Knowledge Base Repair: From Active Integrity Constraints to Active TBoxes

Andreas Herzig

(joint work with Guillaume Feuillade and Christos Rantsoudis)

DL&NMR workshops, Rhodes & AoE

12 September 2020

Outline

Databases and integrity constraints

Integrity constraints (IC): fundamental part of a database schema

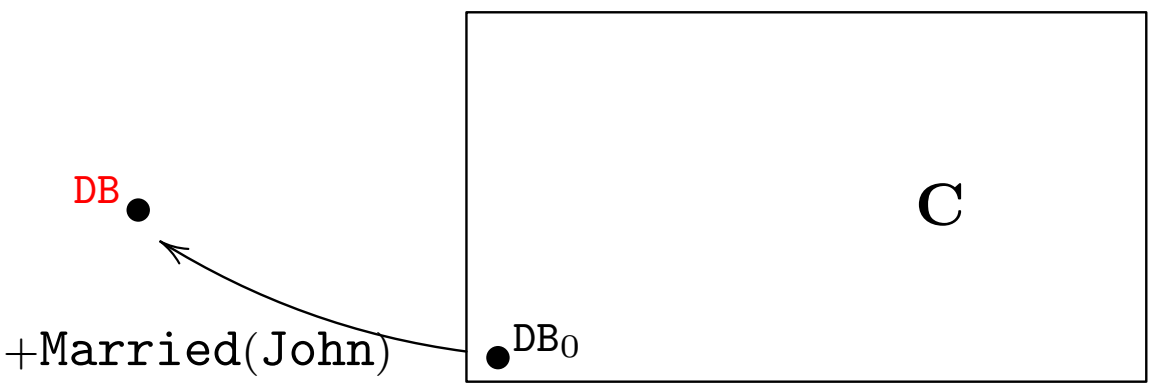
- ideally: $\mathtt{DB} \models \mathbf{C}$
- in practice: $\mathtt{DB} \not\models \mathbf{C}$ often happens

$$\mathbf{C} = (\forall x)[\mathtt{Bachelor}(x) \wedge \mathtt{Married}(x) \rightarrow \bot]$$

$$\mathtt{DB}_0 = \{\mathtt{Bachelor}(\mathtt{John})\}$$

$$\Downarrow \; +\mathtt{Married}(\mathtt{John})$$

$$\mathtt{DB} = \{\mathtt{Bachelor}(\mathtt{John}), \mathtt{Married}(\mathtt{John})\}$$



Repairs

Two solutions

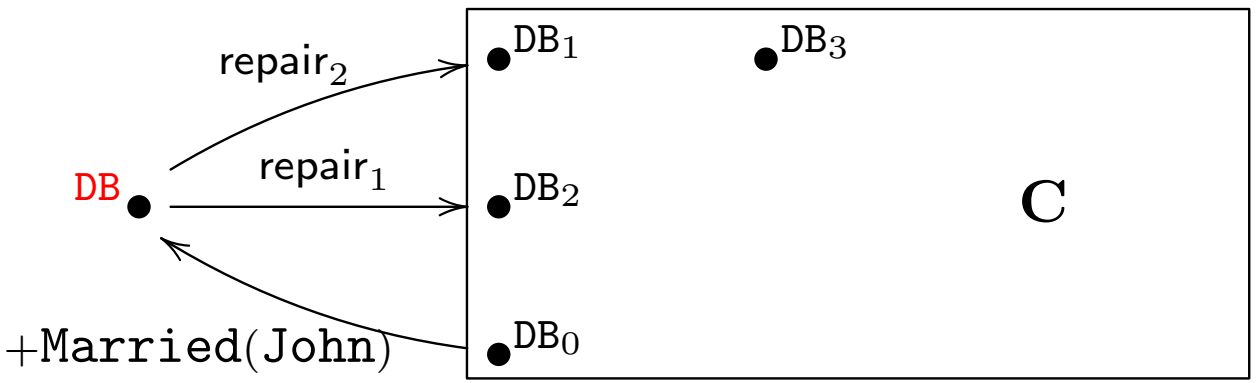
1. make DB consistent again
   - repair ('data cleaning')
2. live with inconsistent DB
   - hypothetical repair: 'consistent query answering'
   - consistent answer holds in all possible repairs

possible repairs of $\mathtt{DB} = \{\mathtt{Bachelor(John)}, \mathtt{Married(John)}\}$:

$$\mathrm{DB}_1 = \{\mathtt{Bachelor(John)}\}$$
$$\mathrm{DB}_2 = \{\mathtt{Married(John)}\}$$
$$\mathrm{DB}_3 = \{\mathtt{Bachelor(John)}, \mathtt{Bachelor}(Jim)\}$$

