technische universität dortmund

Selected Results and Related Issues of Confidentiality-Preserving Controlled Interaction Execution

Joachim Biskup

Technische Universität Dortmund
Germany

FoMSESS 2016, Essen, February 2016

technische universität dortmund

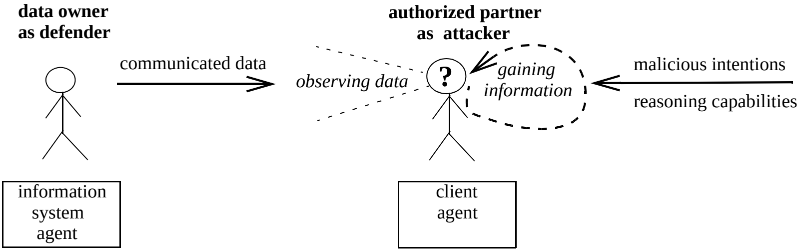
Acknowledgements

- **Basic Works:**
  - Sicherman/deJonge/van de Riet: ACM TODS 83
  - Bonatti/Kraus/Subrahmanian: IEEE TKDE 95
  - Biskup: DKE 00
- **Co-authors:**
  - *main cooperation partner since 2001:* Piero Bonatti
  - *researchers (Ph.D. students) at TU Dortmund:* Ralf Menzel, Torben Weibert, Lena Wiese, Jan-Hendrik Lochner, Cornelia Tadros, Marcel Preuß
  - *Diploma/Master students at TU Dortmund:* Martin Bring, Michael Bulinski, Christine Dahn, Katharina Diekmann, Christian Gogolin, Dirk Schalge, Jens Seiler
  - *externals:* David Embley, Clemente Galdi, Sven Hartmann, Lan Li, Sebastian Link, Luigi Sauro
- **Funding:** DFG grants Bi 311/12_1+2 and SFB 876/A5

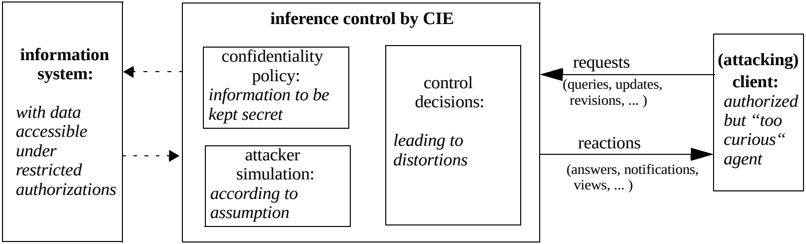
Table of Contents

Inference Control for Logic-Oriented Information Systems

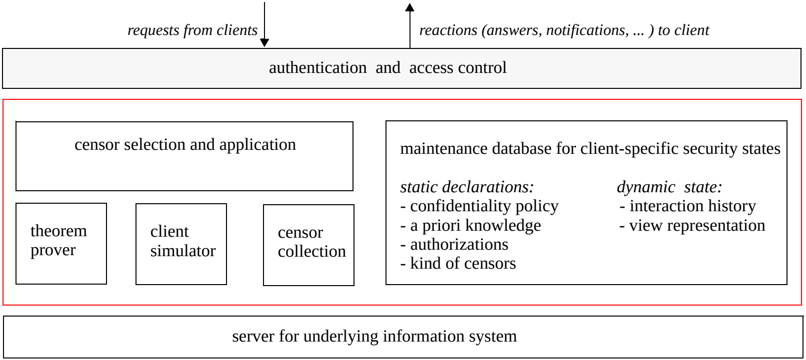
Attacker-Defender Situation Underlying Inference Control



Inference Control by Controlled Interaction Execution



Rough Architecture of Controlled Interaction Execution



A Simple Propositional Framework

Example

instance: health record of some patient Lisa,
represented by propositional atoms,
denoting the "true part" of an interpretation:

$\{ \texttt{brokenArm}, \texttt{brokenLeg}, \texttt{lowWorkload}, \texttt{highCosts} \}$

confidentiality policy:

