Stratification-based Criteria for Checking Chase Termination*

Sergio Greco, Francesca Spezzano, and Irina Trubitsyna

DEIS, Università della Calabria, 87036 Rende, Italy
{greco,fspezzano,irina}@deis.unical.it

Abstract. Several database areas such as data exchange and integration share the problem of fixing database instance violations with respect to a set of constraints. The chase algorithm solves such violations by inserting tuples and setting the value of nulls. Unfortunately, the chase algorithm may not terminate and the problem of deciding whether the chase process terminates is undecidable. Recently there has been an increasing interest in the identification of sufficient structural properties of constraints which guarantee that the chase algorithm terminates.
In this paper we present more general criteria for chase termination. We first present extensions of the well-known stratification condition and, then, introduce a new criterion, called local stratification (LS), which generalizes both super-weak acyclicity and stratification-based criteria (including the class of constraints which are inductively restricted).

1 Introduction

Several database areas such as data exchange and integration share the problem of fixing database instance violations with respect to a set of constraints [1–6]. The chase algorithm solves such violations by inserting tuples and setting the value of nulls. Unfortunately, the chase algorithm may not terminate and the problem of deciding whether the chase process terminates is undecidable. Recently there has been an increasing interest in the identification of sufficient structural properties of constraints which guarantee that the chase algorithm terminates. Most of these criteria extend weak acyclicity (WA) [7] by analyzing nulls propagation (e.g. the safety criterion (SC) [8]) and constraints firing (e.g. c-stratification (CStr) and inductive restriction (IR) [9], super-weak acyclicity (SwA) [10]).

The idea underlying c-stratification, also used in the IR and SwA criteria, is to consider, in the propagation of nulls, how constraints may fire each other. However, there are simple cases where current criteria are not able to understand that all chase sequences are finite.

Example 1. Consider the following set of constraints Σ_1 consisting of the TGD

$$\forall x \forall y \; E(x,y) \wedge E(y,x) \rightarrow \exists z \; E(y,z) \wedge E(z,x)$$

We can distinguish two cases. If we suppose to have a database instance $D_1 = \{E(a,b), E(b,a)\}$ we have that the TGD is not satisfied, but the constraint will be applied only once by the chase as the resulting database $D_1' = \{E(a,b), E(b,a),$

* Extended Abstract

Nicola Ferro and Letizia Tanca (Eds.): SEBD 2012, Edizioni Libreria Progetto, Padova, Italia
ISBN: 978-88-96477-23-6,

$E(b, \eta_1), E(\eta_1, a)\}$ is consistent. Otherwise, if we suppose to have a database instance $D_2 = \{E(a, a)\}$, it is already consistent, and no chase step will be applied. It is easy to see that the chase is always terminating for all database instances, but none of the existing criteria guaranteeing the termination of all chase sequences is able to recognize it as terminating. □

In order to cope with this problem, we first propose a new extension of c-stratification, called *WA-stratification* (*WA-Str*) and then introduce a new criterion, called *local stratification* (*LS*), which generalizes both super-weak acyclicity and inductive restriction). Moreover, both *WA-Str* and *LS* guarantee the termination of all chase sequences, for all database instances, in polynomial time.

2 Preliminaries

We introduce the following disjunct sets of symbols: (i) an infinite set $Consts$ of *constants*, (ii) an infinite set $Nulls$ of *labeled nulls* and (iii) an infinite set $Vars$ of variables. A *relational schema* $\mathbf{R}$ is a set of relational predicates R, each with its associated arity $ar(R)$. An *instance* of a relational predicate R of arity n is a set of ground atoms in the form $R(c_1, \ldots, c_n)$, where $c_i \in Consts \cup Nulls$. Such (ground) atoms are also called tuples or facts. We denote by D a database instance constructed on $Consts$ and by J, K the database instances constructed on $Consts \cup Nulls$. Given an instance K, $Nulls(K)$ (resp. $Consts(K)$) denotes the set of labeled nulls (resp. constants) occurring in K. An *atomic formula* (or *atom*) is of the form $R(t_1, \ldots, t_n)$ where R is a relational predicate, $t_1, \ldots, t_n$ are terms belonging to the domain $Consts \cup Vars$ and $n = ar(R)$.

