ADDING PATH-FUNCTIONAL DEPENDENCIES TO THE GUARDED TWO-VARIABLE FRAGMENT WITH COUNTING

GEORGIOS KOURTIS AND IAN PRATT-HARTMANN

School of Computer Science, The University of Manchester, UK
e-mail address: {kourtisg,ipratt}@cs.man.ac.uk

ABSTRACT. The satisfiability and finite satisfiability problems for the two-variable guarded fragment of first-order logic with counting quantifiers, a database, and path-functional dependencies are both ExpTime-complete.

1. INTRODUCTION

In the theory of information systems, a *path-functional dependency* is a data constraint stating that individuals yielding identical values on application of various sequences of functions must themselves be identical. For example, in a database of customers, we may wish to impose the condition that no two individuals have the same customer ID:

$$\forall x \forall y (\text{custID}(x) = \text{custID}(y) \rightarrow x = y). \tag{1.1}$$

Alternatively, we may wish to impose the condition that no two customers have both the same name and postal codes:

$$\forall x \forall y (\text{name}(x) = \text{name}(y) \land$$
$$\text{postCode}(\text{address}(x)) = \text{postCode}(\text{address}(y)) \rightarrow x = y). \tag{1.2}$$

The formula in (1.1) has a single equality on the left-hand side of the implication; and therefore we speak of the condition it expresses as a *unary* path-functional dependency. By contrast, (1.2) has a conjunction of two equalities on the left-hand side of the implication; and we speak of the condition it expresses as a *binary* path-functional dependency.

Path-functional dependencies were introduced by Weddell [Wed89] (see also [IW94, BP00]), and are of particular interest when combined with data-integrity constraints represented using formulas of some *description logics* [TW05b, TW05a]. In this connection, Toman and Weddell [TW08] report various results on the decidability of the satisfiability problem for description logics extended with various forms of path-functional constraints. The purpose of the present paper is to extend those results to a more general logical setting, namely that of the two-variable guarded fragment with counting quantifiers.

The *guarded fragment* of first-order logic, denoted $\mathcal{GF}$, was introduced by Andréka *et al.* [ANvB98] as a generalization of modal logic, in an attempt to explain the latter's good computational behaviour (see e.g. [Var96, Grä99b]). Roughly speaking, $\mathcal{GF}$ is the fragment of first-order logic in which all quantification is relativized (guarded) by atoms featuring all variables free in the context of quantification. Thus, for example, in the guarded formula

$$\forall x(\text{customer}(x) \rightarrow \exists y(\text{lastName}(x, y) \wedge \text{string}(y))),$$

stating that every customer has a last name (which is of type string), the quantifier $\exists y$ is relativized by the binary atom lastName(x, y). This fragment has the so-called finite model property: if a $\mathcal{GF}$-formula is satisfied in some structure, it is satisfied in some finite structure. The problem of determining whether a given $\mathcal{GF}$-formula is satisfiable is 2-EXPTIME-complete [Grä99a], dropping to EXPTIME-complete if the number of variables in formulas is less than some fixed limit $k \geq 2$. Of particular interest in this regard is the case $k = 2$, since this fragment remains EXPTIME-complete even when we add so-called *counting quantifiers* ('there exist at most/at least/exactly m x such that . . . '). The resulting logic, denoted $\mathcal{GC}^2$, allows us to state, for example, that customers are uniquely identified by their customer identification numbers:

$$\forall x(\text{string}(x) \rightarrow \exists_{\leq 1} y(\text{custID}(y, x) \wedge \text{customer}(y))).$$

Indeed, $\mathcal{GC}^2$ subsumes the well-known description logic $\mathcal{ALCQI}$.

The logic $\mathcal{GC}^2$ lacks the finite model property. Nevertheless, the satisfiability and finite satisfiability problems for this fragment are both decidable, and in fact EXPTIME-complete [Kaz04, PH07].

From a logical point of view, a *database* is simply a finite set of ground literals—i.e. atomic statements or negated atomic statements featuring only constants as arguments:

$$\text{custID}(\#111001, 5030875), \quad \text{firstName}(\#111001, \text{'Fred'}), \quad \ldots$$

We assume that such data are subject to *constraints* presented in the form of a logical theory—as it might be, a collection of formulas of $\mathcal{GC}^2$, for instance. Those constraints may imply the existence of individuals not mentioned in the database: indeed, since $\mathcal{GC}^2$ lacks the finite model property, they may consistently imply the existence of infinitely many such individuals. We denote by $\mathcal{GC}^2\mathcal{D}$ the logic that results from adding a database to $\mathcal{GC}^2$; and we denote by $\mathcal{GC}^2\mathcal{DK}$ the logic that results from adding unary and binary path-functional dependencies to $\mathcal{GC}^2\mathcal{D}$. (A formal definition is given in Sec. 2.)

The *satisfiability problem* for $\mathcal{GC}^2\mathcal{DK}$ asks the following: given a database Δ, a $\mathcal{GC}^2$-formula φ and a finite set of unary and binary path-functional dependencies K, are the data in Δ consistent with the constraints expressed in φ and K? The *finite satisfiability problem* for $\mathcal{GC}^2\mathcal{DK}$ asks the same question, but subject to the assumption of a finite universe. We show that both of these problems are EXPTIME-complete, the same as the corresponding problems for $\mathcal{GC}^2$ alone. That is: adding databases and unary or binary path-functional dependencies to $\mathcal{GC}^2$ does not change the complexity class of the satisfiability and finite satisfiability problems.

Our strategy will be to reduce the satisfiability and finite satisfiability problems for $\mathcal{GC}^2\mathcal{DK}$ to the corresponding problems for $\mathcal{GC}^2\mathcal{D}$, by showing how unary and binary path-functional dependencies can be systematically eliminated. The case of unary path-functional dependencies is simple, and is dealt with in the preliminary material of Sec. 2. Binary path-functional dependencies, however, present a greater challenge, and occupy the whole of Sections 3–6. Our point of departure here is the familiar observation that models of

$\mathcal{GC}^2$-formulas, when viewed as graphs, can be assumed to contain no 'short' cycles. We use this observation to characterize putative violations of binary path-functional dependencies in terms of the occurrence of certain acyclic subgraphs, which can then be forbidden by writing additional $\mathcal{GC}^2$-formulas. The remainder of the paper, Sections 7–9, is devoted to establishing that the satisfiability and finite satisfiability problems for $\mathcal{GC}^2\mathcal{D}$ are both in ExpTime. The argument here proceeds by reduction to linear programming feasibility, closely following the proof given in [PH07] that the finite satisfiability of $\mathcal{GC}^2$ is in ExpTime. We show how to modify that proof so as to accommodate the presence of a database.

2. Preliminaries

2.1. **The logics $\mathcal{GC}^2$, $\mathcal{GC}^2\mathcal{D}$ and $\mathcal{GC}^2\mathcal{DK}$.** A *literal* is an atomic formula or the negation of an atomic formula; a literal is *ground* if all arguments occurring in it are constants. If φ is any formula, the *length* of φ, denoted $\|\varphi\|$, is the number of symbols it contains. In addition to the usual boolean connectives, quantifiers $\forall$, $\exists$, and equality predicate, we employ the *counting quantifiers* $\exists_{\leq C}$, $\exists_{\geq C}$ and $\exists_{=C}$ (for all $C \geq 0$). A *sentence* is a formula with no free variables. In most of this paper, we consider *two-variable* formulas—that is, those whose only variables (free or bound) are x and y. Furthermore, we restrict attention to signatures comprised of individual constants or predicates of arity 1 or 2; in particular, function symbols are not allowed. (Function symbols were used in Section 1 for the sake of presentation, but we introduce different notation for path-functional dependencies in the sequel.) We call literals involving only symbols from some signature σ σ-*literals*.

An atomic formula of the form $p(x, y)$ or $p(y, x)$ is called a *guard*. The guarded two-variable fragment with counting, denoted $\mathcal{GC}^2$ is the smallest set of formulas satisfying the following conditions:

- unary or binary literals with all arguments in $\{x, y\}$ are formulas, with $=$ allowed as a binary predicate;
- the set of formulas is closed under Boolean combinations;
- if φ is a $\mathcal{GC}^2$-formula with at most one free variable and u is a variable (i.e. either x or y), then $\forall u.\varphi$ and $\exists u.\varphi$ are formulas.
- if φ is a formula, α is a guard, u is a variable and C is a bit-string, then $\exists_{\leq C} u(\alpha \wedge \varphi)$, $\exists_{\geq C} u(\alpha \wedge \varphi)$ and $\exists_{=C} u(\alpha \wedge \varphi)$ are formulas.

We read $\exists_{\leq C} u.\varphi$ as 'there exist at least C u such that φ', and similarly for $\exists_{\geq C}$ and $\exists_{=C}$. The formal semantics is as expected. To improve readability, we write $\exists_{\leq 0} u(\alpha \wedge \neg\varphi)$ as $\forall u(\alpha \rightarrow \varphi)$, and $\exists_{\geq 1} u(\alpha \wedge \varphi)$ as $\exists u(\alpha \wedge \varphi)$. Where no confusion results, we equivocate between bit-strings and the integers they represent, it being understood that the size of the expression $\exists_{=C}$ is approximately $\log C$ (and not C). Observe that formulas of the form $\exists_{\leq C} x.p(x)$ are not in $\mathcal{GC}^2$, because counting quantifiers must be guarded by atoms featuring both variables. (This is not a gratuitous restriction: adding such formulas to $\mathcal{GC}^2$ renders the fragment NExpTime-time hard.) In the sequel, formulas that are obviously logically equivalent to a $\mathcal{GC}^2$-formula will typically be counted as $\mathcal{GC}^2$-formulas by courtesy.

A *database* is a set Δ of ground (function-free) literals. We call Δ *consistent* if it includes no pair of ground literals $\{\alpha, \neg\alpha\}$. Where the signature σ is clear from context, we call Δ *complete* if for any ground σ-literal $\lambda \notin \Delta$, $\Delta \cup \{\lambda\}$ is inconsistent. Given a database Δ, a *completion* for Δ is any complete set $\Delta^* \supseteq \Delta$ of ground (function-free) σ-literals. It is obvious that every consistent database has a consistent completion. Databases are

interpreted as sets of atomic formulas in the expected way. For convenience, we shall employ the unique names assumption throughout: distinct individual constants are interpreted as distinct individuals. Hence, when a particular interpretation is clear from context, we sometimes equivocate between constants and their denotations to streamline the presentation. To further reduce notational clutter, we allow ourselves to treat a database Δ as a single conjunctive formula—i.e., writing Δ in place of $\bigwedge \Delta$.

The *two-variable guarded fragment with counting and databases*, denoted $\mathcal{GC}^2\mathcal{D}$, is defined exactly as for $\mathcal{GC}^2$, except that we have the additional syntax rule

- ground unary or binary literals are formulas.

Although $\mathcal{GC}^2\mathcal{D}$ allows formulas in which ground literals appear within the scope of quantifiers, in practice, it is much more natural to separate out the '$\mathcal{GC}^2$-part' from the 'database-part': thus all $\mathcal{GC}^2\mathcal{D}$-formulas encountered in the sequel will have the form $\varphi \wedge \Delta$, where φ is a $\mathcal{GC}^2$-formula, and Δ a database. Notice, however, that, in $\mathcal{GC}^2\mathcal{D}$, it is forbidden to mix variables and individual constants in atoms: thus, for example, $p(x, c)$ is not a $\mathcal{GC}^2\mathcal{D}$-formula. This is an essential restriction; if mixing variables and constants is allowed, it is easy to encode a grid of exponential size and, as a result, runs of an exponential-time Turing machine, leading to an NExpTime lower bound. If $\mathfrak{A}$ is any structure, we call those elements interpreting an individual constant *active*, and we refer to the set of all such elements as the *active domain* (or sometimes, informally, as *the database*). Elements which are not active are called *passive*.

We assume that any signature σ features a (possibly empty) distinguished subset of binary predicates, which we refer to as *key predicates*. Key predicates are always interpreted as the graphs of irreflexive functions. That is, if f is a key predicate, then in any structure $\mathfrak{A}$ interpreting f, $\mathfrak{A} \models \forall x \exists_{\leq 1} y\, f(x,y)$ and $\mathfrak{A} \models \forall x \exists y (f(x,y) \wedge x \neq y)$. (We remark that this formula is in $\mathcal{GC}^2$.) We use the (possibly subscripted or otherwise decorated) letters f, g, h to range over key predicates, and we use $\bar{f}$, $\bar{g}$, $\bar{h}$ for words over the alphabet of key predicates—i.e., finite sequences of key predicates. Warning: $\bar{f}_1$, $\bar{f}_2$ etc. will always be taken to denote whole sequences of key predicates, not the individual elements of some such sequence $\bar{f}$.

If $\mathfrak{A}$ is a structure, $\bar{f}$ a word $f_0 \cdots f_{k-1}$ over the alphabet of key predicates interpreted by $\mathfrak{A}$, and $a \in A$, we write $\bar{f}^{\mathfrak{A}}(a)$ to denote the result of successively applying the corresponding functions in $\bar{f}$ to a. More formally, $\bar{f}^{\mathfrak{A}}(a) = a_k$ where $a_0 = a$ and, for all i $(0 \leq i < k)$, a_{i+1} is the unique $b \in A$ such that $\mathfrak{A} \models f_i(a_i, b)$. An n-ary *path-functional dependency* κ is an expression

$$\Psi[\bar{f}_1, \ldots, \bar{f}_n] \tag{2.1}$$

where, for all i $(1 \leq i \leq n)$, $\bar{f}_i$ is a word over the alphabet of key predicates. Thus, $\Psi[\ ]$ is simply a piece of logical syntax that allows us to construct a formula from an n-tuple of sequences of key predicates. For any interpretation $\mathfrak{A}$, we take the dependency (2.1) to be *satisfied* in $\mathfrak{A}$, and write $\mathfrak{A} \models \kappa$, if, for all $a, b \in A$,

$$\bar{f}_i^{\mathfrak{A}}(a) = \bar{f}_i^{\mathfrak{A}}(b) \text{ for all } i \ (1 \leq i \leq n) \text{ implies } a = b.$$

That is to say, κ is satisfied in $\mathfrak{A}$ if any two elements of A which agree on the result of applying each of the function sequences corresponding to the words $\bar{f}_1$ to $\bar{f}_n$ are in fact identical. In this paper we shall consider only unary and binary path-functional dependencies, i.e. $n = 1$ or 2. The *two-variable guarded fragment with counting, databases and (unary or binary) path-functional dependencies*, denoted $\mathcal{GC}^2\mathcal{DK}$, is defined exactly as for $\mathcal{GC}^2\mathcal{D}$, except that we have the additional syntax rule

- all unary and binary path-functional dependencies are formulas.

