integrity constraints for general-purpose knowledge bases

luís cruz-filipe[1]

(with isabel nunes[2] & peter schneider-kamp[1])

[1]department of mathematics and computer science
university of southern denmark

[2]department of informatics
faculty of sciences, university of lisbon

foiks 2016
march 8th, 2016

outline

our research context

- ⇝ theory vs practice in knowledge representation
- real-world applications combine different expert systems
- systems use different formalisms/paradigms
- quite often, *ad-hoc* combinations for specific purposes

our research context

- theory vs practice in knowledge representation
  - real-world applications combine different expert systems
  - systems use different formalisms/paradigms
  - quite often, *ad-hoc* combinations for specific purposes

- integration poses new challenges
  - different systems represent the same knowledge in different ways
  - often there is redundancy in the *global* system

our research context

- ⇝ theory vs practice in knowledge representation
  - real-world applications combine different expert systems
  - systems use different formalisms/paradigms
  - quite often, *ad-hoc* combinations for specific purposes
- ⇝ integration poses new challenges
  - different systems represent the same knowledge in different ways
  - often there is redundancy in the *global* system
- ⇝ integrity constraints?

integrity constraints

the goal what should the properties of a good formalism be?

integrity constraints

the goal what should the properties of a good formalism be?

- generalize existing notions in particular frameworks (e.g. relational databases)
- expressive enough to capture conditions spanning several systems
- decidability? good complexity bounds?
- algorithms for repairing inconsistencies

integrity constraints

the goal what should the properties of a good formalism be?

- generalize existing notions in particular frameworks (e.g. relational databases)
- expressive enough to capture conditions spanning several systems
- decidability? good complexity bounds?
- algorithms for repairing inconsistencies

our target active integrity constraints (flesca *et al.*, '04)

- defined for relational databases
- allow to express both *constraints* and *repair actions*
- "good" algorithms for repair

integrity constraints

the goal what should the properties of a good formalism be?

- generalize existing notions in particular frameworks (e.g. relational databases)
- expressive enough to capture conditions spanning several systems
- decidability? good complexity bounds?
- algorithms for repairing inconsistencies

our target active integrity constraints (flesca *et al.*, '04)

- defined for relational databases
- allow to express both *constraints* and *repair actions*
- "good" algorithms for repair
- . . . and we like them

what's happening around us

people have worried about this in. . .

relational dbs the classical setting

what's happening around us

people have worried about this in. . .

relational dbs the classical setting

deductive dbs mostly mid-1980s
$\rightsquigarrow$ separate integrity constraints from data
$\rightsquigarrow$ integrity constraints as preferred models

what's happening around us

people have worried about this in. . .

relational dbs the classical setting

deductive dbs mostly mid-1980s
$\rightsquigarrow$ separate integrity constraints from data
$\rightsquigarrow$ integrity constraints as preferred models

ontologies last 15–20 years
$\rightsquigarrow$ open-world semantics makes the problem different
$\rightsquigarrow$ integrity constraints as terminological axioms
(but with a different semantics)

what's happening around us

people have worried about this in. . .

relational dbs — the classical setting

deductive dbs — mostly mid-1980s

- ⇝ separate integrity constraints from data
- ⇝ integrity constraints as preferred models

ontologies — last 15–20 years

- ⇝ open-world semantics makes the problem different
- ⇝ integrity constraints as terminological axioms (but with a different semantics)

heterogeneous systems — last 10 years

- ⇝ in multi-context systems (our setting)
- ⇝ internalize integrity constraints

what's happening around us

people have worried about this in. . .

relational dbs	the classical setting
deductive dbs	mostly mid-1980s ⇝ separate integrity constraints from data ⇝ integrity constraints as preferred models
ontologies	last 15–20 years ⇝ open-world semantics makes the problem different ⇝ integrity constraints as terminological axioms (but with a different semantics)
heterogeneous systems	last 10 years ⇝ in multi-context systems (our setting) ⇝ internalize integrity constraints
in general	no continuation, no apparent consensus

our contribution

- integrity constraints in a general-purpose framework
- captures previous constructions as special cases
- clean separation between consistency and integrity
- express preferences between different models
- easily extended to include a notion of repair

outline

multi-context systems

main idea

- reasoning systems ("contexts")
- connected by datalog-style rules ("bridge rules")

multi-context systems

main idea

- reasoning systems ("contexts")
- connected by datalog-style rules ("bridge rules")

brewka & eiter '07

heterogenous non-monotonic multi-context systems

heterogeneous contexts can use different logics

non-monotonic bridge rules can contain negation

multi-context several different systems

multi-context systems

main idea

- reasoning systems ("contexts")
- connected by datalog-style rules ("bridge rules")

brewka & eiter '07 heterogenous non-monotonic multi-context systems

heterogeneous contexts can use different logics

non-monotonic bridge rules can contain negation

multi-context several different systems

more precisely *relational* multi-context systems allow us to use first-order variables in bridge rules

the contexts

logic abstractly, a logic is characterized by:

- the set of its well-formed *knowledge bases* (syntax)
- the set of its possible *belief sets* (models)
- a function assigning to each knowledge base a set of acceptable belief sets (semantics)

plus some technicalities regarding the use of variables (see paper)

the contexts

logic abstractly, a logic is characterized by:

- the set of its well-formed *knowledge bases* (syntax)
- the set of its possible *belief sets* (models)
- a function assigning to each knowledge base a set of acceptable belief sets (semantics)

example first-order logic over a signature Σ

- knowledge bases are sets of well-formed formulas over Σ
- belief sets are pairs of first-order interpretations and assignments over Σ
- the acceptable belief sets wrt a knowledge base are the models of that knowledge base

the contexts

logic abstractly, a logic is characterized by:

- the set of its well-formed *knowledge bases* (syntax)
- the set of its possible *belief sets* (models)
- a function assigning to each knowledge base a set of acceptable belief sets (semantics)

context a context is defined as a tuple containing:

- its underlying logic
- a particular knowledge base
- a set of bridge rules (coming up)

plus some technicalities regarding the use of variables (see paper)

bridge rules

definition datalog-style rules to exchange information

$$(k : s) \leftarrow \bigwedge_{i=1}^{q} (c_i : p_i), \bigwedge_{j=q+1}^{m} \text{not } (c_j : p_j)$$

where k, c_i, c_j are context identifiers and s, p_i, p_j instantiate to elements of knowledge bases

bridge rules

definition datalog-style rules to exchange information

$$(k : s) \leftarrow \bigwedge_{i=1}^{q} (c_i : p_i), \bigwedge_{j=q+1}^{m} \text{not } (c_j : p_j)$$

where k, c_i, c_j are context identifiers and s, p_i, p_j instantiate to elements of knowledge bases

semantics usual interpretation: if each p_i holds in context c_i, and no p_j holds in context c_j, then add s to the knowledge base in context k

$\rightsquigarrow$ usual safeness assumption

equilibria

belief state a belief state is a collection of belief sets (one for each context)

equilibria

belief state a belief state is a collection of belief sets (one for each context)

equilibrium an equilibrium is a belief set that is compatible with all knowledge bases and bridge rules:
each S_i is an acceptable belief state wrt $(kb_i \cup \mathsf{app}_i(S))$, where $\mathsf{app}_i(S)$ collects all heads of bridge rules in c_i whose bodies hold in S

equilibria

belief state a belief state is a collection of belief sets (one for each context)

equilibrium an equilibrium is a belief set that is compatible with all knowledge bases and bridge rules:
each S_i is an acceptable belief state wrt $(kb_i \cup \mathsf{app}_i(S))$, where $\mathsf{app}_i(S)$ collects all heads of bridge rules in c_i whose bodies hold in S

$\leadsto$ think logic programming...

equilibria

belief state a belief state is a collection of belief sets (one for each context)

equilibrium an equilibrium is a belief set that is compatible with all knowledge bases and bridge rules:
each S_i is an acceptable belief state wrt $(kb_i \cup \mathrm{app}_i(S))$,
where $\mathrm{app}_i(S)$ collects all heads of bridge rules in c_i whose bodies hold in S

$\rightsquigarrow$ think logic programming...

consistency an mcs is *inconsistent* if it does not have equilibria

$\rightsquigarrow$ several different types of equilibria exist (as in logic programming)

integrity constraints

syntax an integrity constraint is written as a bridge rule with empty head

semantics an mcs M **strongly**/*weakly* satisfies an ic r if:

- M is consistent
- **all**/*some* of M's equilibria make the body of r false

integrity constraints

syntax an integrity constraint is written as a bridge rule with empty head

semantics an mcs M **strongly**/*weakly* satisfies an ic r if:

- M is consistent
- **all**/*some* of M's equilibria make the body of r false

$\rightsquigarrow$ consistency is required to get **strong** $\Rightarrow$ *weak*

integrity constraints

syntax an integrity constraint is written as a bridge rule with empty head

semantics an mcs M **strongly**/*weakly* satisfies an ic r if:

- M is consistent
- **all**/*some* of M's equilibria make the body of r false

⇝ consistency is required to get **strong** ⇒ *weak*

internalization we can reduce satisfaction of integrity constraints to logical (in)consistency by adding an *inconsistency context* (unitary logic)

⇝ good for theoretical results, bad for intuition

properties

- weak satisfaction of integrity constraints by an mcs reduces to logical consistency of an mcs
- strong satisfaction of integrity constraints by an mcs reduces to logical inconsistency of an mcs

properties

- weak satisfaction of integrity constraints by an mcs reduces to logical consistency of an mcs
- strong satisfaction of integrity constraints by an mcs reduces to logical inconsistency of an mcs
- decidability of satisfaction is equivalent to decidability of consistency
- weak satisfaction is usually as hard as context consistency
- if context complexity is in class Σ_i^P, then strong satisfaction is in class Δ_{i+1}^P

outline

relational databases

databases as mcs

we can view a relational database *DB* as an mcs whose only context has:

- sets of ground first-order formulas as knowledge bases
- sets of ground literals as belief sets
- the natural closed-world semantics
- *DB* as the knowledge base
- no bridge rules

relational databases

databases as mcs

we can view a relational database DB as an mcs whose only context has:

- sets of ground first-order formulas as knowledge bases
- sets of ground literals as belief sets
- the natural closed-world semantics
- DB as the knowledge base
- no bridge rules

$\leadsto$ integrity constraints (in denial form) over DB can be written directly in our syntax

relational databases

databases as mcs we can view a relational database *DB* as an mcs whose only context has:

- sets of ground first-order formulas as knowledge bases
- sets of ground literals as belief sets
- the natural closed-world semantics
- *DB* as the knowledge base
- no bridge rules

$\leadsto$ integrity constraints (in denial form) over *DB* can be written directly in our syntax

correspondence theorem *DB* satisfies a set η of integrity constraints iff the mcs induced by *DB* strongly/weakly satisfies the integrity constraints induced by η

distributed databases

- distributed databases store information in different nodes
- often there is duplication of data for efficiency/resilience
- integrity constraints can specify inter-node consistency

distributed databases

- distributed databases store information in different nodes
- often there is duplication of data for efficiency/resilience
- integrity constraints can specify inter-node consistency

formally

- each node (database) is a context as before
- there are no bridge rules
- suppose unary predicate p exists in contexts 1 and 2

$$\leftarrow (1 : p(X)), \text{not } (2 : p(X))$$
$$\leftarrow (2 : p(X)), \text{not } (1 : p(X))$$

specify that both contexts agree on p

distributed databases

- distributed databases store information in different nodes
- often there is duplication of data for efficiency/resilience
- integrity constraints can specify inter-node consistency

formally

- each node (database) is a context as before
- there are no bridge rules
- suppose unary predicate p exists in contexts 1 and 2

$$\leftarrow (1 : p(X)), \text{not } (2 : p(X))$$
$$\leftarrow (2 : p(X)), \text{not } (1 : p(X))$$

specify that both contexts agree on p

$\rightsquigarrow$ different consistency strategies can be expressed as repair preferences...

deductive databases

idea centralized database with several distinct views

- "extensional" database does not contain rules, only data
- "intensional" views do not contain data, only rules
- only the database can be changed

deductive databases

idea centralized database with several distinct views

- "extensional" database does not contain rules, only data
- "intensional" views do not contain data, only rules
- only the database can be changed

as an mcs one context for the database, one for each view

- central database is one context (as before)
- each view is a context with empty knowledge base
- all inference is encoded as bridge rules in each view

$\rightsquigarrow$ also allows for views depending on views

deductive databases

idea centralized database with several distinct views

- "extensional" database does not contain rules, only data
- "intensional" views do not contain data, only rules
- only the database can be changed

as an mcs one context for the database, one for each view

- central database is one context (as before)
- each view is a context with empty knowledge base
- all inference is encoded as bridge rules in each view

advantages integrity constraints can...

- ...talk about derived predicates
- ...express consistency among views

compare repairing inconsistencies vs view-update problem

the open world case (i/ii)

ontologies widely used in practice in knowledge representation

- thought of as universal intermediate language
- typically use description logics
- incomplete view of the world (open-world semantics)

the open world case (i/ii)

ontologies widely used in practice in knowledge representation

- thought of as universal intermediate language
- typically use description logics
- incomplete view of the world (open-world semantics)

description logic essentially variants of first-order logic, with special syntax

- predicates are unary (*concepts*) or binary (*roles*)
- separate *terminology* (relationships) and *axioms* (instances)
- typically include higher-order constructors, i.e. transitive closure
- enjoy decidability and low complexity of reasoning