Given a relational schema $\mathbf{R}$, a *tuple generating dependency* (TGD) over $\mathbf{R}$ is a formula of the form $\forall \mathbf{x} \forall \mathbf{z}\ \phi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}\ \psi(\mathbf{x}, \mathbf{y})$, where $\phi(\mathbf{x}, \mathbf{z})$ and $\psi(\mathbf{x}, \mathbf{y})$ are conjunctions of atomic formulas over $\mathbf{R}$; $\phi(\mathbf{x}, \mathbf{z})$ is called the *body* of r, denoted as $Body(r)$, while $\psi(\mathbf{x}, \mathbf{y})$ is called the *head* of r, denoted as $Head(r)$. An *equality generating dependency* (EGD) over $\mathbf{R}$ is a formula of the form $\forall \mathbf{x}\ \phi(\mathbf{x}) \rightarrow x_1 = x_2$, where x_1 and x_2 are among the variables in $\mathbf{x}$.

In the following we will often omit the universal quantification, since we assume that variables appearing in the body are universally quantified and variables appearing only in the head are existentially quantified. In some cases we also assume that the head and body conjunctions are sets of atoms.

Definition 1 (Homomorphism [7]). Let K_1 and K_2 be two instances over $\mathbf{R}$ with values in $Consts \cup Nulls$. A *homomorphism* $h : K_1 \rightarrow K_2$ is a mapping from $Consts(K_1) \cup Nulls(K_1)$ to $Consts(K_2) \cup Nulls(K_2)$ such that: (1) $h(c) = c$, for every $c \in Consts(K_1)$, and (2) for every fact $R_i(t)$ of K_1, we have that $R_i(h(t))$ is a fact of K_2 (where, if $t = (a_1, ..., a_s)$, then $h(t) = (h(a_1), ..., h(a_s))$). □

Similar to homomorphisms between instances, a homomorphism h from a conjunctive formula $\phi(\mathbf{x})$ to an instance J is a mapping from the variables $\mathbf{x}$ to $Consts(J) \cup Nulls(J)$ such that for every atom $R(x_1, \ldots, x_n)$ of $\phi(\mathbf{x})$ the fact $R(h(x_1), \ldots, h(x_n))$ is in J.

For any database instance D and set of constraints Σ over a database schema $\mathbf{R}$, a *solution* for (D, Σ) is an instance J such that $D \subseteq J$ and $J \models \Sigma$ (i.e. J satisfies all constraints in Σ). A *universal solution* J is a solution such that for every solution J' there exists a homomorphism $h : J \rightarrow J'$. The set of universal solutions for (D, Σ) will be denoted by $USol(D, \Sigma)$.

Nicola Ferro and Letizia Tanca (Eds.): SEBD 2012, Edizioni Libreria Progetto, Padova, Italia
ISBN: 978-88-96477-23-6,

Definition 2 (Chase step [7]). Let K be a database instance.

(1) Let r be a TGD $\phi(\mathbf{x},\mathbf{z}) \rightarrow \exists \mathbf{y}\, \psi(\mathbf{x},\mathbf{y})$. Let h be a homomorphism from $\phi(\mathbf{x},\mathbf{z})$ to K such that there is no extension of h to a homomorphism h' from $\phi(\mathbf{x},\mathbf{z}) \wedge \psi(\mathbf{x},\mathbf{y})$ to K. We say that r can be *applied* to K with homomorphism h. Let K' be the union of K with the set of facts obtained by: (a) extending h to h' such that each variable in $\mathbf{y}$ is assigned a fresh labeled null, followed by (b) taking the image of the atoms of ψ under h'. We say that the result of applying r to K with h is K', and write $K \overset{r,h}{\rightarrow} K'$.