Again, to reduce notational clutter, we treat a set of path-functional dependencies K as a single conjunctive formula—i.e., writing K in place of $\bigwedge K$. Although $\mathcal{GC}^2\mathcal{DK}$ allows formulas in which path-functional dependencies appear within the scope of quantifiers, in practice, it is much more natural to separate out the '$\mathcal{GC}^2$-part' from the database and the path-functional dependencies: thus all $\mathcal{GC}^2\mathcal{DK}$-formulas encountered in the sequel will have the form $\varphi \wedge \Delta \wedge K$, where φ is a sentence of $\mathcal{GC}^2$, Δ is database, and K a set of path-functional dependencies.

Note that, although key predicates give us the ability to *express* functional dependencies, they remain, syntactically speaking, predicates, not function-symbols. In fact in the logics considered here, there are no function-symbols. In particular the expressions (1.1) and (1.2) are for expository purposes only: in $\mathcal{GC}^2\mathcal{DK}$, we should write, respectively, the unary and binary path-functional dependencies

$$\text{Ⴂ[custID],} \qquad \text{Ⴂ[name, postCode address],}$$

where custID, postCode and address are binary (key) predicates.

Taking key predicates to denote total (as opposed to partial) functions represents no essential restriction, as partial functions can always be encoded using total functions interpreted over expanded domains featuring 'dummy' objects. The restriction to *irreflexive* functions, though not so easily eliminable, is in most cases perfectly natural (the telephone number of a person is not a person), and greatly simplifies much of the ensuing argumentation, obviating the constant need to consider various special cases.

If $\mathcal{L}$ is any of the languages $\mathcal{GC}^2$, $\mathcal{GC}^2\mathcal{D}$ or $\mathcal{GC}^2\mathcal{DK}$, a sentence φ of $\mathcal{L}$ is (*finitely*) *satisfiable* if, for some (finite) structure $\mathfrak{A}$ interpreting its signature, $\mathfrak{A} \models \varphi$. We define the (*finite*) *satisfiability* problem for any of the logics in the expected way: given a formula φ of $\mathcal{L}$, return Y if φ is (finitely) satisfiable; N otherwise. It was shown in [Kaz04] that the satisfiability problem for $\mathcal{GC}^2$ is ExpTime-complete, and in [PH07] that the finite satisfiability problem for $\mathcal{GC}^2$ is ExpTime-complete. In this paper, we show that the same complexity bounds apply to $\mathcal{GC}^2\mathcal{D}$ and $\mathcal{GC}^2\mathcal{DK}$.

The following lemma assures us that we may confine attention to $\mathcal{GC}^2\mathcal{DK}$ formulas of the standard form $\varphi \wedge \Delta \wedge K$.

Lemma 2.1. *Let ψ be a $\mathcal{GC}^2\mathcal{DK}$ formula. We can compute, in time bounded by exponential function of $|\psi|$, a set Ψ of $\mathcal{GC}^2\mathcal{DK}$ formulas with the following properties: (i) each formula in Ψ is bounded in size by a polynomial function of $|\psi|$ and has the form $\varphi \wedge \Delta \wedge K$, where φ is a $\mathcal{GC}^2$-formula, Δ a complete database and K a set of path-functional dependencies; (ii) φ is (finitely) satisfiable if and only if some member of Ψ is.*

Sketch proof. Consider any mapping θ from the set of path-functional dependencies and ground atoms occurring in ψ to the logical constants $\{\top, \bot\}$, and let $K = \Delta = \emptyset$. For every path-functional dependency κ occurring in ψ, if $\theta(\kappa) = \top$, add κ to K; otherwise, add to Δ a sequence of literals (possibly with fresh individual constants) encoding a violation of κ in the obvious way. Let ψ' be the result of replacing each path-functional dependency κ in ψ by $\theta(\kappa)$. For every ground atom α occurring in ψ', if $\theta(\alpha) = \top$, add α to Δ; otherwise, add $\neg\alpha$ to Δ. Let φ be the result of replacing each ground atom α in ψ' by $\theta(\alpha)$. Let Ψ be the set of of all formulas obtained in this way, for all possible mappings θ. $\square$

The logic $\mathcal{GC}^2$ sometimes surprises us with its expressive power. The next lemma provides a simple example, that will prove useful in the sequel. Let $\mathrm{dst}(x, y, z)$ be an abbreviation for the formula $x \neq y \wedge y \neq z \wedge x \neq z$, stating that x, y and z are distinct.

Lemma 2.2. *Let*

$$\varphi_1(x) := \exists y \exists z \big(\mathrm{dst}(x, y, z) \wedge \alpha(x, y) \wedge \beta(y, z) \big),$$
$$\varphi_2(y) := \exists x \exists z \big(\mathrm{dst}(x, y, z) \wedge \alpha(x, y) \wedge \beta(y, z) \big),$$

be formulas over σ, where $\alpha(x, y)$ and $\beta(y, z)$ are $\mathcal{GC}^2$-formulas. Then, we can compute, in polynomial time, $\mathcal{GC}^2$-formulas $\varphi_1^*(x)$ and $\varphi_2^*(y)$, which are logically equivalent to $\varphi_1(x)$ and $\varphi_2(y)$ respectively.

Proof. Let

$$\varphi_1^*(x) := \exists y \big(x \neq y \wedge \alpha(x, y) \wedge \neg\beta(y, x) \wedge \exists x(x \neq y \wedge \beta(y, x)) \big)$$
$$\vee \exists y \big(x \neq y \wedge \alpha(x, y) \wedge \beta(y, x) \wedge \exists_{\geq 2} x(x \neq y \wedge \beta(y, x)) \big);$$

$$\varphi_2^*(y) := \exists x \big(x \neq y \wedge \alpha(x, y) \wedge \neg\beta(y, x) \wedge \exists x(x \neq y \wedge \beta(y, x)) \big)$$
$$\vee \exists x \big(x \neq y \wedge \alpha(x, y) \wedge \beta(y, x) \wedge \exists_{\geq 2} x(x \neq y \wedge \beta(y, x)) \big). \qquad \square$$

2.2. **Graphs of structures.** If f is a key predicate, we introduce a new binary predicate f^{-1}, referred to as the *converse* of f, and subject to the requirement that, for any structure $\mathfrak{A}$, $\mathfrak{A} \models \forall x \forall y \big(f(x, y) \leftrightarrow f^{-1}(y, x) \big)$. (We remark that this formula is in $\mathcal{GC}^2$.) There is no requirement that f^{-1} be functional, though of course it must be irreflexive. We refer to any key predicate f or its converse f^{-1} as a *graph predicate*. We use the (possibly subscripted or otherwise decorated) letters r, s, t to range over graph predicates. If $r = f^{-1}$, we take r^{-1} to denote f. In the sequel, we fix a signature σ such that, for any key predicate f in σ, f^{-1} is also in σ. That is: the set of graph predicates of σ is closed under converse. We use letters $\bar{r}$, $\bar{s}$, $\bar{t}$ for words over the alphabet of graph predicates—i.e., finite sequences of graph predicates. If $\bar{r} = r_1 \cdots r_\ell$, we write $\bar{r}^{-1}$ for the word $r_\ell^{-1} \cdots r_1^{-1}$.

Let $\mathfrak{A}$ be a structure interpreting σ over domain A, and let $E = E_1 \cup E_2$ be the set of unordered pairs of elements of A, given by

$$E_1 = \{(a, b) \in A^2 \mid \mathfrak{A} \models r(a, b) \text{ for some graph predicate } r \text{ of } \sigma\}$$
$$E_2 = \{(a, b) \in A^2 \mid a \text{ and } b \text{ are distinct and both active}\}.$$

Then $G = (A, E)$ is a (possibly infinite) graph, and we refer to G as the *graph of $\mathfrak{A}$*. The edges of the graph are essentially the union of the interpretations of graph predicates, but with the database totally connected. By a *cycle* in $\mathfrak{A}$, we mean a finite sequence of distinct elements $\bar{a} = a_0 \cdots a_{\ell-1}$, with $\ell \geq 3$, such that, writing $a_\ell = a_0$, $(a_i, a_{i+1}) \in E$ for all i $(0 \leq i < \ell)$. The *length* of $\bar{a}$ is ℓ. A cycle in $\mathfrak{A}$ is said to be *active* if all its elements are active (i.e. are interpretations of constants), *passive* if all its elements are passive, and *mixed* if it contains both passive and active elements. We call $\mathfrak{A}$ ℓ-*quasi-acyclic* if all passive or mixed cycles in $\mathfrak{A}$ have length greater than ℓ.

The following lemma is a kind of pumping lemma for interpretations for the logic $\mathcal{GC}^2\mathcal{D}$. It states that, if a $\mathcal{GC}^2\mathcal{D}$-formula φ is satisfied in a (finite) structure $\mathfrak{A}$, then φ is also satisfied in a (finite) structure $\mathfrak{B}$ containing no short passive or mixed cycles in $\mathfrak{B}$.

Lemma 2.3 [PH09, Lemma 13]. *Let φ be a $\mathcal{GC}^2\mathcal{D}$-formula. Suppose $\mathfrak{A} \models \varphi$, with $D \subseteq A$ the set of active elements, and let $\ell > 0$. Then there exists an ℓ-quasi-acyclic model $\mathfrak{B} \models \varphi$ such that $D \subseteq B$ and $\mathfrak{A}|_D = \mathfrak{B}|_D$. Moreover, if $\mathfrak{A}$ is finite, then we can ensure that $\mathfrak{B}$ is finite.*

We remark that the proof of Lemma 2.3 makes essential use of guardedness.

Having discussed the graphs defined by $\mathcal{GC}^2\mathcal{D}$-structures, we turn now to configurations of small collections of elements of those structures. Let $\mathfrak{A}$ be a structure interpreting σ over domain A, let $\bar{a} = a_0 \cdots a_\ell$ ($\ell \geq 0$) be a word over the alphabet A, and let $\bar{r} = r_0, \dots r_{\ell-1}$ be a word over the alphabet of graph predicates of σ. The pair $[\bar{a}, \bar{r}]$ is a *walk* of *length* ℓ if, for all i ($0 \leq i < \ell$) $\mathfrak{A} \models r_i[a_i, a_{i+1}]$. Alternatively, we say that $\bar{a}$ is an $\bar{r}$-*walk*; and where $\bar{r}$ is clear from context, we speak of *the walk* $\bar{a}$. There is no requirement that the a_i all be distinct, though, of course, the irreflexivity of r_i means that $a_i \neq a_{i+1}$ ($0 \leq i < \ell$); however, if the a_i *are* all distinct, then we speak of the *path* $\bar{a}$. Now let $\bar{a}' = a_0, \dots, a_{\ell-1}$, with $\bar{r} = r_0, \dots r_{\ell-1}$ be as before. The pair $(\bar{a}', \bar{r})$ is a *tour* of *length* ℓ if, for all i ($0 \leq i < \ell$) $\mathfrak{A} \models r_i[a_i, a_{i+1}]$ where addition in indices is performed modulo ℓ. Again, we speak of $\bar{a}'$ as being an $\bar{r}$-*tour*, etc. Intuitively, we identify the tour $(\bar{a}', \bar{r})$ with the walk $[\bar{a}'a_0, \bar{r}]$, starting and ending at a_0. Note that (ϵ, ϵ) (corresponding to $\ell = 0$) is a tour, namely, the empty tour starting (and ending) at any element a. A path $\bar{a}$ is said to be *active* if all its elements are active, *passive* if all its elements are passive, and *mixed* otherwise; similarly for (non-empty) tours. In the case of the empty tour, we count it as being active if we think of it as beginning at an active element, and passive otherwise: this slight informality should cause no confusion in practice. Notice that a tour of length at least 3 in which all elements are distinct is a cycle in $\mathfrak{A}$.

Consider now any walk $[\bar{a}, \bar{f}]$ in $\mathfrak{A}$, where $\bar{a} = a_0 \cdots a_\ell$. Noting that the a_i are not necessarily distinct, let $V = \{a_i \mid 0 \leq i \leq \ell\}$, and let us say that a and b in V are *neighbours* if, for some i ($0 \leq i < \ell$), either $a = a_i$ and $b = a_{i+1}$, or $b = a_i$ and $a = a_{i+1}$. Let E be the set of unordered pairs (a, b) from V such that a and b are neighbours. Then $G = (V, E)$ is a graph, which we refer to as the *locus* of the walk $[\bar{a}, \bar{f}]$. If $(\bar{a}, \bar{f})$ is a non-empty tour in $\mathfrak{A}$ beginning at a_0, then we take the locus of $(\bar{a}, \bar{f})$ to be the locus of the walk $[\bar{a}a_0, \bar{f}]$. Thus loci are static records of all the steps taken during a walk or tour, but with information about the order of those steps suppressed. Fig. 1 shows a possible locus of a tour $(a_0a_1a_2a_3a_4a_3a_5a_3a_4a_3a_2a_1, f_0 \cdots f_{11})$ of length 12; the element a_3 is encountered four times in this tour. It is easy to see that the locus of a walk (or tour) in $\mathfrak{A}$ is a subgraph of the graph of $\mathfrak{A}$, though of course it will in general not be an induced subgraph. We call a walk (or tour) *acyclic* if its locus contains no cycles. The tour whose locus is depicted in Fig. 1 is acyclic. It should be obvious that, if $\mathfrak{A}$ is an ℓ-quasi-acyclic structure, and $(\bar{a}, \bar{f})$ a passive tour in $\mathfrak{A}$ of length at most ℓ, then $(\bar{a}, \bar{f})$ must be acyclic.