- all do the job: $\therefore \mathrm{DB}_i \models \mathbf{C}$
- . . . but there are too many
- . . . and some are not intended

Repairs: minimal change



Minimal repair = a DB′ *closest* to DB such that DB′ $\models$ C

- intuitively $\mathrm{DB}_3 = \{\mathtt{Bachelor(John)}, \mathtt{Bachelor}(Jim)\}$ less close to DB than DB_1 and DB_2
- which definition of closeness?
  - symmetric difference
  - cf. Possible Models Approach PMA [Winslett, AAAI 1988]
  - PMA repairs; bring about PMA updates

Repairs: minimal change is not enough

- number of PMA repairs can still be huge
- typically some of them are unintended
- PMA repairs of $\mathtt{DB} = \{\mathtt{Bachelor(John)}, \mathtt{Married(John)}\}$:

$$\mathtt{DB}_1 = \{\mathtt{Bachelor(John)}\}$$
$$\mathtt{DB}_2 = \{\mathtt{Married(John)}\}$$

  - both are set inclusion minimal
  - $\mathtt{DB}_1$ unintended
- can we do better by making $\mathbf{C}$ more informative?

More informed integrity constraints

Active Integrity Constraints (AIC) [Flesca et al., PPDP 2004]

$$\boxed{\text{active IC} = \text{static IC} + \text{update actions}}$$

Static : $(\forall x)[\texttt{Bachelor}(x) \wedge \texttt{Married}(x) \rightarrow \bot]$

Active : $(\forall x)[\texttt{Bachelor}(x) \wedge \texttt{Married}(x) \rightarrow \bot, \{-\texttt{Bachelor}(x)\}]$

Intuitions

- *"an extension of integrity constraints that allows to specify for each constraint the actions to be performed to satisfy it"* [Flesca et al., PPDP 2004]
- *"an AIC encodes explicitly both an integrity constraint and preferred basic actions to repair it, if it is violated"* [Caroprese&Truszczynski, TPLP 2011]

Active Integrity Constraints: predecessors

History

1. active databases [Ceri&Widow, 1994]
   - More general ECA (event-condition-action) rules:

     "if *event* occurs and *condition* holds then do *action*"
   - procedural
   - repair may not terminate
2. database repair constraints [Greco et al., TKDE 2003]
   - subsumed [Caroprese et al., 2006]
3. revision programs [Marek&Truszczyński, TCS 1998]
   - logic programs
   - semantics differ, but are close [Caroprese&Truszczyński, TPLP 2011]

Aims of talk

1. Revisit intuitions and semantics
2. Dynamic Logic analysis of AICs
   - similar in spirit: Kiringa's logical account of ECA rules in the Sitcalc [Kiringa, LICS 2001; Kiringa&Reiter 2003]
3. Transfer to description logics
   - define active TBoxes
   - extend ALC by dynamic operators

Outline

Basic notations

- propositional logic
  - hypothesis: everything is grounded
  - propositional variables $\mathbb{P} = \{p, q \ldots\}$
- databases are sets of propositional variables $\mathtt{DB} \subseteq \mathbb{P}$
- static integrity constraints are sets of literals (clauses) $\mathbf{C}$
- update actions are assignments $\alpha = +p$ and $\alpha = -p$
  - *consistent* set of update actions $\mathtt{U}$: not ($+p \in \mathtt{U}$ and $-p \in \mathtt{U}$)
- update of $\mathtt{DB}$ by $\mathtt{U}$:

$$\mathtt{DB} \circ \mathtt{U} = \big(\mathtt{DB} \setminus \{p \,:\, -p \in \mathtt{U}\}\big) \cup \{p \,:\, +p \in \mathtt{U}\}$$

  - if $\mathtt{U}$ is consistent: order of application irrelevant

Active Integrity Constraints: syntax

$$\boxed{r = \langle \mathbf{C}(r), \mathbf{R}(r) \rangle}$$

- $\mathbf{C}(r)$ is a clause (a set of literals)
- $\mathbf{R}(r)$ is a set of update actions making some literals of $\mathbf{C}(r)$ true:

$$\mathbf{R}(r) \subseteq \{+p : p \in \mathbf{C}(r)\} \cup \{-p : \neg p \in \mathbf{C}(r)\}$$

$$\langle \neg \texttt{Bachelor} \vee \neg \texttt{Married}, \{-\texttt{Bachelor}\} \rangle$$

- database DB + finite sets of AICs $\eta = \{r_1, \ldots, r_n\}$

Active Integrity Constraints: which semantics?