$\{ \texttt{lowWorkload} \wedge \texttt{highCosts} \}$

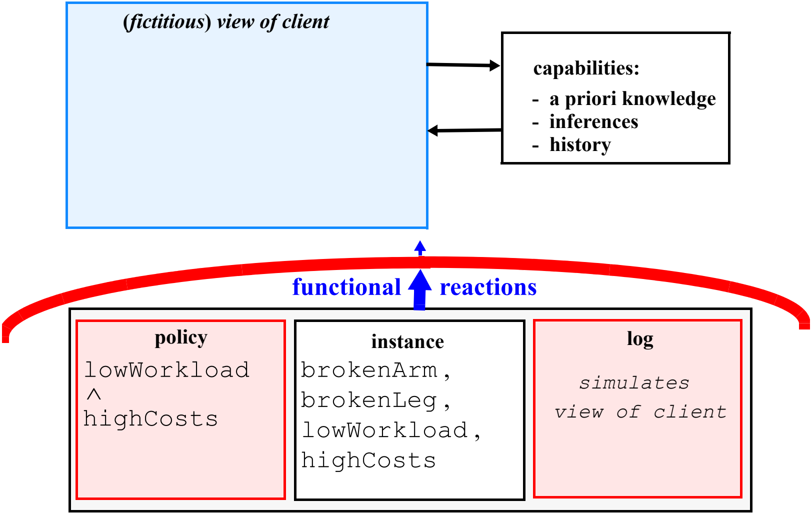
client's a priori knowledge:

$\{ \texttt{brokenArm} \Rightarrow \texttt{lowWorkload},$
$\texttt{brokenLeg} \Rightarrow \texttt{highCosts} \}$

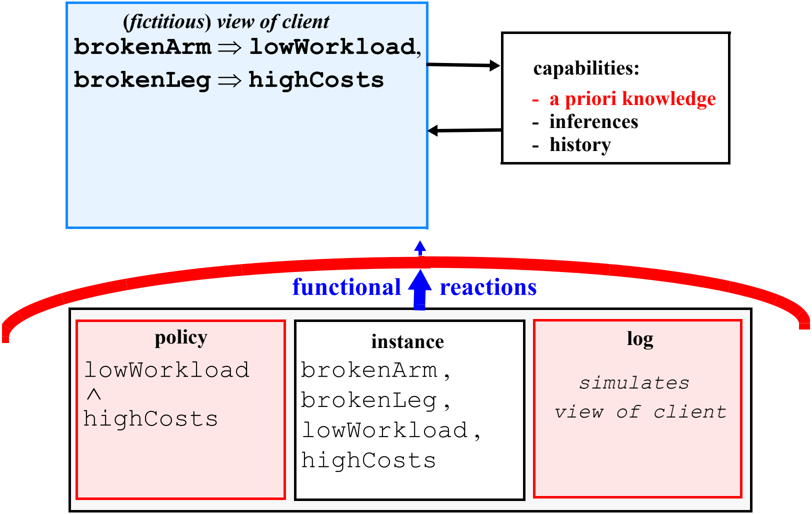
sequence of queries:

$\langle \texttt{brokenArm}(?), \texttt{brokenLeg}(?) \rangle$

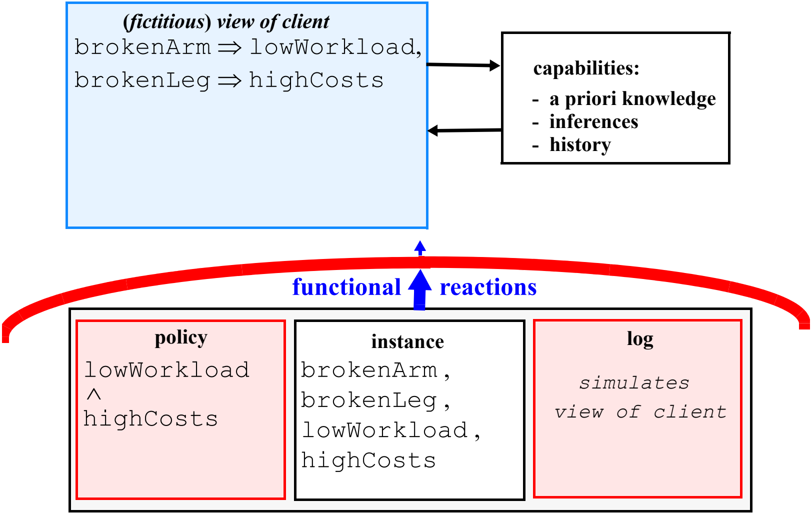
Refusal approach for queries ⟨ ⟩



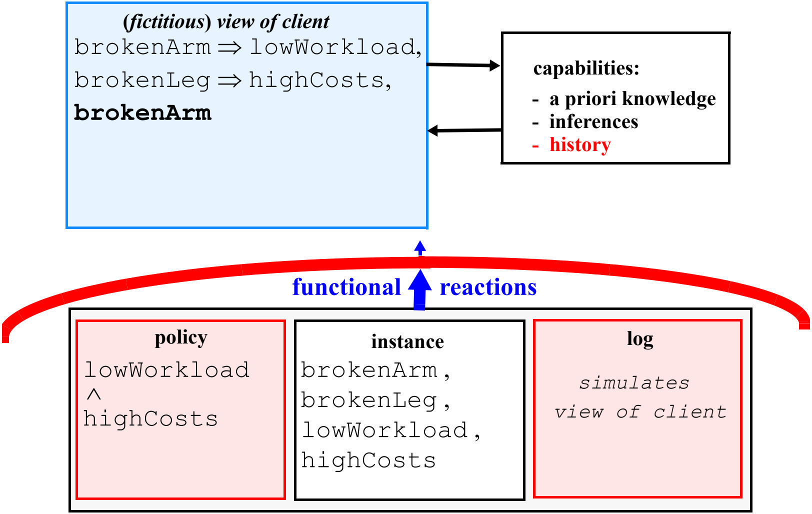
Refusal approach for queries ⟨ ⟩



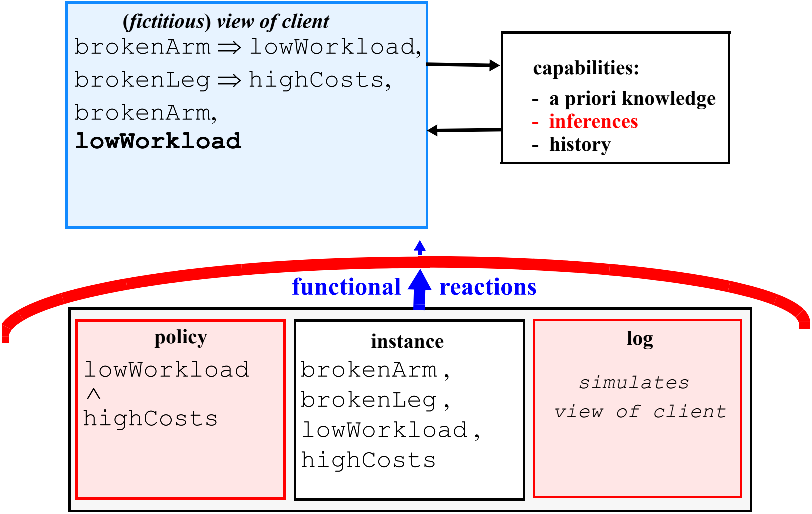
Refusal approach for queries ⟨ brokenArm(?) ⟩



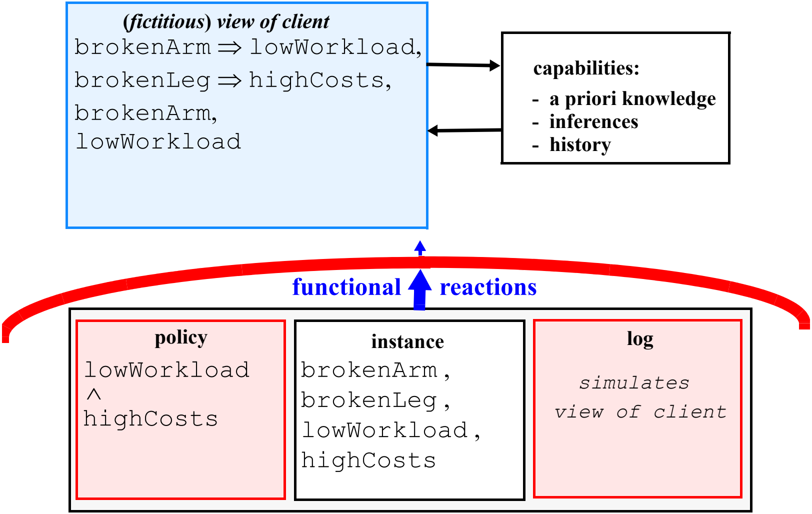
Refusal approach for queries ⟨ brokenArm(?) ⟩