2.3. **Path-functional dependencies and their violations.** With these preliminaries behind us, we turn our attention to the analysis of path-functional dependencies in particular structures. We start, for simplicity, with the unary case. Our goal will be to encode a given unary path-functional dependency as a certain $\mathcal{GC}^2$ formula. Let κ be a unary path-functional dependency $\mathcal{P}[\bar{f}]$, where $\bar{f} = f_0 \cdots f_{k-1}$. If $0 \leq i \leq k$, denote the prefix $f_0 \cdots f_{i-1}$ by $\bar{f}_i$, and let $\kappa_i = \mathcal{P}[\bar{f}_i]$; we call κ_i a *prefix* of κ. It is easily seen that κ entails each of its prefixes—i.e., $\mathfrak{A} \models \kappa \to \kappa_i$. Moreover, the empty unary path-functional dependency, $\mathcal{P}[\epsilon]$, is trivially valid. A set K of unary path-functional dependencies is *prefix-closed*, if for any $\kappa \in K$, every prefix of κ is in K. Given any set K of unary path-functional dependencies,
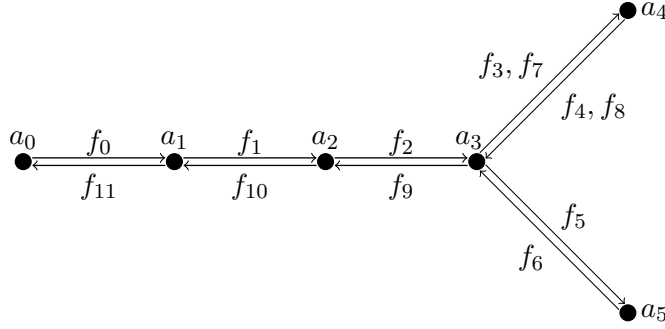
FIGURE 1. A tour $(a_0a_1a_2a_3a_4a_3a_5a_3a_4a_3a_2a_1, f_0\cdots f_{11})$ and its locus.

we may—without affecting satisfiability—ensure that it is prefix closed by adding at most linearly many additional unary path-functional dependencies.

Suppose now that κ is violated in some σ-structure $\mathfrak{A}$, and let $a,b \in A$ be a violating pair for κ, i.e. $a \neq b$ but the $\bar{f}$-walks starting at a and b end in the same element. Define the sequence of elements $\bar{a} = a_0\cdots a_k$ where $a_0 = a$ and, for all i ($0 \leq i < k$), $\mathfrak{A} \models f_i(a_i, a_{i+1})$; and define $\bar{b} = b_0\cdots b_k$ where $b_0 = b$ and, for all i ($0 \leq i < k$), $\mathfrak{A} \models f_i(b_i, b_{i+1})$. Noting that $a_k = b_k$, let i be the smallest number $0 \leq i \leq k$ such that $a_i = b_i$, i.e. the point at which $\bar{a}$ and $\bar{b}$ converge. Clearly $i > 0$, since a and b are distinct. We say that the violation of κ at a,b is *critical* if $i = k$, i.e. if $\bar{a}$ and $\bar{b}$ converge only at their last element. It is then obvious that, if K is prefix closed, and some $\kappa \in K$ is violated, then some $\kappa' \in K$ is critically violated.

Checking for critical violations of unary path-functional dependencies is simple. Bearing in mind that $\mathsf{q}[\epsilon]$ is trivial valid, suppose $\kappa = \mathsf{q}[\bar{f}f]$. For any prefix $\bar{f}'$ of $\bar{f}$ (proper or improper), let $p_{\bar{f}'}$ be a fresh unary predicate, and let P_κ be the set $\mathcal{GC}^2$-formulas

$$\left\{\forall x.p_\epsilon(x)\right\} \cup \left\{\forall x\big(p_{\bar{f}'f'}(x) \leftrightarrow \exists y(p_{\bar{f}'}(y) \wedge f'(y,x))\big) \mid \bar{f}'f' \text{ a prefix of } \bar{f}\right\}.$$

In the presence of P_κ, we may read $p_{\bar{f}'}(x)$ as stating that x is at the end of an $\bar{f}'$-walk in $\mathfrak{A}$. Thus, in a model $\mathfrak{A} \models P_\kappa$, $\kappa = \mathsf{q}[\bar{f}f]$ has a critical violation just in case the formula

$$\psi_\kappa(y) := \exists x\exists z(\mathrm{dst}(x,y,z) \wedge (p_{\bar{f}}(x) \wedge f(x,y)) \wedge (p_{\bar{f}}(z) \wedge f(z,y)).$$

is satisfied in $\mathfrak{A}$. (In particular, if κ has a critical violation at a, b, ψ_κ will be satisfied at the (common) final elements of the $\bar{f}$-walks starting at a and b.) By Lemma 2.2, $\psi_\kappa(y)$ may be equivalently written as a $\mathcal{GC}^2$-formula $\psi_\kappa^*(y)$. Thus, if K is a prefix-closed set of unary path-functional dependencies, a $\mathcal{GC}^2\mathcal{DK}$-formula $\varphi \wedge K \wedge \Delta$ is satisfiable over some domain A if and only if the $\mathcal{GC}^2\mathcal{D}$-formula

$$\varphi \wedge \bigwedge\bigcup\{P_\kappa \mid \kappa \in K\} \wedge \bigwedge_{\kappa \in K} \forall y \neg\psi_\kappa^*(y) \wedge \Delta$$

is satisfiable over the same domain. This provides a polynomial time reduction from the (finite) satisfiability problem for $\mathcal{GC}^2\mathcal{DK}$ restricted to *unary* path-functional dependencies to the corresponding problem for $\mathcal{GC}^2\mathcal{D}$. Since we show in the sequel that the latter problem is in ExpTime, so is the former.

We now turn our attention to the more difficult case of binary path-functional dependencies. Let κ be a binary path-functional dependency $\mathsf{q}[\bar{f}, \bar{g}]$, where $\bar{f} = f_0\cdots f_{k-1}$. For

all i ($0 \leq i < k$), denote by $\bar{f}_i$, the prefix $f_1 \cdots f_{i-1}$ of $\bar{f}$, and let $\kappa_i = \Psi[\bar{f}_i, \bar{g}]$. We call κ_i a *left-prefix* of κ. Thus κ entails each of its left-prefixes; moreover, $\Psi[\epsilon, \bar{g}]$ is trivially valid. If K is a set of binary path-functional dependencies, say that K is *left-prefix-closed*, if for $\kappa \in K$, any left-prefix of κ is in K. Any set K of binary path-functional dependencies may—without affecting satisfiability—be made left-prefix-closed by adding at most linearly many additional binary path-functional dependencies. We could instead have spoken of right-prefix-closed sets of functional dependencies, defined in the obvious way; the choice between these notions is completely arbitrary.

Suppose now that $\kappa = \Psi[\bar{f}, \bar{g}]$ is violated in some σ-structure $\mathfrak{A}$, and let $a, b \in A$ be a violating pair for $\Psi[\bar{f}, \bar{g}]$, i.e. $a \neq b$ but the $\bar{f}$-walks starting at a and b end in the same element, and moreover the $\bar{g}$-walks starting at a and b end in the same element. Writing $\bar{f} = f_0 \cdots f_{k-1}$, define the sequence of elements $\bar{a} = a_0 \cdots a_k$ where $a_0 = a$ and, for all i ($0 \leq i < k$), $\mathfrak{A} \models f_i(a_i, a_{i+1})$. Similarly define $\bar{b} = b_0 \cdots b_k$ where $b_0 = b$ and, for all i ($0 \leq i < k$), $\mathfrak{A} \models f_i(b_i, b_{i+1})$. We call the violation of κ at a, b *critical* if $a_i \neq b_i$ for all i ($0 \leq i < k$). It is again obvious that, if K is a left-prefix-closed set of binary path-functional dependencies, and some $\kappa \in K$ is violated, then some $\kappa' \in K$ is critically violated. Thus, as with unary path-functional dependencies, so with their binary counterparts, we may confine our attention to critical violations.

The difficulty is that critical violations of binary path-functional dependencies cannot be straightforwardly expressed using $\mathcal{GC}^2$-formulas as in the unary case. To understand the problem, consider a binary path-functional dependency $\kappa = \Psi[\bar{f}f, \bar{g}]$, which has a critical violation at a, b. Writing $\bar{f} = f_0 \cdots f_{k-1}$ and $f_k = f$, define $\bar{a} = a_0 \cdots a_{k+1}$ where $a_0 = a$ and, for all i ($0 \leq i \leq k$), $\mathfrak{A} \models f_i(a_i, a_{i+1})$, and define $\bar{b} = b_0 \cdots b_{k+1}$ where $b_0 = b$ and, for all i ($0 \leq i \leq k$), $\mathfrak{A} \models f_i(b_i, b_{i+1})$. Thus, $a_{k+1} = b_{k+1}$. Writing $\bar{g} = g_0 \cdots g_{\ell-1}$, define $\bar{a}' = a'_0 \cdots a'_\ell$ where $a'_0 = a$ and, for all i ($0 \leq i < \ell$), $\mathfrak{A} \models g_i(a'_i, a'_{i+1})$; and define $\bar{b}'$ similarly, but starting with $b'_0 = b$. Thus, $a'_\ell = b'_\ell$. Referring to Fig. 2, it follows that

$$a_0 a_1 \cdots a_k a_{k+1} b_k \cdots b_1 b_0 b'_1 \cdots b'_\ell a'_{\ell-1} \cdots a'_1$$

is an $\bar{f} f f^{-1} \bar{f}^{-1} \bar{g} \bar{g}^{-1}$-tour in $\mathfrak{A}$. The problem is how to characterize the existence of such tours with only $\mathcal{GC}^2$-formulas and conditions on the database at our disposal.

Let us consider the tour of Fig. 2 more closely. Since key predicates are by assumption irreflexive, we know that neighbouring elements in this tour are distinct. Furthermore, since the violation of κ is by assumption critical, we also know that $a_k \neq b_k$. That is, the elements a_k, a_{k+1} and b_k are all distinct. We can make this observation—which is fundamental to the entire development of Section 3, 4 and 5—work for us if we rotate the tour so that it starts at a_k. Write $c = a_k$, $d = a_{k+1} = b_{k+1}$ and $e = b_k$, so that $\mathfrak{A} \models f(c, d)$ and $\mathfrak{A} \models f(e, d)$, with c, d and e all distinct. This yields a tour

$$(cde\bar{e}, f f^{-1} \bar{f}^{-1} \bar{g} \bar{g}^{-1} \bar{f} f)$$

for some sequence of elements $\bar{e}$. For brevity, we shall always write $\bar{\mathbf{r}}_\kappa$ in the sequel to denote the 'rotated' sequence of graph predicates

$$\bar{\mathbf{r}}_\kappa = f f^{-1} \bar{f}^{-1} \bar{g} \bar{g}^{-1} \bar{f} f$$

obtained from the binary path-functional dependency $\Psi[\bar{f}f, \bar{g}]$. Thus, κ is critically violated in a structure $\mathfrak{A}$ if and only if $\mathfrak{A}$ contains an $\bar{\mathbf{r}}_\kappa$-tour whose first three elements are distinct.

Note that the diagram of Fig. 2 depicts an $\bar{\mathbf{r}}_\kappa$-tour, and not its locus. In particular, there is no assumption that the sequences $\bar{a}$, $\bar{b}$, $\bar{a}'$ and $\bar{b}'$ do not contain repeated elements, and no assumption that they are disjoint. Indeed, we recall Lemma 2.3, which allows us to
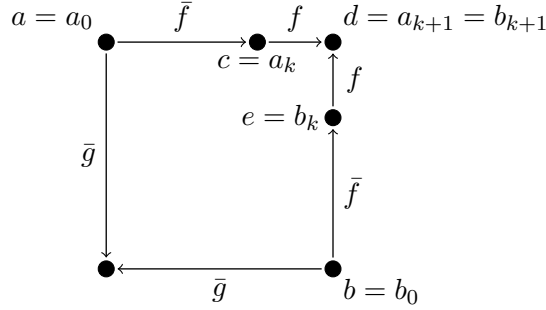
FIGURE 2. A critical violating tour of $\Psi[\bar{f}f, \bar{g}]$. The sequences $a_0 \cdots a_k$ and $b_0 \cdots b_k$ are $\bar{f}$-walks; the elements c, d and e are *distinct*.

restrict attention to ℓ-quasi-acyclic structures for various ℓ. If $\ell > |\bar{\mathbf{r}}_\kappa|$, then all violations of κ involving only passive elements will yield acyclic $\bar{\mathbf{r}}_\kappa$-tours. This forms our main line of attack: Sections 3 and 4 are concerned with the classification of $\bar{\mathbf{r}}_\kappa$-tours; Section 5 uses this classification to encode the non-existence of critical violations using $\mathcal{GC}^2$-formulas and conditions on the database; finally, Section 6 assembles all these observations to yield a reduction of the (finite) satisfiability problem for $\mathcal{GC}^2\mathcal{DK}$ to the corresponding problem for $\mathcal{GC}^2\mathcal{D}$.

3. Decompositions of Walks and Tours

We have seen that critical violations of binary path-functional dependencies in a structure correspond to the existence of certain tours in the graph of that structure. This section presents the basic tools we require in the sequel for decomposing walks and tours in structures. None of the reasoning involved goes beyond elementary graph theory. Lemmas 3.1–3.2 concern acyclic walks and tours; Lemmas 3.3–3.5 concern tours in ℓ-quasi-acyclic structures.

Lemma 3.1. *Let $\mathfrak{A}$ be a structure and $[\bar{a}, \bar{t}]$ be an acyclic walk in $\mathfrak{A}$. Then, for some $m \geq 1$, the sequences $\bar{a}$ and $\bar{t}$ can be decomposed*

$$\bar{a} = \bar{b}_0 b_0 \cdots \bar{b}_{m-1} b_{m-1} \bar{b}_m b_m$$
$$\bar{t} = \bar{r}_0 r_0 \cdots \bar{r}_{m-1} r_{m-1} \bar{r}_m$$

such that: (i) the sequences $\bar{b}_1, \ldots, \bar{b}_m$ are pairwise disjoint; (ii) for all j $(0 \leq j \leq m)$, $(\bar{b}_j, \bar{r}_j)$ is a tour starting at b_j; and (iii) $[b_0 \cdots b_k, r_0 \cdots r_{m-1}]$ is a path in $\mathfrak{A}$ (Fig. 3).