Various semantics

- repairs *tout court*, alias PMA repairs (v.s.)
- founded repairs [Caroprese et al., ICLP 2006]
- justified repairs [Caroprese&Truszczynski, TPLP 2011]
- well-founded repairs [Cruz Felipe et al., TASE 2013]
- dynamic repairs [Feuillade&Herzig, JELIA 2013]
- grounded repairs [Bogaerts&Cruz Felipe, AIJ 2018]

. . . and each in several versions

- drop minimality requirement $\Longrightarrow$ weak versions
  - for PMA repairs: makes updates drastic
- minimise exceptions
  - preferred update actions are soft constraints, can be violated
  - if static part of η consistent then repair exists

Active Integrity Constraints: different intuitions

Permission vs. obligation

when $\mathbf{C}(r)$ is violated:

1. permission that the repair contains some $\alpha \in \mathbf{R}(r)$
   ... but $\mathbf{C}(r)$ might as well be repaired by other AICs

   "If $\mathtt{DB} \not\models \mathbf{C}(r)$, *then* DB *is inconsistent. It is allowed to repair this inconsistency by executing one or more of the* $\alpha_i \in \mathbf{R}(r)$*."* [Bogaerts&Cruz Felipe, AIJ 2018; notation adapted]

2. obligation that the repair contains some $\alpha \in \mathbf{R}(r)$

   "If $r \in \eta$ *and, for every non-updatable literal in* $\ell \in \mathbf{C}(\eta)$ *there is an update action* $-\ell \in \mathtt{U}$ *then [...] the result of the update* $\mathtt{DB} \circ \mathtt{U}$ *satisfies all nonupdatable literals in* $\mathbf{C}(r)$*. To guarantee that* $\mathtt{DB} \circ \mathtt{U}$ *satisfies* r, $\mathtt{DB} \circ \mathtt{U}$ *must satisfy at least one literal in* $\mathbf{C}(r)$*. To this end* U *must contain at least one update action from* $\mathbf{R}(r)$*."* [Caroprese&Truszczynski, TPLP 2011; notation adapted]

Active Integrity Constraints: different intuitions, ctd.

Permission vs. obligation: consequences

when $\mathbf{C}(r)$ is violated ...

1. 'permission' reading:

$$\langle p \vee q, \{+p, +q\}\rangle \text{ equivalent to } \begin{cases} \langle p \vee q, \{+p\}\rangle \\ \langle p \vee q, \{+q\}\rangle \end{cases}$$

   $\Longrightarrow$ all $\mathbf{R}(r)$ singletons ("η normalised")

2. 'obligation' reading:
   - η cannot be normalised
   - computation is a priori more local than 'permission' reading: ' if $\mathbf{C}(r)$ violated then repair via $\mathbf{R}(r)$ regardless of other AICs'

... but what does "$\mathbf{C}(r)$ is violated" mean? Just "DB $\not\models \mathbf{C}(r)$"?

Active Integrity Constraints: sharpening intuitions by means of abstract examples (1)

Example: one violation, no interaction

$$\mathtt{DB} = \emptyset \text{ and } \eta = \{\langle p \vee q, \{+p, +q\}\rangle, \langle q \vee \neg r, \{+q\}\rangle\}$$

- repairs are $\mathtt{U}_1 = \{+p\}$ and $\mathtt{U}_2 = \{+q\}$
- if $\mathtt{DB} \not\models \mathbf{C}(r)$ and for all other r', $\mathtt{DB} \models \mathbf{C}(r')$ and $\mathbf{C}(r')$ does not *interact* with $\mathbf{C}(r)$, (resolution rule doesn't apply)
  then the repairs are just the update actions in $\mathbf{R}(r)$:

$$\mathtt{U}_i = \{\alpha_i\} \text{ for } \alpha_i \in \mathbf{R}(r)$$

Active Integrity Constraints: sharpening intuitions by means of abstract examples (2)