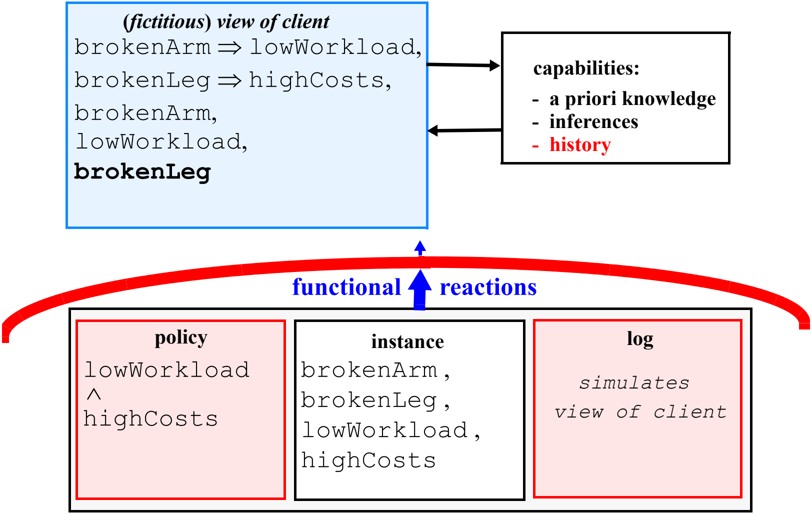
Refusal approach for queries ⟨ brokenArm(?) ⟩



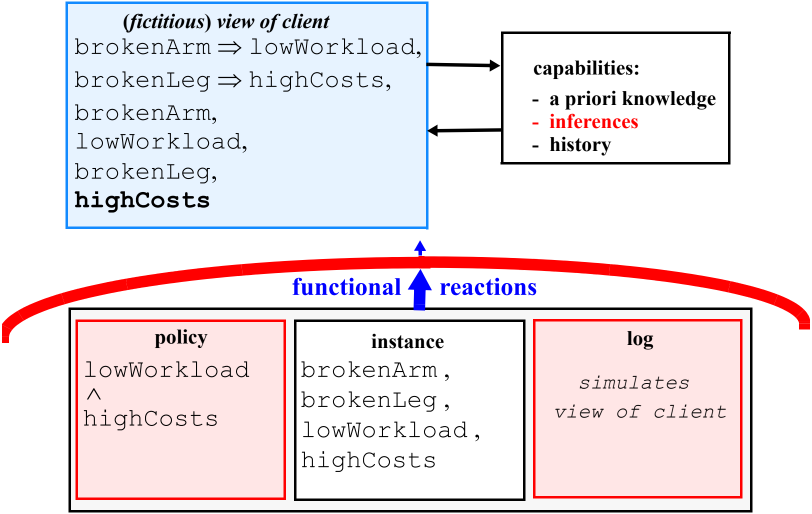
Refusal approach for queries ⟨ brokenArm(?) , brokenLeg(?) ⟩



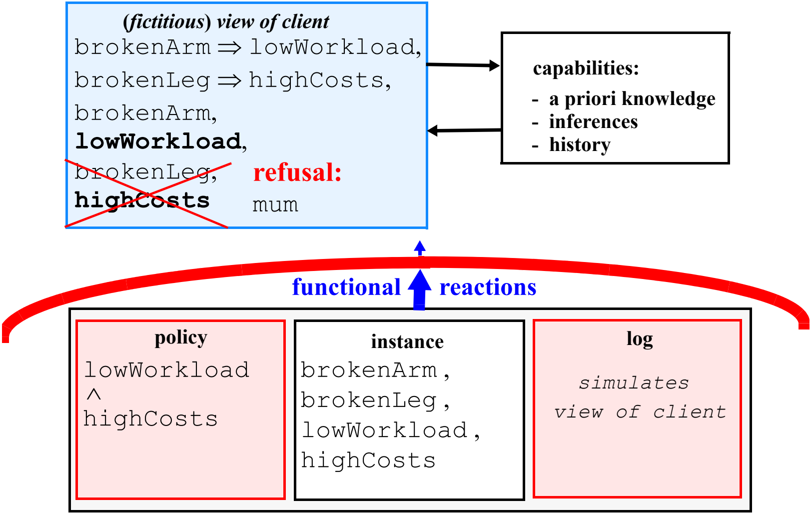
Refusal approach for queries ⟨ brokenArm(?) , brokenLeg(?) ⟩



Refusal approach for queries ⟨ brokenArm(?) , brokenLeg(?) ⟩



Refusal approach for queries ⟨ brokenArm(?) , brokenLeg(?) ⟩



Refusal approach for queries ⟨ brokenArm(?) , brokenLeg(?) ⟩

(fictitious) view of client

brokenArm ⇒ lowWorkload,
brokenLeg ⇒ highCosts,
brokenArm,
lowWorkload,
~~brokenLeg,~~
~~**highCosts**~~

refusal:
mum

capabilities:

- **a priori knowledge**
- **inferences**
- **history**

problems:
meta-inferences, ...