Proof. Let $\bar{a} = a_0 \cdots a_\ell$ $(\ell > 0)$. Let $\iota(0) = 0$ and define $b_0 = a_{\iota(0)} = a_0$. Suppose now that $\iota(i)$ and b_i have been defined, with $i \leq \ell$. If $b_i \neq a_\ell$, then let $\iota(i+1)$ be the largest number j $(\iota(i) < j \leq \ell)$ such that $a_{j-1} = b_i$, and define $b_{i+1} = a_{\iota(i+1)}$ and $\bar{b}_i = a_{\iota(i)} \cdots a_{\iota(i+1)-2}$. Likewise, define $r_i = t_{\iota(i+1)-1}$ and $\bar{r}_i = t_{\iota(i)} \cdots t_{\iota(i+1)-2}$. Then $\mathfrak{A} \models r_i(b_i, b_{i+1})$ and $(\bar{b}_i, \bar{r}_i)$ is a possibly empty acyclic tour starting at b_i. When, eventually, $b_i = a_\ell$, define $m = i$, $\bar{b}_m = a_{\iota(i)} \cdots a_{\ell-1}$ and $\bar{r}_m = f_{\iota(i)} \cdots f_{\ell-1}$. The disjointness of the $\bar{b}_i$ is immediate. $\square$
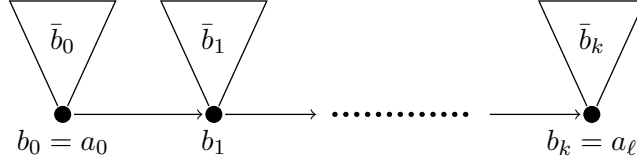
FIGURE 3. A decomposition of a walk $a_0 \cdots a_\ell$ into its spine $b_0 \cdots b_k$, with a (possibly empty) acyclic subtour at each b_i, $0 \leq i \leq k$.
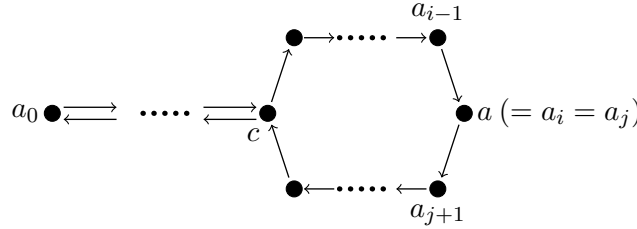

FIGURE 4. A cycle when $a_{i-1} \neq a_{j+1}$, as described in the proof of Lemma 3.2; $\bar{b}'''$ is the path from c to a_{i-1} and $\bar{b}''c$ is the path from a_{j+1} to c.

We call the path $[b_0 \cdots b_k, r_0 \cdots r_{k-1}]$ in the decomposition of $[\bar{a}, \bar{t}]$ given by Lemma 3.1 the *spine* of $[\bar{a}, \bar{t}]$. Where the sequence $\bar{t}$ is of no interest, we simply say that $b_0 \ldots b_k$ is the *spine* of $\bar{a}$.

Lemma 3.2. *Let $\mathfrak{A}$ be a structure and $\bar{a} = a_0, \ldots, a_{\ell-1}$ a non-empty acyclic tour in $\mathfrak{A}$. Let a be any element of $\bar{a}$ not equal to a_0, let i, j $(0 < i \leq j < \ell)$ be the smallest and the largest index respectively such that $a_i = a_j = a$, and let $a_\ell = a_0$. Then $a_{j+1} = a_{i-1}$.*

Proof. Observe first that the words $a_0, \ldots, a_{i-1}$ and $a_{j+1}, \ldots, a_\ell$ are non-empty, since $a_i = a_j \neq a_0 = a_\ell$; thus, these words define walks in $\mathfrak{A}$. Let $\bar{b}$ be the spine of the walk $a_0, \ldots, a_{i-1}$, and let $\bar{b}'$ be the spine of the walk $a_{j+1}, \ldots, a_\ell$. Thus, $\bar{b}$ is a path from a_0 to a_{i-1} and $\bar{b}'$ a path from a_{j+1} to a_ℓ. Let $\bar{b}''c$ be the shortest prefix of $\bar{b}'$ that intersects $\bar{b}$, and let $\bar{b}'''$ be the suffix of $\bar{b}$ beginning with the element c. Then, if $a_{i-1} \neq a_{j+1}$, $\bar{b}'''a_i\bar{b}''$ is a cycle, contrary to hypothesis. (See Fig. 4 for an illustration.) $\qquad\square$

The following lemmas concern tours in ℓ-quasi-acyclic structures. We start with the observation that, if a tour of length ℓ in such a structure exits the database at some point, then it must re-enter at the same point. Recall that a mixed tour is one featuring both active and passive elements; and remember that, in the graph of a structure $\mathfrak{A}$, the database is totally connected.

Lemma 3.3. *Let $\mathfrak{A}$ be an ℓ-quasi-acyclic structure and $\bar{a} = a_0 \cdots a_{\ell-1}$ a tour in $\mathfrak{A}$. Suppose i $(0 \leq i < \ell)$ is such that $a_0 \cdots a_i$ are all active, but a_{i+1} passive. Writing $a_\ell = a_0$, let j be the smallest index j $(i < j < \ell)$ such that a_{j+1} is active. Then $a_{i+1} = a_j$ and $a_{j+1} = a_i$.*

Proof. By construction, $a_{i+1} \cdots a_j$ is a passive walk from a_{i+1} to a_j. Let the spine of this walk be $\bar{b}$; note that the sequence $\bar{b}$ is non-empty, and its initial and final elements are, respectively, a_{i+1} and a_j. Suppose first that $a_i \neq a_{j+1}$. Since these elements are active, they are joined by an edge in the graph of $\mathfrak{A}$, whence $a_i\bar{b}a_{j+1}$ is a mixed cycle in $\mathfrak{A}$ of length (at least 3 and) at most ℓ. Since $\mathfrak{A}$ is ℓ-quasi-acyclic this is a contradiction, whence $a_i = a_{j+1}$.

Suppose, now that $a_{i+1} \neq a_j$, whence $|\bar{b}| \geq 2$. Then, since $a_i = a_{j+1}$, it follows that $a_i \bar{b}$ is a mixed cycle of length (at least 3 and) at most ℓ. Again, this is a contradiction, whence $a_{i+1} = a_j$. $\qquad \square$

The next lemma is a similar to the last, but with 'enter' and 'exit' transposed. The reasoning is essentially identical.

Lemma 3.4. *Let $\mathfrak{A}$ be an ℓ-quasi-acyclic structure and $\bar{a} = a_0 \cdots a_{\ell-1}$ a tour in $\mathfrak{A}$. Suppose i ($0 \leq i < \ell$) is such that $a_0 \cdots a_i$ are all passive, but a_{i+1} active. Writing $a_\ell = a_0$, let j be the largest index j ($i < j < \ell$) such that a_j is active and a_{j+1} is passive. Then $a_{i+1} = a_j$ and $a_{j+1} = a_i$.*

The final two lemmas of this section will play an important role in Sec. 6.

Lemma 3.5. *Let $\mathfrak{A}$ be an ℓ-quasi-acyclic structure, and $(\bar{a}, \bar{t})$ a mixed tour in $\mathfrak{A}$ of length ℓ, beginning with some passive element a. Then $(\bar{a}, \bar{t})$ can be decomposed as $(\bar{c}\bar{a}^*\bar{b}, \bar{s}\bar{t}^*\bar{r})$ such that, for some active element b: (i) $[\bar{b}a; \bar{r}]$ is an acyclic walk from b to a in which b occurs exactly once; (ii) $[\bar{c}b; \bar{s}]$ is an acyclic walk from a to b in which b occurs exactly once; and (iii) $(\bar{a}^*; \bar{t}^*)$ is a tour beginning at b.*

Proof. Write $\bar{a} = a_0 \cdots a_{\ell-1}$, let $b = a_i$ be the first active element of $\bar{a}$, and let j ($i \leq j < \ell$) be the greatest index such that $a_j = b$. Since a is passive, $i > 0$. Write $\bar{c} = a_0 \cdots a_{i-1}$, $\bar{a}^* = a_i \cdots a_{j-1}$ and $\bar{b} = a_j \cdots a_{\ell-1}$. Let $\bar{s}$, $\bar{t}^*$ and $\bar{r}$ be the corresponding decomposition of $\bar{t}$. Since $(\bar{a}, \bar{t})$ is a mixed tour, it is acyclic; so therefore are the walks $[\bar{b}a; \bar{r}]$ and $[\bar{c}b; \bar{s}]$ which it contains. $\qquad \square$

Lemma 3.6. *Suppose $\mathfrak{A}$ is an ℓ-quasi-acyclic structure containing a tour, $(\bar{a}^*; \bar{t}^*)$, with $|t^*| = \ell$, and suppose the initial element of $\bar{a}^*$ is active. Then there exist decompositions $\bar{a}^* = \bar{b}_0 b_0 \cdots \bar{b}_{m-1} b_{m-1} \bar{b}_m$ and $\bar{t}^* = \bar{r}_0 r_0 \cdots \bar{r}_{m-1} r_{m-1} \bar{r}_m$, such that: (i) $(b_0 \cdots b_{m-1}, r_0 \cdots r_{m-1})$ is an active tour; (ii) each $(\bar{b}_i; \bar{r}_i)$ is an acyclic tour beginning at b_i ($1 \leq i < m$), and $(\bar{b}_m, \bar{r}_m)$ is an acyclic tour beginning at b_0 (Fig. 5).*

Proof. Write $\bar{a}^* = a_0, \ldots, a_{\ell-1}$, with a_0 active. As usual, let $a_\ell = a_0$. Let $\iota(0) = 0$ and define $b_0 = a_{\iota(0)} = a_0$. Suppose that $\iota(i)$ and b_i have been defined, with b_i active and $\iota(i) \leq \ell$. Let $\iota(i + 1)$ be the smallest number j ($\iota(i) < j \leq \ell$) such that a_j is active and distinct from b_i, assuming first that such an element exists. Note that, if $\iota(i + 1) \neq \iota(i) + 1$, then $a_{\iota(i)+1}$ is passive, whence, by Lemma 3.3, the next active element in the tour $\bar{a}$ after $a_{\iota(i)+1}$ is $a_{\iota(i)} = b_i$. By applying this argument repeatedly, we see that $a_{\iota(i+1)-1} = b_i$. Define $b_{i+1} = a_{\iota(i+1)}$ and $\bar{b}_i = a_{\iota(i)} \cdots a_{\iota(i+1)-2}$. Likewise, define $r_i = t_{\iota(i+1)-1}$ and $\bar{r}_i = t_{\iota(i)} \cdots t_{\iota(i+1)-2}$. Thus $(\bar{b}_i, \bar{r}_i)$ is a (possibly empty) mixed—hence acyclic—tour starting at b_i, and $\mathfrak{A} \models r_i(b_i, b_{i+1})$. Now suppose that there is no number j ($\iota(i) < j \leq \ell$) such that a_j is active and distinct from b_i. Set $m = \iota(i)$, and define $\bar{b}_m = a_{\iota(i)} \cdots a_{\ell-1}$. Likewise, define $\bar{r}_m = t_m \cdots t_{\ell-1}$. If $\bar{b}_m$ is non-empty, it leaves the database, and, whenever it does so, it re-enters at the same point, by Lemma 3.3. Hence $(\bar{b}_m, \bar{r}_m)$ is a (possibly empty) mixed—hence acyclic—tour starting at b_m whence $b_m = a_\ell = a_0$. $\qquad \square$

4. Detection of Violations

We now show how to detect violations of binary path-functional dependencies, using the decompositions of tours developed in the previous section. Recall from Section 2 that a
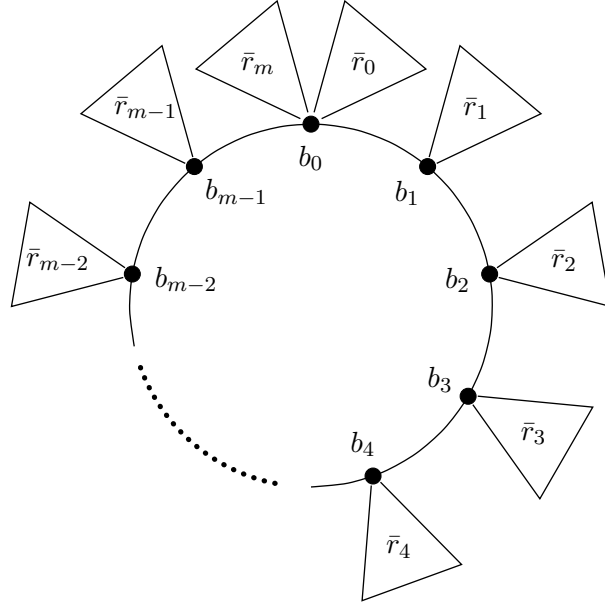
FIGURE 5. Decomposition of an $\bar{f}$-tour $(\bar{a}^*, \bar{t}^*)$ of length $\ell > 0$, beginning in the database, in an ℓ-quasi-acyclic structure (Lemma 3.6).

critical violation of a binary path-functional dependency κ in a structure $\mathfrak{A}$ is identified with an $\bar{\mathbf{r}}_\kappa$-tour whose first three elements are distinct. Such a tour can be decomposed in various ways, depending on which of these three elements belong to the active domain (i.e., the database). Altogether, we isolate nine configurations of $\bar{\mathbf{r}}_\kappa$-tours, illustrated in Fig. 6, with the circle representing a sub-tour in the database (which is further decomposed as in Fig. 5) and triangles representing acyclic sub-tours. The task of this section is to establish that the nine configurations of Fig. 6 exhaust the possible $\bar{\mathbf{r}}_\kappa$-tours. In Section 5, we show how to use $\mathcal{GC}^2$-formulas together with conditions on the database to rule out each of these configurations, and thus to guarantee that κ is not critically violated. In the following sequence of lemmas, we fix $\kappa = \P[\bar{f}f, \bar{g}]$, where $\bar{g} \neq \epsilon$, as usual writing $\bar{\mathbf{r}}_\kappa$ for $ff^{-1}\bar{f}^{-1}\bar{g}\bar{g}^{-1}\bar{f}$. Notice that, by construction, $|\bar{\mathbf{r}}_\kappa| \geq 4$.