(2) Let r be an EGD $\phi(\mathbf{x}) \rightarrow x_1 = x_2$. Let h be a homomorphism from $\phi(\mathbf{x})$ to K such that $h(x_1) \neq h(x_2)$. We say that r can be *applied* to K with homomorphism h. More specifically, we distinguish two cases. (a) If both $h(x_1)$ and $h(x_2)$ are in *Consts* the result of applying r to K with h is "failure", and $K \overset{r,h}{\rightarrow} \perp$. (b) Otherwise, let K' be K where we identify $h(x_1)$ and $h(x_2)$ as follows: if one is a constant, then the labeled null is replaced everywhere by the constant; if both are labeled nulls, then one is replaced everywhere by the other. We say that the result of applying r to K with h is K', and write $K \overset{r,h}{\rightarrow} K'$. □

Definition 3 (Chase [7]). Let Σ be a set of TGDs and EGDs, and let K be an instance.

(1) A *chase sequence of K with Σ* is a sequence (finite or infinite) of chase steps $K_i \overset{r,h_i}{\rightarrow} K_{i+1}$, with $i = 0, 1, ...$, $K_0 = K$ and r a dependency in Σ.

(2) A *finite chase of K with Σ* is a finite chase sequence $K_i \overset{r,h_i}{\rightarrow} K_{i+1}$, $0 \leq i < m$, with the requirement that either (a) $K_m = \perp$ or (b) there is no dependency r of Σ and there is no homomorphism h_m such that r can be applied to K_m with h_m. We say that K_m is the result of the finite chase. We refer to case (a) as the case of a *failing finite chase* and we refer to case (b) as the case of a *successful finite chase.* □

In [7] it has been shown that, for any instance D and set of constraints Σ: (i) if J is the result of some successful finite chase of $\langle D, \Sigma \rangle$, then J is a universal solution; (ii) if some failing finite chase of $\langle D, \Sigma \rangle$ exists, then there is no solution.

Chase Termination Criteria. We now present a brief overview on the well-known chase termination conditions that guarantee for every database D the termination of all chase sequences in PTIME in the size of D. Given a criterion C, the class of constraints satisfying C will be denoted by $\mathcal{C}$.

Weak acyclicity. Let Σ be a set of TGDs over a database schema $\mathbf{R}$, then $pos(\Sigma)$ denotes the set of positions R_i such that R denotes a relational predicate of $\mathbf{R}$ and there is an R-atom appearing in Σ. Weak acyclicity (*WA*) is based on the construction of a directed graph $dep(\Sigma) = (pos(\Sigma), E)$, called the *dependency graph*, where E is defined as follows. For every TGD $\phi(\mathbf{x},\mathbf{z}) \rightarrow \exists \mathbf{y} \psi(\mathbf{x},\mathbf{y})$ in Σ, then: i) for every x in $\mathbf{x}$ occurring in position R_i in ϕ and in position S_j in ψ, add an edge $R_i \rightarrow S_j$ (if it does not already exist); ii) for every x in $\mathbf{x}$, appearing in position R_i in ϕ and for every y in $\mathbf{y}$ appearing in position T_k in ψ, add a special edge $R_i \overset{*}{\rightarrow} T_k$ (if it does not already exist). Σ is *weakly acyclic* if $dep(\Sigma)$ has no cycle going through a special edge.

Safety. The safety condition (*SC*) [9] is based on the notion of affected positions. An *affected position* denotes a position in which null values may appear,

Nicola Ferro and Letizia Tanca (Eds.): SEBD 2012, Edizioni Libreria Progetto, Padova, Italia
ISBN: 978-88-96477-23-6, Copyright (c) 2012 - Edizioni Libreria Progetto and the authors

that is it can also take values from $Nulls$. A position R_i is said to be affected if there is a constraint $r : \phi(\mathbf{x}, \mathbf{z}) \rightarrow \exists \mathbf{y}\, \psi(\mathbf{x}, \mathbf{y})$ in Σ and either i) there is a variable y in $\mathbf{y}$ appearing in position R_i in ψ, or ii) there is a variable x in $\mathbf{x}$ appearing both in position R_i in ψ and only in affected positions in the body of r. The set of affected positions of Σ is denoted by $aff(\Sigma)$.