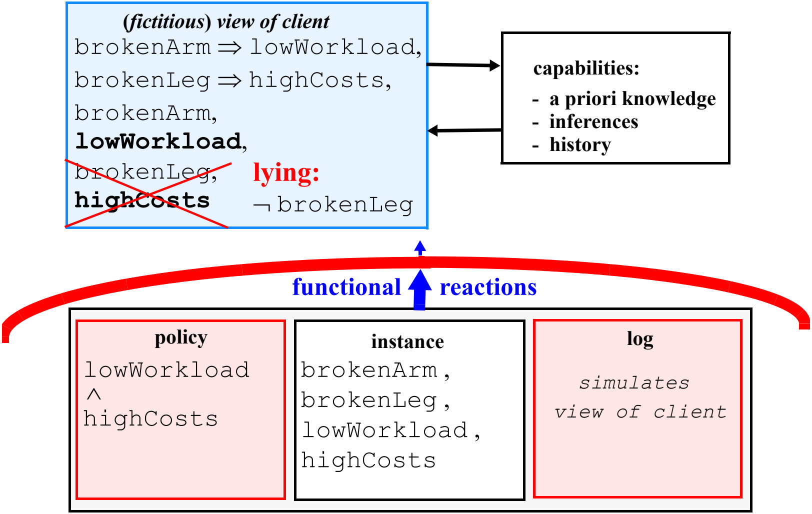
functional reactions

policy
lowWorkload
∧
highCosts

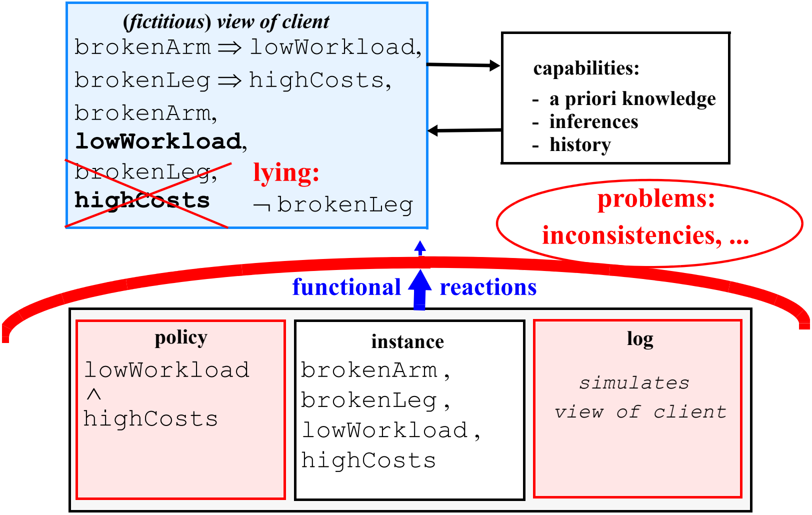
instance
brokenArm ,
brokenLeg ,
lowWorkload ,
highCosts

log
simulates view of client

Lying approach for queries ⟨ brokenArm(?) , brokenLeg(?) ⟩



Lying approach for queries $\langle$ brokenArm(?) , brokenLeg(?) $\rangle$



technische universität dortmund

Complete Propositional Information System

- **framework:** classical (finite) propositional logic
- **stored instance:** a (semantic) model (truth assignment) *db*
- **query:** any sentence φ
- **query evaluation:** truth evaluation
  $eval(db, \varphi) := \texttt{if } db \models \varphi \texttt{ then } \varphi \texttt{ else } \neg\varphi$
- **access rights:** any sequence of queries $\varphi_1, \ldots, \varphi_i, \ldots$
- **confidentiality policy:** set of sentences *psec*
- **a priori knowledge:** set of sentences *prior*

Functional Interaction Processing – Without Control

round wise, at each point in time i:

- **client:** submits a query request φ_i
- **system:** returns answer reaction $eval(db, \varphi_i)$
- **client:**
  - maintains current "syntactic" view $synView_i := prior \cup \{eval(db, \varphi_1), \ldots, eval(db, \varphi_i)\}$
  - together with all sentences entailed, leading to current "semantic" view $semView_i$
- **system:** might simulate the client