Lemma 4.1. *Suppose $\mathfrak{A}$ is a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure containing a passive tour, $(\bar{a}, \bar{\mathbf{r}}_\kappa)$, whose first three elements, a_0, a_1 and a_2, are distinct. Then there exist decompositions $\bar{a} = a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{a}_2$ and $\bar{\mathbf{r}}_\kappa = ff^{-1}\bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$, such that (i) $(\bar{a}_0, \bar{t}_0)$ is an acyclic tour in $\mathfrak{A}$ starting at a_2; (ii) $(\bar{a}_1, \bar{t}_1)$ is an acyclic tour in $\mathfrak{A}$ starting at a_1; (iii) $(\bar{a}_2, \bar{t}_2)$ is an acyclic tour in $\mathfrak{A}$ starting at a_0; (iv) $t_2(a_1, a_0)$ and $t_1(a_2, a_1)$ are true in $\mathfrak{A}$ (Fig. 6(I)).*

Proof. Since $(\bar{a}, \bar{\mathbf{r}}_\kappa)$ is passive, it is acyclic. Write $\bar{a} = a_0 \cdots a_{\ell-1}$, $\bar{r} = r_0 \cdots r_{\ell-1}$ and $a_\ell = a_0$; thus, $r_0 = f$ and $r_1 = f^{-1}$. Let i be the largest index ($2 \leq i < \ell$) such that $a_i = a_2$, let $\bar{a}_0 = a_2 \cdots a_{i-1}$ and $\bar{t}_0 = r_2 \cdots r_{i-1}$. Thus, $(\bar{a}_0, \bar{t}_0)$ is an acyclic tour starting at a_2. By Lemma 3.2, $a_{i+1} = a_1$; write $t_1 = r_i$. Similarly, let j be the largest index ($i + 1 \leq j < \ell$) such that $a_j = a_1$, and let $\bar{a}_1 = a_{i+1} \cdots a_{j-1}$ and $\bar{t}_1 = r_{i+1} \cdots r_{j-1}$; thus, $(\bar{a}_1, \bar{t}_1)$ is an acyclic tour starting at a_1. By Lemma 3.2 again, $a_{j+1} = a_0$; write $t_2 = r_j$. Let $\bar{a}_2 = a_{j+1}, \ldots, a_{\ell-1}$ and $\bar{t}_2 = r_{j+1}, \ldots, r_{\ell-1}$. This completes the tour (going back to a_0) and $(\bar{a}_2, \bar{t}_2)$ is acyclic. $\square$
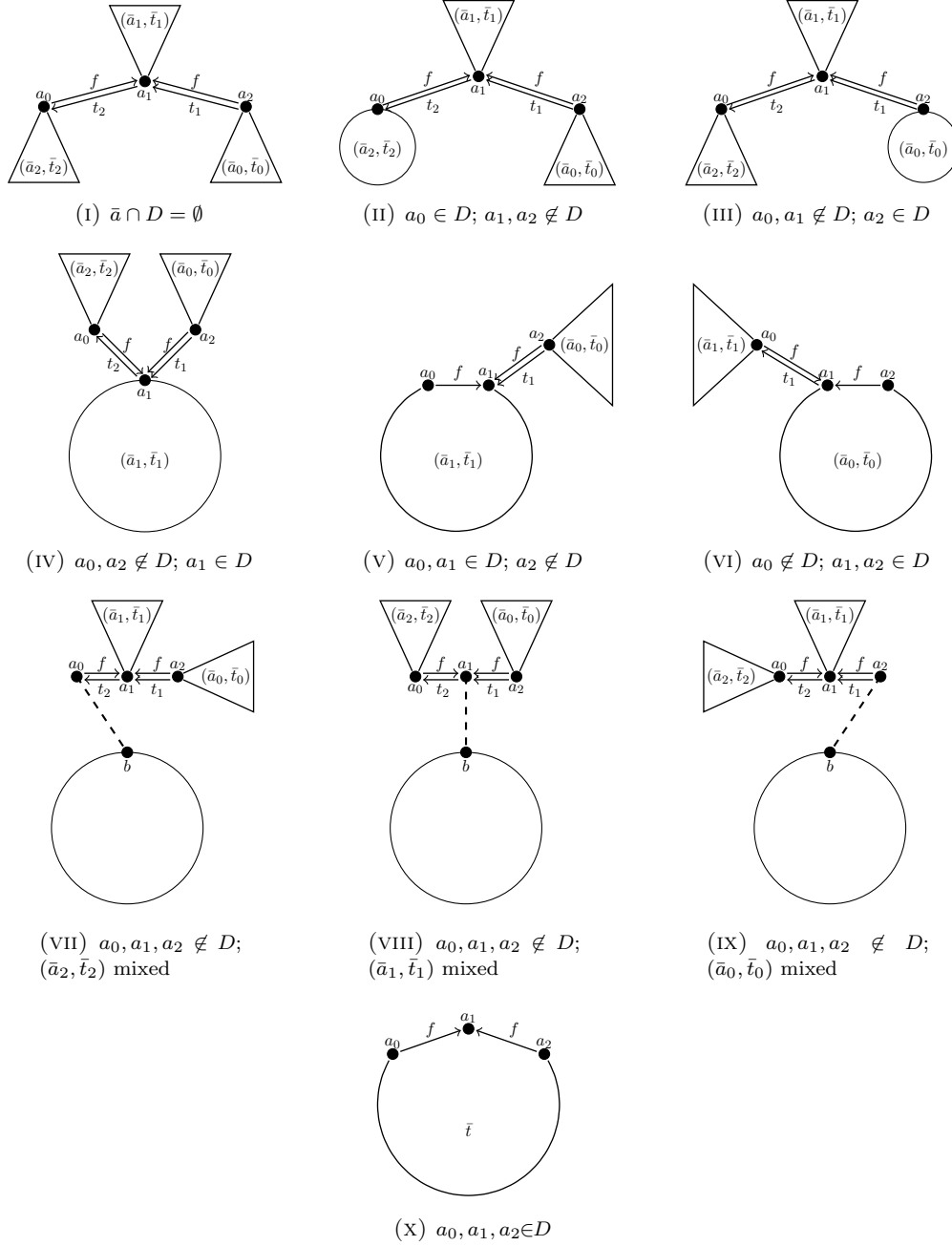
FIGURE 6. Possible configurations of a violating $\bar{\mathbf{r}}_\kappa$-tour $\bar{a}$ with initial (distinct) elements a_0, a_1 and a_2, in a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure with active domain D. Triangles indicate acyclic subtours; the circle indicates a subtour involving elements of D; see Lemmas 4.1–4.8.

Lemma 4.2. *Suppose $\mathfrak{A}$ is a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure containing a mixed tour, $(\bar{a}, \bar{\mathbf{r}}_\kappa)$, whose first three elements, a_0, a_1 and a_2, are distinct. Suppose further that a_0 is active, but a_1 and a_2 are passive. Then there exist decompositions $\bar{a} = a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{a}_2$ and $\bar{\mathbf{r}}_\kappa = ff^{-1}\bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$, such that (i) $(\bar{a}_0, \bar{t}_0)$ is an acyclic tour in $\mathfrak{A}$ starting at a_2; (ii) $(\bar{a}_1, \bar{t}_1)$ is an acyclic tour in $\mathfrak{A}$ starting at a_1; (iii) $(\bar{a}_2, \bar{t}_2)$ is a tour in $\mathfrak{A}$ starting at a_0; (iv) $t_2(a_1, a_0)$ and $t_1(a_2, a_1)$ are true in $\mathfrak{A}$ (Fig. 6(II)).*

Proof. Write $\bar{a} = a_0 \cdots a_{\ell-1}$, $\bar{r} = r_0 \cdots r_{\ell-1}$ and $a_\ell = a_0$; thus, $r_0 = f$ and $r_1 = f^{-1}$. Let j be the smallest index $(2 \le j < \ell)$ such that a_{j+1} is active. Since a_0 is active and a_1 passive, it follows by Lemma 3.3 that $a_j = a_1$ and $a_{j+1} = a_0$. Thus, $\bar{a}' = a_1 a_2 \cdots a_{j-1}$ is a passive—and hence acyclic—tour. Considering the tour $\bar{a}'$, let i be the largest index $(2 \le i < j)$ such that $a_i = a_2$. By Lemma 3.2, $a_{i+1} = a_1$. Let $\bar{a}_0 = a_2 \cdots a_{i-1}$ and $\bar{t}_0 = r_2 \cdots r_{i-1}$; let $\bar{a}_1 = a_{i+1} \cdots a_{j-1}$ and $\bar{t}_1 = r_{i+1} \cdots r_{j-1}$; and let $\bar{a}_2 = a_{j+1} \cdots a_{\ell-1}$ and $\bar{t}_2 = r_{j+1} \cdots r_{\ell-1}$. In addition, write $t_1 = r_i$ and $t_2 = r_j$. Thus, $\bar{a} = a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{a}_2$ and $\bar{\mathbf{r}}_\kappa = ff^{-1}\bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$. Moreover, $(\bar{a}_0, \bar{t}_0)$ and $(\bar{a}_1, \bar{t}_1)$ are acyclic tours, and $(\bar{a}_2, \bar{t}_2)$ is a tour. $\square$

Lemma 4.3. *Suppose $\mathfrak{A}$ is a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure containing a mixed tour, $(\bar{a}, \bar{\mathbf{r}}_\kappa)$, whose first three elements, a_0, a_1 and a_2, are distinct. Suppose further that a_2 is active, but a_0 and a_1 are passive. Then there exist decompositions $\bar{a} = a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{a}_2$ and $\bar{\mathbf{r}}_\kappa = ff^{-1}\bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$, such that (i) $(\bar{a}_0, \bar{t}_0)$ is a tour in $\mathfrak{A}$ starting at a_2; (ii) $(\bar{a}_1, \bar{t}_1)$ is an acyclic tour in $\mathfrak{A}$ starting at a_1; (iii) $(\bar{a}_2, \bar{t}_2)$ is an acyclic tour in $\mathfrak{A}$ starting at a_0; (iv) $t_2(a_1, a_0)$ and $t_1(a_2, a_1)$ are true in $\mathfrak{A}$ (Fig. 6(III)).*

Proof. Analogous to the proof of Lemma 4.2. $\square$

Lemma 4.4. *Suppose $\mathfrak{A}$ is a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure containing a mixed tour, $(\bar{a}, \bar{\mathbf{r}}_\kappa)$, whose first three elements, a_0, a_1 and a_2, are distinct. Suppose further that a_1 is active, but a_0 and a_2 are passive. Then there exist decompositions $\bar{a} = a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{a}_2$ and $\bar{\mathbf{r}}_\kappa = ff^{-1}\bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$, such that (i) $(\bar{a}_0, \bar{t}_0)$ is an acyclic tour in $\mathfrak{A}$ starting at a_2; (ii) $(\bar{a}_1, \bar{t}_1)$ is a tour in $\mathfrak{A}$ starting at a_1; (iii) $(\bar{a}_2, \bar{t}_2)$ is an acyclic tour in $\mathfrak{A}$ starting at a_0; (iv) $t_2(a_1, a_0)$ and $t_1(a_2, a_1)$ are true in $\mathfrak{A}$ (Fig. 6(IV)).*

Proof. Analogous to the proof of Lemma 4.2. $\square$

Lemma 4.5. *Suppose $\mathfrak{A}$ is a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure containing a mixed tour, $(\bar{a}, \bar{\mathbf{r}}_\kappa)$, whose first three elements, a_0, a_1 and a_2, are distinct. Suppose further that a_0 and a_1 are active, but a_2 is passive. Then there exist decompositions $\bar{a} = a_0 a_1 \bar{a}_0 a_2 \bar{a}_1$ and $\bar{\mathbf{r}}_\kappa = ff^{-1}\bar{t}_0 t_1 \bar{t}_1$, such that (i) $(\bar{a}_0, \bar{t}_0)$ is an acyclic tour in $\mathfrak{A}$ starting at a_2; (ii) $(a_0 \bar{a}_1, f\bar{t}_1)$ a tour in $\mathfrak{A}$ where $\bar{a}_1$ is a non-empty sequence starting with a_1; (iii) $t_1(a_2, a_1)$ is true in $\mathfrak{A}$ (Fig. 6(V)).*

Proof. Write $\bar{a} = a_0 \cdots a_{\ell-1}$, $\bar{r} = r_0 \cdots r_{\ell-1}$ and $a_\ell = a_0$; thus, $r_0 = f$ and $r_1 = f^{-1}$. Let i be the smallest index $(2 \le i < \ell)$ such that a_{i+1} is active. Since a_ℓ is active, i exists. By Lemma 3.3, $a_i = a_2$ and $a_{i+1} = a_1$. Let $\bar{a}_0 = a_2 \cdots a_{i-1}$ and $\bar{t}_0 = r_2 \ldots r_{i-1}$. By construction, $(\bar{a}_0, \bar{t}_0)$ is a passive—hence acyclic—tour beginning at a_2. Now let $t_1 = r_i$, $\bar{a}_1 = a_{i+1} \cdots a_{\ell-1}$ and $\bar{t}_0 = r_{i+1} \cdots r_{\ell-1}$. $\square$

Lemma 4.6. *Suppose $\mathfrak{A}$ is a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure containing a mixed tour, $(\bar{a}, \bar{\mathbf{r}}_\kappa)$, whose first three elements, a_0, a_1 and a_2, are distinct. Suppose further that a_1 and a_2 are active, but a_0 is passive. Then there exist decompositions $\bar{a} = a_0 a_1 \bar{a}_0 a_1 \bar{a}_1$ and $\bar{\mathbf{r}}_\kappa = ff^{-1}\bar{t}_0 t_1 \bar{t}_1$, such that (i) $(a_1 \bar{a}_0, f^{-1}\bar{t}_0)$ a tour in $\mathfrak{A}$ where $\bar{a}_0$ is a non-empty sequence starting*

with a_2; (ii) $(\bar{a}_1, \bar{t}_1)$ is an acyclic tour in $\mathfrak{A}$ starting at a_0; (iii) $t_1(a_1, a_0)$ is true in $\mathfrak{A}$ (Fig. 6(VI)).

Proof. Analogous to the proof of Lemma 4.4. ◻

We remark at this point that, if $\mathfrak{A}$ is a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure containing a mixed tour $(\bar{a}; \bar{\mathbf{r}}_\kappa)$ whose first three elements, a_0, a_1 and a_2, are distinct, then it cannot be the case that a_0 and a_2 are active, but a_1 passive, as $a_0 a_1 a_2$ would then form a cycle in the graph of $\mathfrak{A}$. In the statement of the next lemma, all arithmetic performed on indices is assumed to be modulo 3.