Given a set of TGDs Σ, the *propagation graph* of Σ, denoted as $prop(\Sigma) = (aff(\Sigma), E')$, is a subset of $dep(\Sigma) = (pos(\Sigma), E)$ such that E' contains the edges in E whose positions are affected (since $aff(\Sigma) \subseteq pos(\Sigma)$). Moreover, Σ is said to be safe if $prop(\Sigma)$ does not contain cycles with special edges.

C-Stratification. The idea behind *c-stratification* ($CStr$) [11, 9] is to decompose the set of constraints into independent subsets, where each subset consists of constraints that may fire each other, and check each component separately for weak acyclicity. Given a set of constraints Σ and two constraints $r_1, r_2 \in \Sigma$, we say that $r_1 \prec_c r_2$ iff there exists a relational database instance K_1 and two homomorphisms h_1 and h_2 such that i) $K_1 \xrightarrow{*,r_1,h_1} K_2$, ii) $K_2 \not\models h_2(r_2)$ and iii) $K_1 \models h_2(r_2)$, where the oblivious chase step $K_1 \xrightarrow{*,r_1,h_1} K_2$ states that there is a homomorphism h_1' extending h_1 which associates every existentially variable y in $h_1(r_1)$ to a fresh labeled null. Intuitively, $r_1 \prec_c r_2$ means that firing r_1 can cause the firing of r_2. We say that Σ is c-stratified iff the constraints in every cycle of the *c-chase graph* $G(\Sigma) = (\Sigma, \{(r_1, r_2) | r_1 \prec_c r_2\})$ are weakly acyclic.

Inductive Restriction. The inductive restriction criterion (IR) extends both $CStr$ and SC by partitioning constraints in a more refined way. In particular, it first computes the graph $(G'(\Sigma)) \subseteq G(\Sigma)$ and partition Σ into $\Sigma_1, ..., \Sigma_n$, where each Σ_i is a set of dependencies defining a strongly connected components in $G'(\Sigma)$, next, if $n = 1$ the safety criterion is applied to Σ, otherwise the IR criterion is applied inductively to each Σ_i.

Super-weak acyclicity. The super-weak acyclicity (SwA) [10] builds a *trigger graph* $\Upsilon(\Sigma) = (\Sigma, E)$ where edges define relations among constraints. An edge $r_i \rightsquigarrow r_j$ means that a null value introduced by a constraint r_i is propagated (directly or indirectly) into the body of r_j.

Let Σ be a set of TGDs and let $sk(\Sigma)$ be the logic program obtained by skolemizing Σ, i.e. by replacing each existentially quantified variable y appearing in the head of a TGD r by the skolem function $f_y^r(\mathbf{x})$, where $\mathbf{x}$ is the set of variables appearing both in the body and in the head of r. A *place* is a pair (a, i) where a is an atom of $sk(\Sigma)$ and $0 \leq i \leq ar(a)$. Given a TGD r and an existential variable y in the head of r, $Out(r, y)$ denotes the set of places (called *output places*) in the head of $sk(r)$ where a term of the form $f_y^r(\mathbf{x})$ occurs. Let r be a TGD r and let x be a universal variable of r, $In(r, x)$ denotes the set of places (called *input places*) in the body of r where x occurs.