Introducing Control

- **without control:**
  - $synView_i := prior \cup \{eval(db, \varphi_1), \ldots, eval(db, \varphi_i)\}$
  - $semView_i := \{ \ \varphi \ | \ synView_i \models \varphi \ \}$
  - client can obtain complete knowledge about the instance
- **confidentiality policy:**
  - syntactically: declared as *sentences* (called *potential secrets*)
  - semantically: independently of the actual truth value, for the client it should *always* appear to be possible that the sentence is *not* true
- **enforcement:**
  - censor *minimally distorts* the correct truth evaluation $eval(db, \varphi_i)$ into a *controlled answer sentence* ans_i
  - accordingly: now, $synView_i := prior \cup \{ans_1, \ldots, ans_i\}$

Crucial Impact of Distortions

- **purely functionally, without control:**
  the semantic view is obtained as
  closure of the syntactic view under entailment
- **with inference control, facing potential distortions:**
  the semantic view can only be determined by
  considering the details of the censor

Main Challenges for the Client

- why did the censor return the "verbatim" answer ans_i to the query about the truth evaluation of φ_i?
- which possible instances of the information system do lead to that verbatim answer?
- which of the two possible truth evaluations of φ_i do cause that verbatim answer?
- in most general mathematical terms, how to invert the censor function on the function values observed as verbatim answers?

Guidelines for Censor Construction

- express any answer as a *sentence* such that
  – the answer looks like "being informative"
  – the syntactic view $synView_i$ remains consistent
- maintain a suitable *security invariant*, including in particular:

$$\text{for all } \psi \in psec : synView_i \not\models \psi$$

- *computationally* check such *entailments*
  – and possibly further or more general ones –
  to determine the need of a distortion
- form the *answer sentence* such that
  – from the client's point of view –
  it remains *indistinguishable*
  what the correct answer would have been

Basic Refusal Approach

- check whether the correct answer is already known
- ensure indistinguishability by instance-independence
- inspect both the query sentence φ_i and its negation $\neg\varphi_i$

$ans_i :=$
`if` $synView_{i-1} \models eval(db, \varphi_i)$
`then` $eval(db, \varphi_i)$ % the correct answer
`else` `if` (exists ψ)[$\psi \in psec$ and
($synView_{i-1} \cup \{\varphi_i\} \models \psi$ or $synView_{i-1} \cup \{\neg\varphi_i\} \models \psi$)]
`then` $(eval(db, \varphi_i) \vee \neg eval(db, \varphi_i))$ % a tautology, or `mum`
`else` $eval(db, \varphi_i)$ % the correct answer

Basic Lying Approach

- only inspect the correct answer
- ensure consistent answers
- employ stronger violation condition:
  protect the disjunction of all policy elements

$ans_i :=$
if $synView_{i-1} \cup \{eval(db, \varphi_i)\} \models \bigvee_{\psi \in psec} \psi$
then $\neg eval(db, \varphi_i)$ % a lie
else $eval(db, \varphi_i)$ % the correct answer

Basic Combined Approach

- first inspect the correct answer
- if it would lead to a direct violation:
  – in particular to ensure consistent answers –
  additionally inspect its negation
- if the negation would be harmless: return it as a lie
- otherwise: – to escape a hopeless situation –, refuse

$ans_i :=$
$\texttt{if}$ (exists ψ)[$\psi \in psec$ and $synView_{i-1} \cup \{eval(db, \varphi_i)\} \models \psi$]
$\texttt{then}$ $\texttt{if}$ (exists ψ)[$\psi \in psec$ and $synView_{i-1} \cup \{\neg eval(db, \varphi_i)\} \models \psi$]
$\texttt{then}$ $(eval(db, \varphi_i) \vee \neg eval(db, \varphi_i))$ % a tautology, or mum
$\texttt{else}$ $\neg eval(db, \varphi_i)$ % a lie
$\texttt{else}$ $eval(db, \varphi_i)$ % the correct answer

Result: Effectiveness of basic censors for query sequences

- **Framework:** propositional (and any similar ones)
- **Interaction:** *sequence of query answering*
- **Control:**
  basic censors for *refusal*, *lying*, or the *combination*
- **Claim:**
  confidentiality preserving :
  for each actual instance, for each confidentiality policy,
  for each potential secret in that policy,
  for each assumed a priori knowledge,
  and for each sequence of query sentences,
  there exists an alternative instance that
  - satisfies the a priori knowledge as well
  - generates the same controlled answer sentences
  - but does not satisfy the potential secret

technische universität dortmund

Structure of Proofs

- consider any potential secret ψ
- at each point in time *i*, by the security invariant:
  there exists an "alternative instance" that satisfies
  - the current syntactic view
  - but not ψ
- by induction up to *i*:
  the actual instance and the alternative instance
  generate the same controlled answers
- thus, the "alternative instance" witnesses the *possibility*
  that ψ is *not* valid