Lemma 4.7. *Suppose $\mathfrak{A}$ is a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure containing a mixed tour, $(\bar{a}, \bar{\mathbf{r}}_\kappa)$, whose first three elements, a_0, a_1 and a_2, are distinct and all passive. Then there exist decompositions $\bar{a} = a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{a}_2$ and $\bar{\mathbf{r}}_\kappa = f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$, such that: (i) for all j $(0 \leq j < 3)$ $(\bar{a}_j, \bar{t}_j)$ is a tour in $\mathfrak{A}$ starting at a_{2-j}; (ii) two of these tours are acyclic and the third is mixed; (iii) $t_2(a_1, a_0)$ and $t_1(a_2, a_1)$ are true in $\mathfrak{A}$ (Fig. 6(VII)–(IX)).*

Proof. Write $\bar{a} = a_0 \cdots a_{\ell-1}$, $\bar{r} = r_0 \cdots r_{\ell-1}$ and $a_\ell = a_0$; thus, $r_0 = f$ and $r_1 = f^{-1}$. Let h be the smallest index $2 \leq h < \ell$ such that a_{h+1} is active, and k the largest index $h < k < \ell$ such that a_k is active and a_{k+1} is passive. By Lemma 3.4, $a_h = a_{k+1}$, whence $a_2 \cdots a_h a_{k+2} \cdots a_{\ell-1}$ is a passive tour. Suppose first that a_0 is encountered along the walk $a_2 \cdots a_h$, i.e. there exists j $(2 < j < h)$ such that $a_{j+1} = a_0$ (Fig. 6(VII)). Denote by ℓ' the smallest such value of j. It follows that $\bar{b} = a_0 a_1 a_2 \cdots a_{\ell'}$ is a passive tour. Repeating the reasoning of Lemma 4.1, let i be the largest index $(2 \leq i < \ell')$ such that $a_i = a_2$, let $\bar{a}_0 = a_2 \cdots a_{i-1}$ and $\bar{t}_0 = r_2 \cdots r_{i-1}$. Thus, $(\bar{a}_0, \bar{t}_0)$ is a passive—and hence acyclic—tour starting at a_2. By Lemma 3.2 (applied to $\bar{b}$), $a_{i+1} = a_1$; write $t_1 = r_i$. Now let m be the largest index $(i+1 \leq m < \ell')$ such that $a_m = a_1$. Let $\bar{a}_1 = a_{i+1} \cdots a_{m-1}$ and $\bar{t}_1 = r_{i+1} \cdots r_{m-1}$; thus, $(\bar{a}_1, \bar{t}_1)$ is an acyclic tour starting at a_1. By Lemma 3.2 again, $a_{m+1} = a_0$, whence, in fact $m + 1 = \ell'$; write $t_2 = r_m$. Let $\bar{a}_2 = a_{\ell'} \cdots a_{\ell-1}$ and $\bar{t}_2 = r_{\ell'} \cdots r_{\ell-1}$; then, $(\bar{a}_2, \bar{t}_2)$ is a mixed tour starting at a_0.

Now suppose that a_0 is not encountered again along the walk $a_2 \cdots a_h$, but a_1 is, i.e. there exists j $(2 \leq j < h)$ such that $a_{j+1} = a_1$ (Fig. 6(VIII)). Denote by ℓ' the smallest such value of j. Again, let i be the largest index $(2 \leq i < \ell')$ such that $a_i = a_2$, let $\bar{a}_0 = a_2 \cdots a_{i-1}$ and $\bar{t}_0 = r_2 \cdots r_{i-1}$. Thus, $(\bar{a}_0, \bar{t}_0)$ is an acyclic tour starting at a_2. By Lemma 3.2, $a_{i+1} = a_1$; write $t_1 = r_i$. Since $a_{i+1} = a_1$ and $a_h = a_{k+1}$, we know that $\bar{b} = a_0 a_{i+1} \cdots a_h a_{k+2} \ldots a_{\ell-1}$ is a passive tour. Let m be the largest index $1 \leq m < \ell$ such that a_m occurs in $\bar{b}$ and $a_m = a_1$. By Lemma 3.2 (applied to $\bar{b}$), the next element in the tour $\bar{b}$ must be a_0. Since, by assumption, a_0 does not occur among $a_2 \cdots a_h$, we have $k + 1 \leq m < \ell$. Thus, letting $\bar{a}_1 = a_{i+1} \cdots a_{m-1}$ and $\bar{t}_1 = r_{i+1} \cdots r_{m-1}$, $(\bar{a}_1, \bar{t}_1)$ is a mixed tour starting at a_1. Let $t_2 = r_m$. Let $\bar{a}_2 = a_{m+1} \cdots a_{\ell-1}$ and $\bar{t}_1 = r_{m+1} \cdots r_{\ell-1}$; thus, $(\bar{a}_2, \bar{t}_2)$ is a passive—and hence acyclic—tour starting at a_0.

Finally, suppose that neither a_0 nor a_1 is encountered along the walk $a_2 \cdots a_h$ (Fig. 6(IX)). Since $a_h = a_{k+1}$, we know that $\bar{b} = a_0 a_1 \cdots a_h a_{k+2} \ldots a_{\ell-1}$ is a passive—and hence acyclic—tour. Let i be the largest index $(2 \leq i < \ell)$ such that a_i occurs in $\bar{b}$ and $a_i = a_2$. By Lemma 3.2, the next element in the tour $\bar{b}$ must be a_1. Since, by assumption, a_1 does not occur among $a_2 \cdots a_h$, we have $k + 1 \leq i < \ell$, and hence $a_{i+1} = a_1$. Let $\bar{a}_0 = a_2 \cdots a_{i-1}$ and $\bar{t}_0 = r_2 \cdots r_{i-1}$. Thus, $(\bar{a}_0, \bar{t}_0)$ is a mixed tour starting at a_2. Write $t_1 = r_i$. Since $a_{i+1} = a_1$ and $k + 1 \leq i$, we know that $a_0 a_{i+1} \cdots a_{\ell-1}$ is a passive tour. Let j be the largest index $i + 1 \leq j < \ell$ such that $a_j = a_1$, let $\bar{a}_1 = a_{i+1} \cdots a_{j-1}$, and let $\bar{t}_1 = r_{i+1} \cdots r_{j-1}$. Thus,

$(\bar{a}_1, \bar{t}_1)$ is a passive—and hence acyclic—tour starting at a_1. Lemma 3.2, $a_{j+1} = a_0$; let $t_2 = r_j$. Let $\bar{a}_2 = a_{j+1} \cdots a_{\ell-1}$, $\bar{t}_2 = r_{j+1} \cdots r_{\ell-1}$. Thus $(\bar{a}_2, \bar{t}_2)$ is a passive—and hence acyclic—tour starting at a_0. $\qquad\square$

Lemma 4.8. *Suppose $\mathfrak{A}$ is a $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic structure containing a mixed tour, $(\bar{a}, \bar{\mathbf{r}}_\kappa)$, whose first three elements, a_0, a_1 and a_2, are distinct. Suppose further that a_0, a_1 and a_2 are active. Then, writing $\bar{a} = a_0 a_1 \bar{b}$ and $\bar{\mathbf{r}}_\kappa = f f^{-1} \bar{t}$, the pair $[\bar{b} a_0, \bar{t}]$ is a walk in $\mathfrak{A}$ starting at a_2 (Fig. 6(x)).*

Proof. Completely trivial. $\qquad\square$

Let us sum up what we have discussed so far. Fix a path-functional dependency $\kappa = \Psi[\bar{f} f, \bar{g}]$, and let ℓ be the length of $\bar{\mathbf{r}}_\kappa$. Any violation of $\Psi[\bar{f} f, \bar{g}]$ in a structure $\mathfrak{A}$ can be identified with a $\bar{\mathbf{r}}_\kappa$-tour in $\mathfrak{A}$ whose first three elements are distinct. Thus, when determining the satisfiability of a $\mathcal{GC}^2 \mathcal{DK}$-formula $\varphi \wedge \Delta \wedge K$, our goal is to determine whether there exists a model of $\varphi \wedge \Delta$ containing no such tours. Indeed, by Lemma 2.3, we may confine attention to ℓ-quasi-acyclic models of $\varphi \wedge \Delta$. Now suppose $\bar{a}$ is a sequence of elements in some such model suspected of being a $\bar{\mathbf{r}}_\kappa$-tour in $\mathfrak{A}$ whose first three elements are distinct. Depending on which (if any) of these three initial elements of an $\bar{\mathbf{r}}_\kappa$-tour $\bar{a}$ belong to the database, $\bar{a}$ can be decomposed in different ways, as seen in Fig. 6. All these decompositions, except for that of Fig. 6(I), involve elements of the database. The configuration of Fig. 6(I), it turns out, can be ruled out by means of a $\mathcal{GC}^2$-formula. In addition, by introducing additional predicates to our original signature, and writing formulas ensuring that these predicates are satisfied by database elements occurring at certain critical points in the various configurations depicted in Fig. 6(II)–(IX), we can reduce the problem of determining the existence of such violating configurations to a simple database check. The next section addresses the task of establishing the required interpretations of these additional predicates.

5. Encoding critical violations in $\mathcal{GC}^2$

Let $\kappa = \Psi[\bar{f} f, \bar{g}]$ be a binary path-functional dependency, and let $\bar{\mathbf{r}}_\kappa = f f^{-1} \bar{f}^{-1} \bar{g} \bar{g}^{-1} \bar{f}$ as above. In this section, we present three technical devices consisting of sets of $\mathcal{GC}^2$-formulas denoted, respectively, F_κ, B_κ and I_κ. The first device enables us to describe acyclic tours in structures, the second to characterize certain sorts of ternary branching structures, and the third to characterize elements connected to each other by a pair of walks forming an acyclic tour. In the next section, we shall employ these devices to characterize the ten cases depicted in Fig. 6, and thus to rule out the presence of violations of κ in a structure using $\mathcal{GC}^2$-formulas. We write $t \in \bar{\mathbf{r}}_\kappa$ to indicate that t occurs in $\bar{\mathbf{r}}_\kappa$, and $\bar{t} \lhd \bar{\mathbf{r}}_\kappa$ to indicate that $\bar{t}$ is a sub-word (i.e. a contiguous sub-sequence) of $\bar{\mathbf{r}}_\kappa$. Observe that the number of sub-words of $\bar{\mathbf{r}}_\kappa$ is $|\bar{\mathbf{r}}_\kappa|(|\bar{\mathbf{r}}_\kappa| + 1)/2$.

Let the signature σ_κ^1 consist of σ together with the unary predicates $\mathrm{fan}\langle \bar{t} \rangle$, one for each sub-word $\bar{t}$ of $\bar{\mathbf{r}}_\kappa$. The intention is that $\mathrm{fan}\langle \bar{t} \rangle$ will be satisfied by an element a in certain sorts of structures just in case a is the start of an acyclic $\bar{t}$-tour. Accordingly, let F_κ be the set consisting of the formula $\forall x \, \mathrm{fan}\langle \epsilon \rangle(x)$ together with all the formulas

$$\forall x \left(\mathrm{fan}\langle \bar{t} \rangle(x) \leftrightarrow \bigvee_{\substack{r, \bar{r}, s, \bar{s}: \\ \bar{t} = r \bar{r} s \bar{s}}} \exists y \, (x \neq y \wedge r(x, y) \wedge \mathrm{fan}\langle \bar{r} \rangle(y) \wedge s(y, x) \wedge \mathrm{fan}\langle \bar{s} \rangle(x)) \right),$$

where $\bar{t} \lhd \bar{\mathbf{r}}_\kappa$ is non-empty. Evidently the size of F_κ is polynomial in the size of $\bar{\mathbf{r}}_\kappa$.
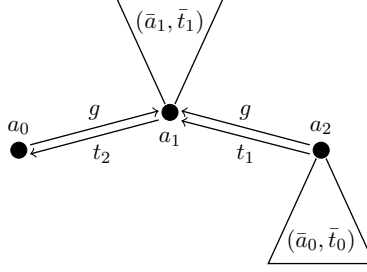
FIGURE 7. Satisfaction of the predicate $\text{branch}_1\langle g, t_1, t_2, \bar{t}_0, \bar{t}_1\rangle$ at a_0.

Lemma 5.1. *Suppose $\mathfrak{A}$ is a σ-structure such that $\mathfrak{A} \models F_\kappa$. Let a be an element of A and $\bar{t}$ a sub-word of $\bar{\mathbf{r}}_\kappa$. If $\mathfrak{A} \models \text{fan}\langle\bar{t}\rangle(a)$, then a is the start of a $\bar{t}$-tour in $\mathfrak{A}$; conversely, if a is the start of an acyclic $\bar{t}$-tour in $\mathfrak{A}$, then $\mathfrak{A} \models \text{fan}\langle\bar{t}\rangle(a)$.*

Proof. We prove by induction on the length of $\bar{t}$ that if $\mathfrak{A} \models \text{fan}\langle\bar{t}\rangle(a)$, then a is the start of a $\bar{t}$-tour in $\mathfrak{A}$. If $\bar{t} = \epsilon$, then the result is evident (recall that every single element is the start of an ϵ-tour). Now, if $\bar{t} \neq \epsilon$, since $\mathfrak{A} \models F_\kappa$, we can write $\bar{t} = r\bar{r}s\bar{s}$, for some $\bar{r}$ and $\bar{s}$, such that there exists an $a' \in A$ with $a \neq a'$, $\mathfrak{A} \models r(a, a')$, $\mathfrak{A} \models s(a', a)$, $\mathfrak{A} \models \text{fan}\langle\bar{r}\rangle(a')$ and $\mathfrak{A} \models \text{fan}\langle\bar{s}\rangle(a)$. Thus, by inductive hypothesis, a' is the start of an $\bar{r}$-tour and a is the start of an $\bar{s}$-tour. Clearly, then, a is the start of the $\bar{t}$-tour.