Given a set of variables V, a substitution θ of V is a function mapping each $v \in V$ to a finite term $\theta(v)$ built upon constants and function symbols. Two places (a, i) and (a', i) are unifiable and we write $(a, i) \sim (a', i)$ iff there exist two substitutions θ and θ' of (respectively) the variables a and a' such that $a[\theta] = a'[\theta']$. Given two sets of places Q and Q' we write $Q \sqsubseteq Q'$ iff for all $q \in Q$ there exists some $q' \in Q'$ such that $q \sim q'$.

For any set Q of places, $Move(\Sigma, Q)$ denotes the smallest set of places Q' such that $Q \subseteq Q'$, and for every constraint $r = B_r \rightarrow H_r$ in $sk(\Sigma)$ and every

Nicola Ferro and Letizia Tanca (Eds.): SEBD 2012, Edizioni Libreria Progetto, Padova, Italia
ISBN: 978-88-96477-23-6, Copyright (c) 2012 - Edizioni Libreria Progetto and the authors

variable x, if $\Pi_x(B_r) \sqsubseteq Q'$ then $\Pi_x(H_r) \subseteq Q'$, where $\Pi_x(B_r)$ and $\Pi_x(H_r)$ denote the sets of places in B_r and H_r where x occurs.
Given a set Σ of TGDs and two TGDs $r_1, r_2 \in \Sigma$, we say that r_1 triggers r_2 in Σ and write $r_1 \rightsquigarrow r_2$ iff there exists an existential variable y in the head of r_1, and a universal variable x_2 occurring both in the body and head of r_2 such that $In(r_2, x) \sqsubseteq Move(\Sigma, Out(r_1, y))$. A set of constraints Σ is super-weakly acyclic iff the trigger graph $\Upsilon(\Sigma) = (\Sigma, \{(r_1, r_2) | r_1 \rightsquigarrow r_2\})$ is acyclic. W.r.t. other criteria, SwA also takes into account that a variable may occur more than once in the same atom. $\mathcal{SwA}$ extends $\mathcal{SC}$, but is not comparable with $\mathcal{CSt}r$.

3 WA-Stratification

We start by introducing some improvements for the c-stratification criterion. First of all, observe that c-stratification does not specify what kind of cycles are checked (i.e. simple or general) [12]. Checking simple cycles is not correct as it may not consider all possible chase sequences, but checking general cycles, means that for each strongly connected component there is one cycle including all nodes in the component which subsumes all other cycles on the same component (in terms of constraints to be considered). Thus, a first observation on (c-)stratification (in terms of correctness, if simple cycles are considered, or in terms of efficiency, if all cycles are considered) is that it refers to cycles instead of strongly connected components. A further observation is that it uses oblivious chase for checking termination of standard chase and its applicability is limited.

Definition 4 (WA-Stratification). Given a set of dependencies Σ and $r_1, r_2 \in \Sigma$, we say that $r_1 < r_2$ iff there exist a relational database instance K, homomorphisms h_1, h_2 and a set S of atoms, such that

1. $K_1 \not\models h_1(r_1)$,
2. $K_1 \overset{r_1, h_1}{\rightarrow} K_2$,
3. $K_1 \cup S \models h_2(r_2)$,
4. $K_2 \cup S \not\models h_2(r_2)$ and
5. $Null(S) \cap (Null(K_2) - Null(K_1)) = \emptyset$ (i.e. S does not contain new null values introduced in K_2).

We say that Σ is *WA-stratified* (*WA-Str*) iff the constraints in every nontrivial strongly connected component of the *firing graph* $\Gamma(\Sigma) = (\Sigma, \{(r_1, r_2) | r_1 < r_2\})$ are weakly acyclic. □

With respect to c-stratification, *WA-Str* also considers in the satisfaction of constraint r_2, in addition to the database K_1, a set of atoms S (cond. (3)) and atoms in S cannot contain null values introduced in the application of the constraint r_1 (cond. (5)). Moreover, since we are considering strongly connected components (instead of cycles) these components must not be trivial, that is they must have at least one edge, otherwise the constraint cannot be fired cyclically. As a further important observation, in the above definition we consider standard chase for both constructing the graph $\Gamma(\Sigma)$ and checking weak acyclicity.