Example: one violation, with interaction

$$\mathtt{DB} = \emptyset \text{ and } \eta = \big\{\langle p, \{+p\}\rangle, \langle \neg p \vee q \vee r, \{+q\}\rangle\big\}$$

- repair is $\mathtt{U} = \{+p, +q\}$ (and not: $\{+p, +r\}$)
- $\neg p \vee q \vee r$ is not violated by DB, but after update by $+p$!
- hence $\mathtt{DB} \not\models \mathbf{C}(r)$ not enough a criterion
- in general: membership in U may have to be hypothesised
  - problem: circularity of support (v.i.)

up to now: no difference between permitted and obligatory reading

Active Integrity Constraints: sharpening intuitions by means of abstract examples (3)

Example: two violations, no interaction

$$\mathtt{DB} = \emptyset \text{ and } \eta = \{\langle p \vee q, \{+p\}\rangle, \langle p \vee q, \{+q\}\rangle\}$$

- different readings lead to different intuitions!
  - 'permission': repairs are $\mathtt{U}_1 = \{+p\}$ and $\mathtt{U}_2 = \{+q\}$
  - 'obligation": repair is $\mathtt{U} = \{+p, +q\}$
    - $\mathtt{U}$ is not minimal $\Longrightarrow$ not a PMA repair!
    - active part of η badly designed?

Active Integrity Constraints: sharpening intuitions by means of abstract examples (4)

Example: two violations, no interaction

$$\mathtt{DB} = \emptyset \text{ and } \eta = \{\langle p, \{+p\}\rangle, \langle p \vee q, \{+q\}\rangle\}$$

- different readings lead to different intuitions!
  1. 'permission': repair is $\mathtt{U} = \{+p\}$ ($\{+p, +q\}$ not minimal)
  2. 'obligation": repair is $\mathtt{U}' = \{+p, +q\}$
     - not a PMA repair!
     - active part of η badly designed?

Active Integrity Constraints: sharpening intuitions by means of abstract examples (5)

Example: one violation, with interaction

$$\mathtt{DB} = \emptyset \text{ and } \eta = \big\{\langle p \vee q, \{+p\}\rangle, \langle p \vee \neg q, \{+p\}\rangle, \langle \neg p \vee q, \{+q\}\rangle\big\}$$

- intuitions differ
  1. "circularity of support" $\Longrightarrow$ should have no repair [Caroprese&Truszczynski, TPLP 2011]
  2. we: repair should be $\mathtt{U} = \{+p, +q\}$
     reason: extensionality principle applies to first two AICs
     - static part equivalent to p
     - dynamic parts identical

     $\Longrightarrow$ first two AICs equivalent to $\langle p, \{+p\}\rangle$
     $\Longrightarrow$ cf. previous example

Active Integrity Constraints: summary of intuitions

1. permission reading and obligation reading come with different intuitions
   - obligations more local $\Longrightarrow$ should lead to a simpler account
2. obligation reading often leads to non-minimal repairs
   - may indicate flawed choices of update actions for some AICs
3. a new principle: extensionality
   - more general identity criteria for sets of AICs?
   - other postulates? cf. belief revision&update literature

Outline

Well-founded Repairs [Bogaerts&Cruz Felipe, AIJ 2018]

Idea

- choose violated AIC $r \in \eta$ (i.e., $\mathbf{C}(r)$ not satisfied)
- update by one of the actions in the active part $\mathbf{R}(r)$
- iterate until no more violation

Definition

PMA repair $\mathtt{U}$ for $\mathtt{DB}$ w.r.t. η is well-founded if

$\mathtt{U} = \{\alpha_1, \ldots, \alpha_n\}$ (for some ordering) such that

for every α_i there is an AIC $r_i \in \eta$ with

- $\mathtt{DB} \circ \{\alpha_1, \ldots, \alpha_{i-1}\} \not\models \mathbf{C}(r_i)$ (r_i is violated)
- $\alpha_i \in \mathbf{R}(r_i)$

compatible with permission reading and with obligation reading

Founded Repairs [Caroprese et al., ICLP 2006]

Idea

- update actions $\alpha \in \mathtt{U}$ should be supported by some active constraint r
  - r would be violated without α

Definition

PMA repair $\mathtt{U}$ for $\mathtt{DB}$ w.r.t. η is founded if

for every $\alpha \in \mathtt{U}$ there is an AIC $r \in \eta$ such that

- $\alpha \in \mathbf{R}(r)$ and
- $\mathtt{DB} \circ (\mathtt{U} \setminus \{\alpha\}) \not\models \mathbf{C}(r)$

permission reading (definition checks that every $\alpha \in \mathtt{U}$ is permitted)

Founded Repairs: examples (ctd.)

