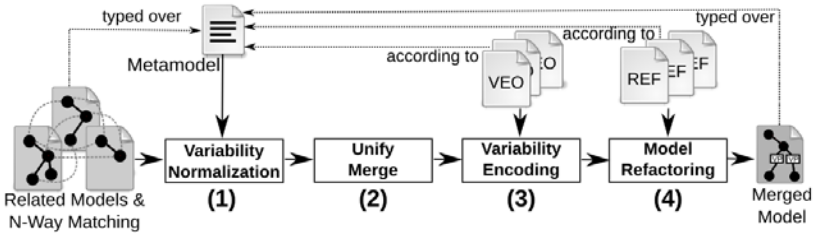
System	Approach	# Classes			Time Consumption for each Step (s)				
		UNI	ENC	REF	(1)	(2)	(3)	(4)	Sum
PPU	ID	70	194	263	2.9	0.3	4.2	3.5	**11**
	SIM	55	219	476	4.3	0.3	12.9	62.4	**80**
	EMFC	37	205	579	4.3	0.3	29.9	158.0	**192**
	OR50	70	202	305	0.4	0.4	6.4	7.8	**15**
	Empty	418	418	439	2.9	0.4	257.9	795.8	**1057**
bCMS	ID	620	1085	2902	12.6	0.6	472.2	11.0	**496**
	SIM	436	916	1946	9.8	0.4	223.2	7.1	**241**
	EMFC	633	1083	2940	13.8	0.6	461.6	10.7	**487**
	OR50	685	1085	2902	2.5	0.6	470.8	11.0	**485**
	Empty	948	1109	1089	5.2	0.6	718.0	19.0	**743**
ArgoUML	ID	1782	1836	2732	123.8	23.5	2221.5	11916.1	**14285**

- Overall CPU time ranges from 11 seconds to 3 hours, where most CPU time (90%) is consumed by encoding and refactoring
- Computational effort depends on both the model size and precision of input matches

CONCLUSION AND OUTLOOK

- Our methodology preserves precision of near optimal matchings and remarkably improves precision of (realistically) imprecise input matches
- Our metholodogy scales to input models of realistic sizes (while aiming at batch scenarios rather than online usage)
- We plan to generalize our approach to other modeling languages and domains
- We further plan to exploit variability information to better control efficiency/effectiveness trade-offs (e.g., prioritization of refactorings of elements shared by many variants, …)

Thank You!



http://pi.informatik.uni-siegen.de/projects/variance/ecmfa19