Example 2. Consider again the set of constraints Σ_1 of Example 1. It is easy to see that, by considering standard chase, does not exist an initial database instance such that the constraint can fire itself, while, by considering the oblivious chase, the constraint fires itself ad infinitum. Thus, the set of constraints Σ_1 is WA-stratified, but not c-stratified. □

Nicola Ferro and Letizia Tanca (Eds.): SEBD 2012, Edizioni Libreria Progetto, Padova, Italia
ISBN: 978-88-96477-23-6,

The following proposition states that *WA-Str* criterion is more general than *CStr* and is not comparable with *SC*. Consequently it is not comparable even with $\mathcal{SwA}$ as $\mathcal{SC}$ is strictly contained in $\mathcal{SwA}$ and $\mathcal{CS}tr$ is not comparable with $\mathcal{SwA}$.

Proposition 1. $\mathcal{CS}tr \subsetneq \mathcal{WA}\text{-}\mathcal{S}tr$ and $\mathcal{SC} \nparallel \mathcal{WA}\text{-}\mathcal{S}tr$. □

It is important to observe that *WA-Str* criterion could be improved by testing safety instead of weak acyclicity over the firing graph. Further improvements could be obtained by considering super-weak acyclicity instead of safety.

Definition 5 (SC-Stratification and SwA-Stratification). Given a set of TDGs Σ, we say that (1) Σ is *SC-stratified* (*SC-Str*) if the constraints in every strongly connected component of the firing graph $\Gamma(\Sigma)$ are safe, and (2) Σ is *SwA-stratified* (*SwA-Str*) if the constraints in every strongly connected component of the firing graph $\Gamma(\Sigma)$ are super-weak acyclic. □

We now analyze the complexity of the above criteria starting by defining a bound on the complexity of the firing problem, i.e. the complexity of checking whether $r_1 < r_2$.

Lemma 1. Let $r_1 : \phi_1 \rightarrow A_1 \wedge \cdots A_k$ and $r_2 : B_1 \wedge \cdots B_n \rightarrow \psi_2$ be two TGDs. The problem of checking whether $r_1 < r_2$ is bounded by $O((k+1)^n)$. □

Although the theoretical complexity of the "firing" problem is exponential, in most cases it is very low (e.g. inclusion dependencies, multivalued dependencies [13]), as usually the number n of body atoms in the fired constraint r_2 is small and the number of atoms in the head of constraint r_1 which could be used to fire r_2 through their unification with B_i (i.e. $k_i > 1$) is even smaller. Indeed, if the number of atoms in the body of r_2 is bounded by a constant, the firing problem is in PTIME. Significative subclasses of constraints for which the firing problem becomes polynomial could be identified, but this is not the aim of this paper.

In the following, for a given set of constraints Σ, we shall denote with C_{ij} the complexity of the problem of checking whether $r_i < r_j$, for $r_i, r_j \in \Sigma$, and with $C_m = max\{C_{ij} | r_i, r_j \in \Sigma\}$.

Proposition 2. Let Σ be a set of TGDs, D be a database Then:

- the problem of checking whether Σ is WA-stratified (resp. SC-stratified, SwA-stratified) is bounded by $O(C_m \times |\Sigma|^2)$;
- if Σ is WA-stratified (resp. SC-stratified, SwA-stratified), the length of every chase sequence of Σ over D is polynomial in the size of D. □

The class of constraints satisfying criterion *C-Str*, for $C \in \{WA, SC, SwA\}$, will be denoted by $\mathcal{C}\text{-}\mathcal{S}tr$. The next theorem states the relationships among the above mentioned criteria and other previously defined conditions.

Theorem 1.