Example [Cruz Felipe et al., 2013]

$$\mathtt{DB} = \emptyset \text{ and } \eta = \{\, \langle \neg p \vee q, \{+p\}\rangle, \langle p \vee \neg q, \{+q\}\rangle, \langle \neg p \vee r, \{+r\}\rangle, \langle \neg q \vee r, \{+r\}\rangle \,\}$$

- two founded repairs:
  - $\mathtt{U}_1 = \{+r\}$
  - $\mathtt{U}_2 = \{+p, +q\}$ ("circularity of support")
- reason: there could be $\mathtt{U}' \subset \mathtt{U} \setminus \{\alpha\}$ with $\mathtt{DB} \circ \mathtt{U}' \models \mathbf{C}(r)$!

Grounded Repairs [Bogaert&Cruz Felipe, AIJ 2018]

Idea

- generalises negative condition of foundedness

$$\mathtt{DB} \circ (\mathtt{U} \setminus \{\alpha\}) \not\models \mathbf{C}(r)$$

- hypothesis: all $\mathbf{R}(r)$ are singletons ('all AICs are normal')

Definition

PMA repair $\mathtt{U}$ of $\mathtt{DB}$ w.r.t. η is grounded if

for every $\mathtt{U}' \subset \mathtt{U}$ there is a $r \in \eta$ such that

- $\mathbf{R}(r) \cap (\mathtt{U} \setminus \mathtt{U}') \neq \emptyset$ and
- $\mathtt{DB} \circ \mathtt{U}' \not\models \mathbf{C}(r)$

Properties

- all grounded repairs are well-founded, founded, minimal

Justified repairs [Caroprese&Truszczynski, TPLP 2011]

Definitions

- *non-effect actions* w.r.t. DB and U:

$$\mathtt{neff}_{\mathtt{DB}}(\mathtt{U}) = \{\alpha \,:\, \mathtt{DB} \circ \alpha = \mathtt{DB} \text{ and } (\mathtt{DB} \circ \mathtt{U}) \circ \alpha = \mathtt{DB} \circ \mathtt{U}\}$$

- *non-updatable literals* of r:

$$\begin{aligned}\mathtt{nup}(r) = &\{p \in \mathbf{C}(r) \,:\, +p \notin \mathbf{R}(r)\} \cup \\ &\{\neg p \in \mathbf{C}(r) \,:\, -p \notin \mathbf{R}(r)\}\end{aligned}$$

- U is *closed under* η if for each $r \in \eta$,
  - if $-p \in \mathtt{U}$ for every $p \in \mathtt{nup}(r)$ and $+p \in \mathtt{U}$ for every $\neg p \in \mathtt{nup}(r)$ (r must be triggered)
  - then $\mathbf{R}(r) \cap \mathtt{U} \neq \emptyset$
- U is a *justified action set* if it is a minimal superset of $\mathtt{neff}_{\mathtt{DB}}(\mathtt{U})$ closed under η
- PMA repair U of DB w.r.t. η is justified if $\mathtt{U} \cup \mathtt{neff}_{\mathtt{DB}}(\mathtt{U})$ is a justified action set

Justified repairs

Properties

- no normalisation of active part
  $\Longrightarrow$ indicates obligation reading

Outline

Outline

Background: DL-PA

Dynamic Logic of Propositional Assignments [Balbiani et al., 2013]