Result: Effectiveness of basic censors for published views

- **Framework:** propositional (and any similar ones)
- **Interaction:** view publishing
- **Control:**
  limit of controlled answers
  to any exhaustive query sequence,
  generated by a basic censor for
  refusal, *lying*, or the *combination*
- **Claim:**
  confidentiality preserving

An Abstract Framework
recall last FoMSESS, Bremen, 2015

A Relational Framework

Relational Information System

- **framework:** typed relational model of data
  based on classical *first-order* logic with *DB-semantics*
- **stored instance:** a (semantic) model *db*
  represented by finite relations
  as Herbrand interpretation of predicate symbols/relation names
- **query:**
  closed: any safe sentence φ
  open: any safe formula $\varphi(x_1, \ldots, x_n)$
- **open query evaluation:**
  - set of true tuples $(c_1, \ldots, c_n)$ (seen as substitutions)/
    ground sentences $\varphi(c_1, \ldots, c_n)$
  - pertinent completeness sentence (closed-world assumption)

Issue: Logical foundation of the relational model

- **Observation:**
  - *classical model*:
    any interpretation over any nonempty domain
  - *Herbrand model*:
    any interpretation (set of ground facts) over *syntactic material*
  - *finite model*:
    any interpretation over any *finite* nonempty domain
  - *DB-model*: any *finite* interpretation over
    fixed (possibly typed) *infinite* domain of *constants*
    with unique name axioms

- **Challenge:** reconsider the *theory of relational databases*
  in terms of first-order logic with *DB-semantics*

Result: Effectiveness for query sequences

- **Framework:**
  - relational
  - *DB-semantics* of first-order logic
  - restricted to *Bernays-Schoenfinkel class*:
    – in prenex normal form having an $\forall^*\exists^*$ prefix
    – with decidable universal validity problem
- **Interaction:** sequence of query answering, closed as well as open ones
- **Control:**
  - basic approaches of *refusal*, *lying*, or the *combination*
  - control of sufficiently many closed sentences obtained by a substitution in a *fixed* sequence
  - control of suitably formed *completeness sentences*
- **Claim:** *terminating* and *confidentiality preserving*

technische universität dortmund

Issue: Entailment problems with completeness sentences

- **Observation:**
  standard theorem prover leads to efficiency degradation,
  even under
  - rewriting of completeness sentences,
    exploiting the active domain
  - minimizing the number of prover calls,
    by a divide-and-conquer heuristic

- **Challenge:**
  explore efficient computational approaches to decide
  entailment problems of first-order logic under DB-semantics
  when relational *completeness sentences* are involved

Static View Publishing

Confidentiality-Preserving View Publishing

broad range of well-established frameworks:

- distortions of *statistical databases*
- *value generalization* and *row-suppressing* for achieving *k-anonymity* and *l-diversity* of tables
- *database fragmentation and encryption* for cloud computing
- ...
- confidentiality-preserving view publishing by CIE

Result: Effectiveness of intensional iterative view generation

- **Framework:**
  - abstract, relational, description logics, respectively
  - suitable restrictions
    to ensure computability and to guarantee termination
- **Interaction:** view publishing
- **Control:**
  limit of approximations "from above"
  (starting with total ignorance)
  by exhaustive querying
- **Claim:** confidentiality preserving

Result: Effectiveness of extensional iterative view generation

- **Framework:**
  - propositional, relational, XML, respectively
  - suitable restrictions
    to ensure computability and to guarantee termination
- **Interaction:** view publishing
- **Control:**
  limit of approximations "from below"
  (starting with actual instance)
  by removing constraint violations
- **Claim:** confidentiality preserving

Result: Effectiveness of extensional global view generation

- **Framework:**
  - relational
  - dedicated cases with suitable restrictions
- **Interaction:** view publishing
- **Control:**
  result of individually substituting violating items by applying weakening options that are instance-independently "admissible" and non-interferential
- **Claim:** confidentiality preserving

Issue: Comparison of generalized view generation strategies

- **Observation:**
  - only specific examples for general strategies
  - only sufficient conditions for computability and termination
  - different notions of "optimality"
- **Challenge:**
  - generalize and elaborate the view generation strategies
  - systematically compare their achievements
  - in particular, evaluate availability of information