1. $\mathcal{WA}\text{-}\mathcal{S}tr \subsetneq \mathcal{SC}\text{-}\mathcal{S}tr \subsetneq \mathcal{SwA}\text{-}\mathcal{S}tr$,
2. for $C \in \{WA, SC, SwA\}$, $\mathcal{C} \subsetneq \mathcal{C}\text{-}\mathcal{S}tr$ and
3. $\mathcal{SR} \nparallel \mathcal{SwA}\text{-}\mathcal{S}tr$ and $\mathcal{IR} \nparallel \mathcal{SwA}\text{-}\mathcal{S}tr$. □

Nicola Ferro and Letizia Tanca (Eds.): SEBD 2012, Edizioni Libreria Progetto, Padova, Italia
ISBN: 978-88-96477-23-6, Copyright (c) 2012 - Edizioni Libreria Progetto and the authors

4 Local Stratification

It is trivial that more powerful criteria could be defined by composing criteria which are not comparable. We next present a different generalization of super-weak acyclicity which also generalizes the class $\mathcal{IR}$.

We start by introducing a notion of *fireable place*. We say that a place q appearing in the body of constraint r could be fired by a place q' appearing in the head of constraint r', denoted by $q' < q$, if $q \sim q'$ and $r' < r$. Given two sets of places Q and Q' we say that Q could be fired by Q', denoted by $Q' < Q$ iff for all $q \in Q$ there exists some $q' \in Q'$ such that $q' < q$.

Given a set Q of places, we define $MOVE(\Sigma, Q)$ as the smallest set of places Q' such that $Q \subseteq Q'$, and for every constraint $r = B_r \rightarrow H_r$ in $sk(\Sigma)$ and every variable x, if $Q' < \Pi_x(B_r)$ then $\Pi_x(H_r) \subseteq Q'$, where $\Pi_x(B_r)$ and $\Pi_x(H_r)$ denote the sets of places in B_r and H_r where x occurs.

With respect to the function $Move$, the new function $MOVE$ here considered takes into account the firing of places and not only the unification of places.

Definition 6 (Local Stratification). Given a set Σ of TGDs and two TGDs $r_1, r_2 \in \Sigma$, we say that r_1 triggers r_2 in Σ and write $r_1 \hookrightarrow r_2$ iff there exists an existential variable y in the head of r_1, and a universal variable x occurring both in the body and head of r_2 such that $MOVE(\Sigma, Out(r_1, y)) < In(r_2, x)$. A set of constraints Σ is *locally stratified* (LS) iff the trigger graph $\Delta(\Sigma) = \{(r_1, r_2) | r_1 \hookrightarrow r_2\}$ is acyclic. □

Proposition 3. For every set of TGDs Σ and for every database D
- the problem of checking whether Σ is locally stratified is bounded by $O(C_m \times |\Sigma|^2)$;
- if Σ is locally stratified, the length of every chase sequence of Σ over D is polynomial in the size of D. □

The below theorem states that the class of locally stratified constraints (denoted by $\mathcal{LS}$) is more general than $\mathcal{SwA}$-$\mathcal{S}tr$ and $\mathcal{IR}$.

Theorem 2. $\mathcal{SwA}$-$\mathcal{S}tr \subsetneq \mathcal{LS}$ and $\mathcal{IR} \subsetneq \mathcal{LS}$. □

The next example shows that the containment of $\mathcal{SwA}$-$\mathcal{S}tr \cup \mathcal{IR}$ in $\mathcal{LS}$ is strict.

Example 3. The following set of constraints Σ_3 is locally stratified, but it is neither super-weakly acyclic nor inductively restricted:

$$r_1 : \; N(x) \rightarrow \exists\, y \exists z \; E(x, y) \wedge S(z, y)$$
$$r_2 : \; E(x, y) \wedge S(x, y) \rightarrow N(y)$$
$$r_3 : \; E(x, y) \rightarrow E(y, x)$$