- PDL atomic programs $\Longrightarrow$ *atomic assignments* $+p$, $-p$
- formulas:
  - $\langle\pi\rangle\varphi$: "there is an execution of π after which φ"
- programs:
  - $+p$, $-p$: assignments
  - $\pi_1;\pi_2$: sequential composition
  - $\pi_1 \cup \pi_2$: nondeterministic composition
  - π^*: finite iteration ('Kleene star')
  - π^{c}: converse
  - $\varphi?$: test
- captures standard programming constructions:

$$\textbf{while } \varphi \textbf{ do } \pi \stackrel{\mathsf{def}}{=} (\varphi?;\pi)^*;\neg\varphi?$$

Background: semantics and properties of DL-PA

Semantics

- based on classical valuations DB
  - no Kripke models needed
- Kleene star can be eliminated (not possible in PDL)

$$\langle \pi^* \rangle \varphi \leftrightarrow \langle \pi^{\leq 2^{\mathsf{card}(\mathbb{P}_\pi)}} \rangle \varphi$$

  - consequence: all dynamic operators can be eliminated

PDL vs. DL-PA: complexity of decision problems

	PDL	DL-PA
Model checking	PTIME-complete	PSPACE-complete
Satisfiability	EXPTIME-complete	PSPACE-complete

Repairs in DL-PA

AICs as programs

for active constraint $r = \langle \mathbf{C}(r), \mathbf{R}(r) \rangle$:

$$\pi_r = \neg \mathbf{C}(r)\,?\,; \bigcup_{\alpha \in \mathbf{R}(r)} \alpha$$

Founded and justified repairs

- encoded as DL-PA programs [Feuillade et al., FI 2019]
  - copy propositional variables when checking minimality

Definition [Feuillade et al., FI 2019]

PMA repair $\mathtt{U}$ of $\mathtt{DB}$ w.r.t. η is a dynamic repair if:

$$\langle \mathtt{DB}, \mathtt{DB} \circ \mathtt{U} \rangle \in \left|\left| \mathbf{while}\ \neg \left(\bigwedge_{r \in \eta} \mathbf{C}(r) \right)\ \mathbf{do}\ \left(\bigcup_{r \in \eta} \pi_r \right) \right|\right|$$

Dynamic repairs

Properties

- obligation reading
- generalises well-founded repairs
  - repairs more [Feuillade et al., FI 2019]
- deciding the existence of a dynamic repair is PSPACE-complete

Example

$$\mathtt{DB} = \emptyset \text{ and } \eta = \{ \langle p \vee q, \{+p\} \rangle, \langle \neg p \vee q, \{+q\} \rangle \}$$

- $\{+p, +q\}$ is dynamic weak repair (also well-founded)
- no dynamic repair: minimal repair is $\{+q\}$
- bad design? (v.s.)

Reasoning about repairs in DL-PA

Reasoning tasks

prove properties *in* the logic (instead of in the metalanguage)

Let '$\texttt{repair}$' denote any of the repair programs

- set of candidate repaired databases?

  $\Longrightarrow$ interpretation of the formula $\langle \texttt{repair}^{\texttt{c}} \rangle \varphi_{\texttt{DB}}$
- is it possible at all to repair $\texttt{DB}$?

  $\Longrightarrow$ model check: $\texttt{DB} \models \langle \texttt{repair} \rangle \top$?
- is there a unique repair of $\texttt{DB}$?

  $\Longrightarrow$ model check: $\{\texttt{DB}' : \langle \texttt{DB}, \texttt{DB}' \rangle \in ||\texttt{repair}||\}$ singleton?
- can η repair any database?

  $\Longrightarrow$ validity check: $\models \langle \texttt{repair} \rangle \top$?
- ...

Outline

Event-Condition-Action (ECA) rules

Event-condition-action (ECA) rules

- when an *event* occurs
- and the *condition* is satisfied
- then some *action* is triggered

ECA rules too expressive for a logical analysis?

- much studied in databases [Ceri et al., ACM TDS 1994; Widom&Ceri, 1996; Chomicki&Marcinkowski, IC 2005,...]
- termination problems
- procedural semantics only; no declarative semantics

  > *"their lack of declarative semantics makes it difficult to understand the behavior of multiple ECAs acting together and to evaluate rule-processing algorithms in a principled way"* [Cruz Filipe, 2016]

- AIC = ECA minus events

Adding event conditions to AICs: syntax

Event-condition-action rules

$$r = \langle \mathbf{E}(r), \mathbf{C}(r), \mathbf{R}(r) \rangle$$

- $\langle \mathbf{C}(r), \mathbf{R}(r) \rangle$ is an AIC
- $\mathbf{E}(r)$ is a boolean formula built from assignments
  - partial description of last update actions