the open-world case (ii/ii)

extensions

- combination with rule-based reasoning
- controlled addition of closed-world semantics
- definition of integrity constraints (no follow-up)

the open-world case (ii/ii)

extensions

- combination with rule-based reasoning
- controlled addition of closed-world semantics
- definition of integrity constraints (no follow-up)

as an mcs an ontology can become a context

- knowledge bases are sets of well-formed formulas
- belief sets are *closed* sets of well-formed formulas
- acceptable beliefs are derivable consequences of the knowledge base

integrity constraints in ontologies

motik et al. '11 integrity constraints over a single ontology

- terminological formulas, kept separate from the ontology
- seen as constraints on the knowledge state
- not intended to infer new knowledge

integrity constraints in ontologies

motik et al. '11 integrity constraints over a single ontology

- terminological formulas, kept separate from the ontology
- seen as constraints on the knowledge state
- not intended to infer new knowledge

fang et al. '11 integrity constraints among multiple ontologies

- datalog-style rules, kept separate from the ontologies
- specify consistency among different data sources
- not intended to infer new knowledge

integrity constraints in ontologies

motik et al. '11 integrity constraints over a single ontology

- terminological formulas, kept separate from the ontology
- seen as constraints on the knowledge state
- not intended to infer new knowledge

our proposal partially captures the first scenario

- can use standard (partial) translations from description logic to logic programming
- can add a new context for faithfully expressing integrity constraints
- but cannot talk about unnamed individuals

integrity constraints in ontologies

fang et al. '11 integrity constraints among multiple ontologies

- datalog-style rules, kept separate from the ontologies
- specify consistency among different data sources
- not intended to infer new knowledge

our proposal faithfully captures the second scenario

- our rules are similar to datalog syntax
- mcs make it easy to talk about several ontologies
- the proposal is also unable to talk about unnamed individuals

integrity constraints in ontologies

fang et al. '11 integrity constraints among multiple ontologies

- datalog-style rules, kept separate from the ontologies
- specify consistency among different data sources
- not intended to infer new knowledge

our proposal faithfully captures the second scenario

- our rules are similar to datalog syntax
- mcs make it easy to talk about several ontologies
- the proposal is also unable to talk about unnamed individuals

$\rightsquigarrow$ it is not clear why we want integrity constraints over unnamed individuals. . .

outline

repairs

goal given an mcs that does not satisfy its integrity constraints, how can we repair it?

$\rightsquigarrow$ much more complex than in relational databases

repairs

goal given an mcs that does not satisfy its integrity constraints, how can we repair it?

⇝ much more complex than in relational databases

problem what actions are we allowed to perform?

- we should be able to change the knowledge base...
- ...but not all changes make sense

repairs

goal given an mcs that does not satisfy its integrity constraints, how can we repair it?

$\rightsquigarrow$ much more complex than in relational databases

problem what actions are we allowed to perform?

- we should be able to change the knowledge base...
- ...but not all changes make sense

deductive databases although integrity constraints can talk about tables in the views, changes have to be made to the extensional database

managed multi-context systems

intuition each context contains a *management function*, describing how the knowledge base can be changed

$\rightsquigarrow$ heads of bridge rules now contain *update actions* – calls to the management function

managed multi-context systems

intuition each context contains a *management function*, describing how the knowledge base can be changed

$\leadsto$ heads of bridge rules now contain *update actions* – calls to the management function

repairs we can define a repair of an inconsistent managed mcs as a set of update actions that, when applied, will turn it into an mcs satisfying all integrity constraints

managed multi-context systems

intuition each context contains a *management function*, describing how the knowledge base can be changed

$\rightsquigarrow$ heads of bridge rules now contain *update actions* – calls to the management function

repairs we can define a repair of an inconsistent managed mcs as a set of update actions that, when applied, will turn it into an mcs satisfying all integrity constraints

- generalizes the database case
- we can capture typical additional restrictions on repairs
- suitable for our future plans

active integrity constraints (sketch)

idea rules with heads specifying update actions that can repair the inconsistency

$\rightsquigarrow$ in this context: disjunctive bridge rules (syntactically)

active integrity constraints (sketch)

idea rules with heads specifying update actions that can repair the inconsistency

$\rightsquigarrow$ in this context: disjunctive bridge rules (syntactically)

problems how can we guarantee that the actions specified solve the problem?

- may depend on the actual knowledge base
- undecidable problem in general

outline

our contribution

- integrity constraints in a general-purpose framework
- captures previous constructions as special cases
- clean separation between consistency and integrity
- express preferences between different models
- easily extended to include a notion of repair

thank you!