Considering SwA, we have that $Move(\Sigma, Out(r_1, y)) = \{p_3, p_5, p_{10}, p_{13}, p_2, p_{14}\}$ and $In(r_1, x) = \{p_1\} \sqsubseteq Move(\Sigma, Out(r_1, y))$. The trigger graph is cyclic as $r_1 \rightsquigarrow r_1$ and, therefore, Σ_3 is not super-weakly acyclic. As $r_1 < r_3 < r_2 < r_1$ we have that Σ_3 is not SwA-Str as well. Σ_3 is not IR as $r_1 \prec_c r_3 \prec_c r_2 \prec_c r_1$ and for each pair of constraints r_i, r_j such that $r_i \prec_c r_j$, it is possible to construct a database containing null values in positions E_1, E_2, N_1 and S_2 such that whenever r_i fires r_j a null value is propagated from the head of r_i to the head of r_j.
Moreover, as $r_1 \not\prec r_1$ ($MOVE(\Sigma, (r_1, y)) = \{p_3, p_5, p_{13}\}$ and $In(r_1, x) = \{p_1\} \sqsubseteq MOVE(\Sigma, Out(r_1, y))$), $\Delta(\Sigma_3)$ is acyclic and, thus, Σ_3 is locally stratified. □

Nicola Ferro and Letizia Tanca (Eds.): SEBD 2012, Edizioni Libreria Progetto, Padova, Italia
ISBN: 978-88-96477-23-6,

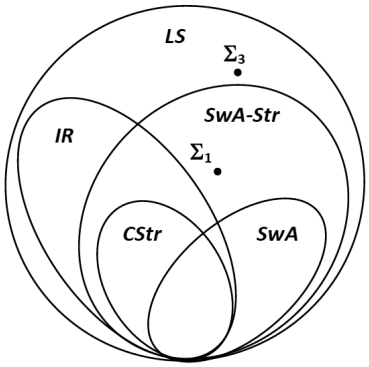

Fig. 1. Criteria Relationships.

5 Conclusions

In this paper we have proposed new criteria for checking chase termination on the base of constraints structural properties. We have shown that the local stratification criterium here introduced, strictly generalize criteria previously proposed in the literature. The relationships among previous criteria and the ones here proposed are reported in Figure 1.

References

1. L. E. Bertossi, "Consistent query answering in databases," *SIGMOD Record*, vol. 35, no. 2, 2006.
2. J. Chomicki, "Consistent query answering: Five easy pieces," in *ICDT*, 2007.
3. G. DeGiacomo, D. Lembo, M. Lenzerini, and R. Rosati, "On reconciling data exchange, data integration, and peer data management," in *PODS*, 2007.
4. R. Fagin, P. G. Kolaitis, and L. Popa, "Data exchange: getting to the core," *ACM Trans. Database Syst.*, vol. 30, no. 1, 2005.
5. P. G. Kolaitis, J. Panttaja, and W. C. Tan, "The complexity of data exchange," in *PODS*, 2006.
6. M. Lenzerini, "Data integration: A theoretical perspective," in *PODS*, 2002.
7. R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa, "Data exchange: semantics and query answering," *Theor. Comput. Sci.*, vol. 336, no. 1, 2005.
8. M. Meier, M. Schmidt, and G. Lausen, "On chase termination beyond stratification," *PVLDB*, vol. 2, no. 1, 2009.
9. M. Meier, M. Schmidt, and G. Lausen, "On chase termination beyond stratification," *CoRR*, vol. abs/0906.4228, 2009.
10. B. Marnette, "Generalized schema-mappings: from termination to tractability," in *PODS*, 2009.
11. A. Deutsch, A. Nash, and J. B. Remmel, "The chase revisited," in *PODS*, 2008.
12. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. The MIT Press, 2001.
13. S. Abiteboul, R. Hull, and V. Vianu, *Foundations of Databases*. Addison-Wesley, 1995.

Nicola Ferro and Letizia Tanca (Eds.): SEBD 2012, Edizioni Libreria Progetto, Padova, Italia
ISBN: 978-88-96477-23-6,