Example: functionality constraint & priority to the input

$$\eta_{\mathtt{emp}} = \{ \langle +\mathtt{emp}_{e,d_1}, \neg\mathtt{emp}_{e,d_1} \vee \neg\mathtt{emp}_{e,d_2}, \{-\mathtt{emp}_{e,d_2}\} \rangle, \\ \langle +\mathtt{emp}_{e,d_2}, \neg\mathtt{emp}_{e,d_1} \vee \neg\mathtt{emp}_{e,d_2}, \{-\mathtt{emp}_{e,d_1}\} \rangle \}$$

Adding event conditions to AICs: syntax (ctd.)

Example, ctd.

- every manager of a project carried out by a department must be an employee of that department
- if e just became manager of project p or if p was just assigned to d_1 then e should become a member of d_1
- if e has just been removed from d_1 then the project should either be removed from d_1, too, or should get a new manager

$$\eta = \eta_{\mathtt{emp}} \cup$$
$$\{\, \langle +\mathtt{mgr}_{e,p} \vee +\mathtt{prj}_{p,d_1}, \neg\mathtt{mgr}_{e,p} \vee \neg\mathtt{prj}_{p,d_1} \vee \mathtt{emp}_{e,d_1}, \{+\mathtt{emp}_{e,d_1}\}\rangle,$$
$$\langle -\mathtt{emp}_{e,d_1}, \neg\mathtt{mgr}_{e,p} \vee \neg\mathtt{prj}_{p,d_1} \vee \mathtt{emp}_{e,d_1}, \{-\mathtt{mgr}_{e,p}, -\mathtt{prj}_{p,d_1}\}\rangle \,\}$$

- when last update $+\mathtt{mgr}_{e,p}$ then repair by $\{+\mathtt{emp}_{e,d_1}, -\mathtt{emp}_{e,d_1}\}$

Adding event conditions to AICs: models

Adding immediate past events

$$\boxed{\text{model } M = \langle \mathtt{DB}, \mathbf{E} \rangle}$$

- $\mathtt{DB} \subseteq \mathbb{P}$ database
- $\mathbf{E} \subseteq \{+p : p \in \mathbb{P}\} \cup \{-p : p \in \mathbb{P}\}$ set of update actions consistent with DB:
  - if $+p \in \mathbf{E}$ then $p \in \mathtt{DB}$
  - if $-p \in \mathbf{E}$ then $p \notin \mathtt{DB}$
- semantics
  - $\langle \mathtt{DB}, \mathbf{E} \rangle \models p$ if $p \in \mathtt{DB}$
  - $\langle \mathtt{DB}, \mathbf{E} \rangle \models +p$ if $+p \in \mathbf{E}$
  - $\langle \mathtt{DB}, \mathbf{E} \rangle \models -p$ if $-p \in \mathbf{E}$

Well-founded repairs with immediate past

Definition

PMA repair $\mathtt{U}$ for $\langle \mathtt{DB}, \mathbf{E} \rangle$ w.r.t. η is well-founded if

$\mathtt{U} = \{\alpha_1, \ldots, \alpha_n\}$ (for some ordering) such that

for every α_i there is an ECA rule $r_i \in \eta$ with

- $\langle \mathtt{DB} \circ \{\alpha_1, \ldots, \alpha_{i-1}\}, \mathbf{E} \cup \{\alpha_1, \ldots, \alpha_{i-1}\} \rangle \models \mathbf{E}(r_i)$
- $\langle \mathtt{DB} \circ \{\alpha_1, \ldots, \alpha_{i-1}\}, \mathbf{E} \cup \{\alpha_1, \ldots, \alpha_{i-1}\} \rangle \not\models \mathbf{C}(r_i)$
- $\alpha_i \in \mathbf{R}(r_i)$

Example, ctd.