Advanced Reasoning

Incomplete Propositional Information System

- so far, information system represents *complete* knowledge:
  - instance as set of atoms
    – a "complete" truth evaluation according to "real world" –
  - model-theoretic semantics
  - query evaluation by truth evaluation
- often, there is only *incomplete* knowledge available:
  - instance as any consistent set of any kind of sentences
    – seen as being true in "real world" –
  - proof-theoretic semantics
  - query evaluation by entailment (also denoted by $\models$)

$$eval(db, \varphi) := \begin{cases} true & \text{if } db \models \varphi \\ false & \text{if } db \models \neg\varphi \\ undefined & \text{otherwise} \end{cases}$$

Employing Propositional Modal Logic of Knowledge

- *knowledge operator* K to speak about "the information system knows that ..."
- resulting query evaluation

$$eval(db, \varphi) := \begin{cases} K\varphi & \text{if } db \models \varphi \\ K\neg\varphi & \text{if } db \models \neg\varphi \\ \neg K\varphi \wedge \neg K\neg\varphi & \text{otherwise} \end{cases}$$

- additional flexibility for distorting answers, exploited by *distortion tables*

technische universität dortmund

Result: Effectiveness of adapted basic censors for query sequences to an incomplete information system

- **Framework:** propositional, incomplete
- **Interaction:** sequence of query answering
- **Control:**
  - adapted censors for *refusal*, *lying*, or the *combination*
  - based on modal logic of knowledge
  - employing a finite distortion table
- **Claim:** confidentiality preserving

Issue: First-order modal logic for censor construction

- **Observation:**
  extending classical first-order logic by modalities
  with Kripke-semantics dealing with "worlds"
  requires highly sophisticated considerations, e.g.:
  – semantics of constants?
  – semantics of nested occurrences of
  a modality and a quantor?
- **Challenge:**
  elaborate the *modal logic approach* to construct censors
  for an *incomplete* information system
  based on *first-order logic*

technische universität dortmund

Result: Effectiveness of adapted basic censors for query sequences to an incomplete information system

- **Framework:** propositionalized first-order, incomplete
- **Interaction:** sequence of query answering
- **Control:**
  - adapted censors for *refusal*, *lying*, or the *combination*
  - based on modal logic of knowledge
  - employing a finite distortion table
- **Claim:** confidentiality preserving

Result: Effectiveness of adapted basic censors for view publishing for an incomplete information system

- **Framework:**
  description logics as tractable fragment of first-order, incomplete
- **Interaction:** view publishing
- **Control:**
  - variants of censors for *refusal* and *lying*
  - limit of the controlled answers
    of any exhaustive sequence of atoms
- **Claim:** confidentiality preserving

technische universität dortmund

Result: Effectiveness of a refusal censor for sequences of belief queries and belief revisions

- **Framework:**
  - non-monotonic propositional for *belief*
  - extended syntax for conditionals (default rules, ...)
  - originally based on *ordinal conditional functions*
  - later generalized for a *class of consequence relations* with "allowed" axiomatization
  - *skeptically* reasoning client
- **Interaction:** mixed sequences of query answering and revision processing
- **Control:** computational adaption of the basic censor for *refusal*
- **Claim:** confidentiality preserving

Issue: Censor constructions for non-monotonic frameworks

- **Observation:**
  - so far, only specific kind of belief reasoning
  - so far, only specific kind of distortion
  - so far, only specific kind of the client's reasoning
- **Challenge:**
  - for further examples of a non-monotonic framework, explore the options to construct a censor
  - concisely generalize such constructions

Advanced Interactions see next presentation by Cornelia Tadros

Conclusion

technische universität dortmund

A Retrospective Guideline

- **traditional successful research:**
  reasoning about *own* knowledge/belief
- **an extended topic:**
  reasoning about about another one's *internal* knowledge/belief based on *observable* communication data
- **the additional security challenge:**
  minimally distorting communication data to confine the receiver's reasoning – as assumed and to be simulated – about the sender's *internal* knowledge/belief

Main Dimensions for Results and Issues: How to Compose?

Framework	Interaction (Sequence of)	Censor	Confidentiality Claim
abstract	query answering	refusal	possibilistic
propositional, complete	view publishing	weakening	probabilistic
propositional, incomplete	update processing	value generalization	"evaluated"
propositional, incomplete belief	revision processing	lying	...
...	program execution	combined refusal and lying	approximated
relational first-order, complete	...	...	...
relational first-order, incomplete	mix of ...		
...	...		
description logics, incomplete			
...			

for *framework* and *interaction*:
– construct *censor(s)*
– prove *claims*