For the converse, suppose that a is the start of an acyclic $\bar{t}$-tour $\bar{a}$ in $\mathfrak{A}$. We prove, by induction on the length of $\bar{t}$, that $\mathfrak{A} \models \text{fan}\langle\bar{t}\rangle(a_0)$. Write $\bar{a} = a_0, \ldots, a_{\ell-1}$ and $\bar{t} = t_0, \ldots, t_{\ell-1}$, as usual, letting $a_\ell = a_0$. If $\bar{t} = \epsilon$ (i.e. $\ell = 0$) then, since $\mathfrak{A} \models F_\kappa$ we have $\mathfrak{A} \models \text{fan}\langle\epsilon\rangle(a)$. Now, suppose that $\bar{h} \neq \epsilon$ (i.e. $\ell > 0$). Then $\mathfrak{A} \models t_0(a_0, a_1)$. Let j be the largest index ($1 \leq j < \ell$) such that $a_j = a_1$. By Lemma 3.2, $a_{j+1} = a_0$, whence $\mathfrak{A} \models t_j(a_1, a_0)$. Write $r = t_0$, $\bar{r} = t_1 \cdots t_{j-1}$, $s = t_j$ and $\bar{s} = t_{j+1} \cdots t_{\ell-1}$. Thus, $\bar{t} = r\bar{r}s\bar{s}$, a_1 is the start of an acyclic $\bar{r}$-tour, and a_0 the start of an acyclic $\bar{s}$-tour. By inductive hypothesis, $\mathfrak{A} \models \text{fan}\langle\bar{r}\rangle(a_1)$ and $\mathfrak{A} \models \text{fan}\langle\bar{s}\rangle(a_0)$. Since $\mathfrak{A} \models F_\kappa$, we have $\mathfrak{A} \models \text{fan}\langle\bar{t}\rangle(a_0)$. $\square$

Now for our second device, which enables us to identify ternary-branching structures of the kind illustrated in Fig. 6(II). Let the signature σ_κ^2 consist of σ_κ^1 together with the unary predicates $\text{branch}_1\langle g, t_1, t_2, \bar{t}_0, \bar{t}_1\rangle$, where g, t_1 and t_2 are letters in $\bar{\mathbf{r}}_\kappa$, and $\bar{t}_0$, $\bar{t}_1$ are sub-words of $\bar{\mathbf{r}}_\kappa$. The intention is that $\text{branch}_1\langle g, t_1, t_2, \bar{t}_0, \bar{t}_1\rangle$ should be satisfied by an element a_0 just in case there exists a $(gg^{-1}\bar{t}_0 t_1 \bar{t}_1 t_2)$-tour $a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1$, where $\bar{a}_0$ is a $\bar{t}_0$-tour starting at a_2, and $\bar{a}_1$ a $\bar{t}_1$-tour starting at a_1, as depicted in Fig. 7. Let B_κ^1 be the set of all formulas

$$\forall x(\text{branch}_1\langle g, t_1, t_2, \bar{t}_0, \bar{t}_1\rangle(x) \leftrightarrow$$
$$\exists y \exists z(x \neq y \wedge y \neq z \wedge x \neq z \wedge$$
$$(g(x, y) \wedge t_2(y, x) \wedge \text{fan}\langle\bar{t}_1\rangle(y)) \wedge (g(z, y) \wedge t_1(z, y) \wedge \text{fan}\langle\bar{t}_0\rangle(z)))),$$

where $g, t_1, t_2 \in \bar{\mathbf{r}}_\kappa$, $\bar{t}_0 \lhd \bar{\mathbf{r}}_\kappa$ and $\bar{t}_1 \lhd \bar{\mathbf{r}}_\kappa$.

Lemma 5.2. *Suppose $\mathfrak{A}$ is a structure such that $\mathfrak{A} \models F_\kappa \cup B_\kappa^1$. Then $a_0 \in A$ satisfies $\text{branch}_1\langle g, t_1, t_2, \bar{t}_0, \bar{t}_1\rangle$ if and only if there exist elements $a_1, a_2 \in A$ such that a_0, a_1, a_2 are distinct, $\mathfrak{A} \models g(a_0, a_1)$, $\mathfrak{A} \models g^{-1}(a_1, a_2)$, $\mathfrak{A} \models t_1(a_2, a_1)$, $\mathfrak{A} \models t_2(a_1, a_0)$, $\mathfrak{A} \models \text{fan}\langle\bar{t}_0\rangle(a_2)$ and $\mathfrak{A} \models \text{fan}\langle\bar{t}_1\rangle(a_1)$.*
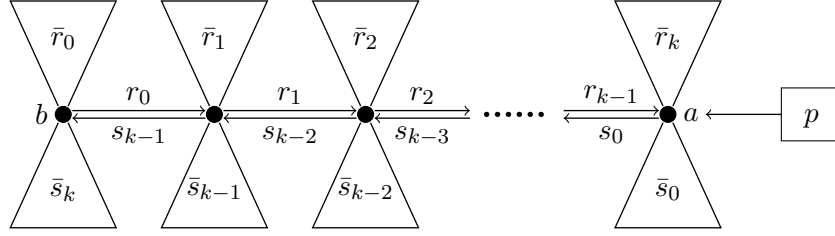
Proof. Immediate. $\square$

FIGURE 8. An isthmus starting at a and terminating in an element b satisfying a unary predicate p; here, $\bar{r} = \bar{r}_0 r_0 \cdots \bar{r}_k r_k$ and $\bar{s} = \bar{s}_0 s_0 \cdots \bar{s}_k s_k$.

We also introduce an additional family of unary predicates $\mathrm{branch}_2\langle g, t_1, t_2, \bar{t}_0, \bar{t}_2\rangle$, satisfied by elements in configurations such as that of a_1 in Fig. 6(IV), where $g = f$. To this end, we define B_κ^2 to be the set of formulas

$$\forall y(\mathrm{branch}_2\langle g, t_1, t_2, \bar{t}_0, \bar{t}_2\rangle(y) \leftrightarrow$$
$$\exists x \exists z (x \neq y \wedge y \neq z \wedge x \neq z \wedge$$
$$(g(x,y) \wedge t_2(y,x) \wedge \mathrm{fan}\langle \bar{t}_2\rangle(x)) \wedge (g(z,y) \wedge t_1(z,y) \wedge \mathrm{fan}\langle \bar{t}_0\rangle(z)))),$$

where $g, t_1, t_2, \bar{t}_0, \bar{t}_1$ have the same range as for B_1.

Lemma 5.3. *Suppose $\mathfrak{A}$ is a structure such that $\mathfrak{A} \models F_\kappa \cup B_\kappa^2$. Then $a_1 \in A$ satisfies $\mathrm{branch}_2\langle g, t_1, t_2, \bar{t}_0, \bar{t}_1\rangle$ if and only if there exist elements $a_0, a_2 \in A$ such that a_0, a_1, a_2 are distinct, $\mathfrak{A} \models g(a_0, a_1)$, $\mathfrak{A} \models g^{-1}(a_1, a_2)$, $\mathfrak{A} \models t_1(a_2, a_1)$, $\mathfrak{A} \models t_2(a_1, a_0)$, $\mathfrak{A} \models \mathrm{fan}\langle \bar{t}_0\rangle(a_2)$ and $\mathfrak{A} \models \mathrm{fan}\langle \bar{t}_2\rangle(a_0)$.*

Of course, the formulas in B_κ^1 and B_κ^2 are not in $\mathcal{GC}^2$, as they feature three variables. However, Lemma 2.2 ensures that they are equivalent to $\mathcal{GC}^2$-formulas. In the sequel, therefore, we shall treat B_κ^1 and B_κ^2 as sets of $\mathcal{GC}^2$-formulas, understanding them to be replaced by their (harder-to-read) $\mathcal{GC}^2$-equivalents. We write $B_\kappa = B_\kappa^1 \cup B_\kappa^2$.

Our third device enables us to identify elements at the start of structures of the kind illustrated in Fig. 8, whose final elements satisfy a given unary predicate p of σ_κ^2. Let the signature σ_κ^3 consist of σ_κ^2 together with the unary predicates $\mathrm{isth}\langle \bar{r}, p, \bar{s}\rangle$, where $\bar{r}$ and $\bar{s}$ are sub-words of $\bar{\mathbf{r}}_\kappa$, and p is any unary predicate of σ_κ^2. (Note that σ_κ^2 contains predicates of the forms $\mathrm{branch}_1\langle g, t_1, t_2, \bar{t}_0, \bar{t}_1\rangle$ and $\mathrm{branch}_2\langle g, t_1, t_2, \bar{t}_0, \bar{t}_2\rangle$.) The intention is that $\mathrm{isth}\langle \bar{r}, p, \bar{s}\rangle$ will be satisfied by an element b in certain sorts of structures just in case there exists an element a satisfying p, as well as an $\bar{r}$-walk from b to a and an $\bar{s}$-walk from a to b, which together form acyclic $\bar{r}\bar{s}$-tour. We refer to such a tour, informally, as an *isthmus*. Accordingly, let I_κ be the set of all formulas

$$\forall x \big[\mathrm{isth}\langle \bar{r}, p, \bar{s}\rangle(x) \leftrightarrow \big((\mathrm{fan}\langle \bar{r}\rangle(x) \wedge p(x) \wedge \mathrm{fan}\langle \bar{s}\rangle(x)) \vee$$
$$\bigvee_{\substack{\bar{r}=\bar{r}'r\bar{r}'', \\ \bar{s}=\bar{s}''s\bar{s}'}} \big(\mathrm{fan}\langle \bar{r}'\rangle(x) \wedge \mathrm{fan}\langle \bar{s}'\rangle(x) \wedge \exists y (x \neq y \wedge r(x,y) \wedge s(y,x) \wedge \mathrm{isth}\langle \bar{r}'', p, \bar{s}''\rangle(y)) \big) \big) \big],$$

where $\bar{r} \lhd \bar{\mathbf{r}}_\kappa$, $\bar{s} \lhd \bar{\mathbf{r}}_\kappa$ and p is a unary predicate of σ_κ^2. Again, the sizes of the signature σ_κ^3 and the set of formulas I_κ are polynomially bounded in $|\bar{\mathbf{r}}_\kappa|$.

Lemma 5.4. *Let $\bar{r}' \lhd \bar{\mathbf{r}}_\kappa$, $\bar{r}'' \lhd \bar{\mathbf{r}}_\kappa$ and $p \in \sigma_\kappa^2$. Suppose $\mathfrak{A} \models F_\kappa \cup I_\kappa$. If $\mathfrak{A} \models \mathrm{isth}\langle \bar{r}, p, \bar{s}\rangle(b)$, then there exists an element $a \in A$ such that $\mathfrak{A} \models p(a)$, a walk $[\bar{b}a, \bar{r}]$ from b to a, and a walk*

$[\bar{c}b, \bar{s}]$ *from a to b. Conversely, if such a, $\bar{b}$, $\bar{r}$, $\bar{c}$, $\bar{s}$ exist, with $(\bar{b}\bar{c}, \bar{r}\bar{s})$ additionally an acyclic tour, then $\mathfrak{A} \models \mathrm{isth}\langle\bar{r}, p, \bar{s}\rangle(b)$.*

Proof. Suppose that $\mathfrak{A} \models \mathrm{isth}\langle\bar{r}, p, \bar{s}\rangle(b)$. To construct the required a, $\bar{b}$, $\bar{r}$, $\bar{c}$ and $\bar{s}$, we proceed by induction on the length of the combined sequence $\bar{r}\bar{s}$. For the case $|\bar{r}\bar{s}| = 0$, if $\mathfrak{A} \models \mathrm{fan}\langle\bar{r}\rangle(b) \wedge p(b) \wedge \mathrm{fan}\langle\bar{s}\rangle(b)$, then by Lemma 5.1, there exists an $\bar{r}$-tour $\bar{b}$ and a $\bar{s}$-tour $\bar{c}$, both starting at b. Now set $a = b$. Thus, $[\bar{b}a, \bar{r}]$ is a walk from b to itself, and $[\bar{c}b, \bar{s}]$ a walk from a to itself, as required. For the case $|\bar{r}\bar{s}| > 0$, since $\mathfrak{A} \models I_\kappa$, we may write $\bar{r} = \bar{r}'r\bar{r}''$ and $\bar{s}'' = \bar{s}''s\bar{s}'$, such that $\mathfrak{A} \models \mathrm{fan}\langle\bar{r}'\rangle(b) \wedge \mathrm{fan}\langle\bar{s}'\rangle(b)$ and there exists $b' \in A$ with $\mathfrak{A} \models r(b, b')$, $\mathfrak{A} \models s(b', b)$ and $\mathfrak{A} \models \mathrm{isth}\langle\bar{r}'', p, \bar{s}''\rangle(b')$. By inductive hypothesis, there exist an element $a \in A$ and walks $[\bar{b}''a, \bar{r}'']$, $[\bar{c}''b', \bar{s}'']$, with b' the first element of $\bar{b}''$, a the first element of $\bar{c}''$, and $\mathfrak{A} \models p(a)$. On the other hand, from Lemma 5.1, b is the start of an $\bar{r}'$-tour, say $\bar{b}'$, and of an $\bar{s}'$-tour, say $\bar{c}'$. Writing $\bar{b} = \bar{b}'b'\bar{b}''$ and $\bar{c} = \bar{c}''b'\bar{c}'$, we see that $\bar{b}a$ is an $\bar{r}$-walk from b to a, and $\bar{c}b$ is an $\bar{s}$-walk from a to b, as required.