- well-founded repair of $\langle \{\mathtt{prj}_{p,d_1}, \mathtt{emp}_{e,d_2}\}, \{+\mathtt{mgr}_{e,p}\} \rangle$:

$$\mathtt{U} = \{+\mathtt{emp}_{e,d_2}, -\mathtt{emp}_{e,d_1}\}$$

can be captured in DL-PA

Outline

From DBs to description logic KBs

DL knowledge bases

- knowledge base KB $=$ TBox $\cup$ ABox
- TBox: *non-contingent* $\Longrightarrow$ typically stable in time
- ABox: *contingent* $\Longrightarrow$ typically changes more frequently
- ideally: TBox $\cup$ ABox $\not\models \bot$

  (and not: ABox $\models$ TBox as in databases)
- typically caused by changes to ABox
- active TBoxes should play the role of AICs

From DBs to description logic KBs

Repairs of inconsistent KBs in the DL literature

- *remove* axioms [Schlobach&Cornet, 2003]
- *weaken* axioms [Troquard et al., 2018]
- main methods: axiom pinpointing, justifications, hitting set, weakening of axioms
- usually no preference between possible repairs considered

Can we import the idea of active constraints?

A simple example

Example TBox

$$\mathcal{T} = \{\texttt{Father} \sqsubseteq \texttt{Male} \sqcap \texttt{Parent},$$
$$\texttt{OnlyChild} \sqsubseteq \forall \texttt{hasSibling}.\bot\}$$

An ABox inconsistent with $\mathcal{T}$

$$\mathcal{A} = \{\texttt{John}:\texttt{Male} \sqcap \texttt{Father} \sqcap \neg\texttt{Parent},$$
$$\texttt{Mary}:\texttt{OnlyChild},$$
$$\texttt{hasSibling}(\texttt{Mary}, \texttt{John})\}$$

A simple example

An enhanced TBox extending $\mathcal{T}$

$$\mathtt{a}\mathcal{T}_1 = \Big\{ \langle \mathtt{Father} \sqsubseteq \mathtt{Male} \sqcap \mathtt{Parent}, \{+\mathtt{Male}, +\mathtt{Parent}\}\rangle, \\ \langle \mathtt{OnlyChild} \sqsubseteq \forall \mathtt{hasSibling}.\bot, \{-\mathtt{OnlyChild}\}\rangle \Big\}$$

A repaired ABox consistent with $\mathtt{a}\mathcal{T}$

$$\mathcal{A}_1 = \{\mathtt{John}:\mathtt{Male} \sqcap \mathtt{Father} \sqcap \mathtt{Parent}, \\ \mathtt{Mary}:\neg\mathtt{OnlyChild}, \\ \mathtt{hasSibling}(\mathtt{Mary}, \mathtt{John})\}$$

A simple example

An enhanced TBox extending $\mathcal{T}$

$$\mathtt{a}\mathcal{T}_2 = \Big\{ \langle \mathtt{Father} \sqsubseteq \mathtt{Male} \sqcap \mathtt{Parent}, \{-\mathtt{Father}\}\rangle,$$
$$\langle \mathtt{OnlyChild} \sqsubseteq \forall \mathtt{hasSibling}.\bot, \{-\mathtt{hasSibling}.\top\}\rangle \Big\}$$

A repaired ABox consistent with $\mathtt{a}\mathcal{T}$

$$\mathcal{A}_2 = \{\mathtt{John}:\mathtt{Male} \sqcap \neg\mathtt{Father} \sqcap \neg\mathtt{Parent},$$
$$\mathtt{Mary} : \mathtt{OnlyChild}\}$$

Repairs based on dynamic TBoxes

Challenges

1. ABoxes have concept constructors & complex concepts
   $\Longrightarrow$ 'atomic' update actions usually insufficient
2. *closed world* semantics vs. *open world* semantics
   $\Longrightarrow$ *satisfiability checking* instead of *model checking*
3. *removing* vs. *forgetting* concepts
   $\Longrightarrow$ choice

Proposals

- [Rantsoudis et al., DL 2017]: syntactic approach
- [Feuillade et al., DL 2018]: semantic approach
  - logic $\mathtt{dyn}\mathcal{ALCO}$
- [Rantsoudis, PhD 2018]

In summary

- reexamined intuitions behind active integrity constraints
  - permission vs. obligation to choose update action
    - obligation reading deserves more investigation
  - principle of extensionality
- surveyed AICs via dynamic logic
  - introduction of dynamic repairs
  - captured trigger events $\Longrightarrow$ ECA rules
- sketched repairs based on active TBoxes

Looking ahead

- good postulates for database repair?
- better complexity results for repairs based on active TBoxes?
- application to defeasible DLs

Thank You!