For the converse, suppose that there exist an element $a \in A$ and walks $[\bar{b}a, \bar{r}]$, $[\bar{c}b, \bar{s}]$, with b the first element of $\bar{b}$ and a the first element of $\bar{c}$, such that $(\bar{b}\bar{c}, \bar{r}\bar{s})$ is an acyclic tour, and $\mathfrak{A} \models p(b)$. We again proceed by induction on $|\bar{r}\bar{s}|$, supposing that the result holds for sequences of smaller combined length. Write $\bar{b}b = b_0 \cdots b_\ell$ and $\bar{c}a = c_0 \cdots c_m$. Thus, $b = b_0 = c_m$ and $a = b_\ell = c_0$. Let i be the largest index $(0 \le i \le \ell)$ such that $b_i = b_0$. If $i = m$, then $a = b$. In that case, a satisfies p; moreover, $(\bar{b}, \bar{r})$ and $(\bar{c}, \bar{s})$ are acyclic tours, and so by Lemma 5.1, b satisfies $\mathrm{fan}\langle\bar{r}\rangle$ and $\mathrm{fan}\langle\bar{s}\rangle$. Since $\mathfrak{A} \models I_\kappa$, a also satisfies $\mathrm{isth}\langle\bar{r}, p, \bar{s}\rangle$. Suppose, then that $i < \ell$, so that $b_i \ne b_{i+1}$. Then $\bar{d} = b_i \cdots b_{\ell-1}c_0 \cdots c_{m-1}$ is an acyclic tour starting at b. Let c be the last element of $\bar{d}$ equal to b_{i+1}. Applying Lemma 3.2 to $\bar{d}$, the next element after c must be $b_i = b_0 = b$; and since $c_0 \ne b_0$ and by construction b_0 does not occur in $b_{i+1} \cdots b_{\ell-1}$, we have $c = c_j$ for some j $(1 \le j < m)$. Thus, $b = b_i = c_{j+1}$. Now let $\bar{b}' = b_0 \cdots b_{i-1}$, $\bar{b}'' = b_{i+1} \cdots b_{\ell-1}$, $\bar{c}'' = c_0 \cdots c_{j-1}$, $\bar{c}' = c_{j+1} \cdots c_{m-1}$; define the sequences $\bar{r}'$, $\bar{r}''$, $\bar{s}''$, $\bar{s}'$ correspondingly. Write $b' = b_{i+1} = c_j$, $r = r_i$ and $s = s_j$. Then $\mathfrak{A} \models r(b, b')$ and $\mathfrak{A} \models s(b', b)$. Moreover, $(\bar{b}', \bar{r}')$ and $(\bar{c}', \bar{s}')$ are acyclic tours beginning at b, $[\bar{b}''a, \bar{s}'']$ is a walk from b' to a and $[\bar{c}''b', \bar{s}'']$ a walk from a to b' such that the tour $(\bar{b}''\bar{c}'', \bar{r}''\bar{s}'')$ is acyclic. By Lemma 3.2, b satisfies $\mathrm{fan}\langle\bar{r}'\rangle$ and $\mathrm{fan}\langle\bar{s}'\rangle$; and by inductive hypothesis, b' satisfies $\mathfrak{A} \models \mathrm{isth}\langle\bar{r}'', p, \bar{s}''\rangle$. Since $\mathfrak{A} \models I_\kappa$, b satisfies $\mathrm{isth}\langle\bar{r}, p, \bar{s}\rangle$. $\square$

6. COMPLEXITY OF $\mathcal{GC}^2\mathcal{DK}$

With these tools at our disposal, we may return to the task of reducing the satisfiability and finite satisfiability problems for $\mathcal{GC}^2\mathcal{DK}$ to the corresponding problems for $\mathcal{GC}^2\mathcal{D}$. As before, let $\kappa = \P[\bar{f}f, \bar{g}]$ be a binary path-functional dependency, and define $\bar{\mathbf{r}}_\kappa = ff^{-1}\bar{f}^{-1}\bar{g}g^{-1}\bar{f}$. We have observed that κ is critically violated in a structure $\mathfrak{A}$ if and only if $\mathfrak{A}$ contains an $\bar{\mathbf{r}}_\kappa$-tour $\bar{a}$ whose first three elements are distinct. Furthermore, we have shown that, in such a case, $\bar{a}$ must fall under exactly one of the ten cases depicted in Fig. 6.

Consider first the case illustrated in Fig. 6(I): $\bar{a}$ is passive. Define $V_{\kappa,(\mathrm{i})}$ to be the $\mathcal{GC}^2$-sentence

$$\exists x \bigvee \big\{ \mathrm{fan}\langle\bar{t}_2\rangle(x) \wedge \mathrm{branch}_1\langle f, t_1, t_2, \bar{t}_0, \bar{t}_1\rangle(x) \mid \bar{\mathbf{r}}_\kappa = ff^{-1}\bar{t}_0t_1\bar{t}_1t_2\bar{t}_2 \big\}.$$

The following lemma states that we can now rule out such critical violations of κ by writing the $\mathcal{GC}^2$-formula $\neg V_{\kappa,(\mathrm{i})}$.

Lemma 6.1. *Suppose $\mathfrak{A} \models F_\kappa \wedge B_\kappa$. If $\mathfrak{A}$ is $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic, but contains a critical violation of κ having the form of a passive $\bar{\mathbf{r}}_\kappa$-tour, then $\mathfrak{A} \models V_{\kappa,(\mathrm{i})}$. Conversely, if $\mathfrak{A} \models V_{\kappa,(\mathrm{i})}$, then $\mathfrak{A}$ contains a critical violation of κ.*

Proof. Suppose $\mathfrak{A}$ contains a passive $\bar{\mathbf{r}}_\kappa$-tour whose first three elements, a_0, a_1 and a_2 are distinct. By Lemma 4.1, there is a decomposition $f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$ of $\bar{\mathbf{r}}_\kappa$ and distinct $a_0, a_1, a_2 \in A$ such that $f(a_0, a_1)$, $f^{-1}(a_1, a_2)$, $t_1(a_2, a_1)$ and $t_2(a_1, a_0)$ all hold, and each a_i $(0 \leq i < 3)$ is the start of a $\bar{t}_{2-i}$ tour. Since $\mathfrak{A}$ is $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic, these tours must be acyclic. By three applications of Lemma 5.1, each a_i satisfies $\mathrm{fan}\langle \bar{t}_{2-i} \rangle$. By Lemma 5.2 and construction of $V_{\kappa,(\mathrm{i})}$, $\mathfrak{A} \models V_{\kappa,(\mathrm{i})}$.

Suppose, conversely, $\mathfrak{A} \models V_{\kappa,(\mathrm{i})}$. By construction of $V_{\kappa,(\mathrm{i})}$ and Lemma 5.2, there is a decomposition $f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$ of $\bar{\mathbf{r}}_\kappa$ and three distinct elements a_0, a_1, a_2 such that $f(a_0, a_1)$, $f^{-1}(a_1, a_2)$, $t_1(a_2, a_1)$, $t_2(a_1, a_0)$ all hold, and each a_i $(0 \leq i < 3)$ satisfies $\mathrm{fan}\langle \bar{t}_{2-i} \rangle$. By Lemma 5.1, each a_i the start of a $\bar{t}_{2-i}$ tour. Thus, a_0 is the start of an $\bar{\mathbf{r}}_\kappa$-tour in $\mathfrak{A}$ whose first elements are distinct, whence $\mathfrak{A}$ contains a critical violation of κ. $\square$

Now consider the case where $(\bar{a}; \bar{\mathbf{r}}_\kappa)$ is as in Fig. 6(II): a_0 is active, but a_1 and a_2 are passive. The following lemma states that we can now rule out such critical violations of κ by checking some conditions on Δ. Note that, if a database Δ is complete with respect to some signature, and p is a predicate in that signature, then, for $\mathfrak{A} \models \Delta$ and $\bar{a}$ a tuple of active elements of $\mathfrak{A}$ of the same arity as p, the conditions $\mathfrak{A} \models p(\bar{a})$ and $p(\bar{a}) \in \Delta$ are equivalent.

Lemma 6.2. *Suppose $\mathfrak{A} \models F_\kappa \wedge B_\kappa \wedge \Delta$, where Δ is complete with respect to σ_κ^3. If $\mathfrak{A}$ is $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic, but contains an $\bar{\mathbf{r}}_\kappa$-tour $a_0 \cdots a_{\ell-1}$ such that a_0, a_1, a_2 are distinct with a_0 active, but a_1 and a_2 passive, then there exists a decomposition*

$$\bar{\mathbf{r}}_\kappa = f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \{\bar{r}_i r_i\}_{i=0}^{m-1} \bar{r}_m$$

and a sequence of database elements $b_0 \cdots b_m$ with $a_0 = b_0 = b_m$, such that: (i) $\mathrm{branch}_1 \langle f, t_1, t_2, \bar{t}_0, \bar{t}_1 \rangle(b_0) \in \Delta$; (ii) $r_i(b_i, b_{i+1}) \in \Delta$ for all i $(0 \leq i < m)$; and (iii) $\mathrm{fan}\langle \bar{r}_i \rangle(b_i) \in \Delta$ for all i $(0 \leq i \leq m)$. Conversely, if such a decomposition and sequence of database elements exists, then $\mathfrak{A}$ contains a critical violation of κ.

Proof. Suppose $\mathfrak{A}$ contains a $\bar{\mathbf{r}}_\kappa$-tour $a_0 \cdots a_{\ell-1}$ such that a_0 is active, but a_1 and a_2 are passive. By Lemma 4.2 there exist decompositions $\bar{a} = a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{a}_2$ and $\bar{\mathbf{r}}_\kappa = f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$, such that $f(a_0, a_1)$, $t_2(a_1, a_0)$, $f(a_2, a_1)$ and $t_1(a_2, a_1)$ are all true in $\mathfrak{A}$, $(\bar{a}_{2-i}, \bar{t}_{2-i})$ $(i = 1, 2)$ is an acyclic tour starting at a_i, and $(\bar{a}_2, \bar{t}_2)$ is a tour in $\mathfrak{A}$ starting at a_0. By two applications of Lemma 5.1 and Lemma 5.2, $\mathrm{branch}_1 \langle f, t_1, t_2, \bar{t}_0, \bar{t}_1 \rangle(a_0) \in \Delta$. Furthermore, by Lemma 3.6, the tour $(\bar{a}_2, \bar{t}_2)$ starting at a_0 can be further decomposed as $(\{\bar{b}_i b_i\}_{i=0}^{m-1} \bar{b}_m, \{\bar{r}_i r_i\}_{i=0}^{m-1} \bar{r}_m)$ such that $(b_0 \cdots b_{m-1}, r_0 \cdots r_{m-1})$ is an active tour, and each $(\bar{b}_i, \bar{r}_i)$ is an acyclic tour beginning at b_i $(1 \leq i \leq m)$. Thus, we have $r_i(b_i, b_{i+1}) \in \Delta$ for all i $(0 \leq i < m)$. Finally, by repeated applications of Lemma 5.1, we have $\mathrm{fan}\langle \bar{r}_i \rangle(b_i) \in \Delta$ all i $(0 \leq i \leq m)$.

Conversely, suppose the decomposition $f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \{\bar{r}_i r_i\}_{i=0}^{m-1} \bar{r}_m$ and database elements $b_0 \cdots b_m$ exist with the advertised properties. By Lemma 5.2 and two applications of Lemma 5.1, $a_0 = b_0$ is the start of an $f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2$-tour, say $\bar{d}$, whose first three elements are distinct. Moreover, by Lemma 5.1, each b_i $(0 \leq i \leq m)$ is the start of an $\bar{r}_i$ tour, say $\bar{b}_i$. Concatenating these tours, a_0 is the start of a $\bar{\mathbf{r}}_\kappa$-tour, namely $\bar{d}\{\bar{b}_i b_i\}_{i=0}^{m-1} \bar{b}_m$, whose first three elements are distinct, whence $\mathfrak{A}$ contains a critical violation of κ. $\square$

Violations of κ having the forms shown in Fig. 6(III)–(VI) can be ruled out in essentially the same way as for Fig. 6(II). Fig. 6(III) is symmetric to Fig. 6(II); Fig. 6(IV) is similar to Fig. 6(II), but using the predicate branch_2 correspondingly. Fig. 6(V) is, again, similar to Fig. 6(II), where the predicate $\text{branch}_1\langle f, t_1, t_2, \bar{t}_0, \bar{t}_1 \rangle$ is satisfied by $a_0 = b_0$, but with the extra requirement that $a_1 = b_1$, $f(b_0, b_1) \in \Delta$, and $\bar{t}_1 = \epsilon$. (Note, in the previous sentence, that the element b_1 is unique because f is functional.) Fig. 6(VI) is symmetric to Fig. 6(V). Consequently, we turn our attention to the case of Fig. 6(VII)–(IX).

Lemma 6.3. *Suppose* $\mathfrak{A} \models F_\kappa \wedge B_\kappa \wedge I_\kappa \wedge \Delta$, *where* Δ *is complete with respect to* σ_κ^3. *If* $\mathfrak{A}$ *is* $|\bar{\mathbf{r}}_\kappa|$-*quasi-acyclic, but contains a mixed* $\bar{\mathbf{r}}_\kappa$-*tour* $\bar{a}$ *whose first three elements are all passive, then there exists a decomposition* $\bar{\mathbf{r}}_\kappa = f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$ *and a sequence of active elements* $b_0 \cdots b_m$, *with* $b_m = b_0$, *such that the following hold:*

(i) *one of the three conditions*

    (a) $\bar{t}_2 = \bar{s}\{\bar{r}_i r_i\}_{i=0}^{m-1} \bar{r}_m \bar{r}$, $\text{isth}\langle \bar{r}, \text{branch}_1\langle f, t_1, t_2, \bar{t}_0, \bar{t}_1 \rangle, \bar{s} \rangle(b_0) \in \Delta$

    (b) $\bar{t}_1 = \bar{s}\{\bar{r}_i r_i\}_{i=0}^{m-1} \bar{r}_m \bar{r}$, $\text{isth}\langle \bar{r}, \text{branch}_2\langle f, t_1, t_2, \bar{t}_2, \bar{t}_0 \rangle, \bar{s} \rangle(b_0) \in \Delta$

    (c) $\bar{t}_0 = \bar{s}\{\bar{r}_i r_i\}_{i=0}^{m-1} \bar{r}_m \bar{r}$, $\text{isth}\langle \bar{r}, \text{branch}_1\langle f, t_2^{-1}, t_1^{-1}, \bar{t}_1, \bar{t}_2 \rangle, \bar{s} \rangle(b_0) \in \Delta$

    obtains;

(ii) $r_i(b_i, b_{i+1}) \in \Delta$ *for all* i *$(0 \le i < m)$; and*

(iii) $\text{fan}\langle \bar{r}_i \rangle(b_i) \in \Delta$ *for all* i *$(0 \le i \le m)$.*

Conversely, if any such decomposition and sequence of database elements exists, then $\mathfrak{A}$ *contains a critical violation of* κ.

Proof. Suppose $\mathfrak{A}$ contains a mixed $\bar{\mathbf{r}}_\kappa$-tour $\bar{a}$ whose first three elements are all passive. By Lemma 4.7, there exist decompositions $\bar{a} = a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{a}_2$ and $\bar{\mathbf{r}}_\kappa = f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \bar{t}_2$, such that: (i) for all j $(0 \le j < 3)$ $(\bar{a}_j, \bar{t}_j)$ is a tour in $\mathfrak{A}$ starting at a_{2-j}; (ii) two of these tours are acyclic and the third is mixed; (iii) $t_2(a_1, a_0)$ and $t_1(a_2, a_1)$ are true in $\mathfrak{A}$.

Assume, for definiteness, that $(\bar{a}_0, \bar{t}_0)$ and $(\bar{a}_1, \bar{t}_1)$ are acyclic, and $(\bar{a}_2, \bar{t}_2)$ is mixed. By two applications of Lemma 5.1 and Lemma 5.2, $\text{branch}_1\langle f, t_1, t_2, \bar{t}_0, \bar{t}_1 \rangle(a_0) \in \Delta$. Now consider the mixed tour $(\bar{a}_2, \bar{t}_2)$, starting at the passive element a_0. Write $a = a_0$, the initial element of this tour. By Lemma 3.5, there exists an element b such that we can write $\bar{a}_2 = \bar{c} \bar{a}^* \bar{b}$ and $\bar{t}_2 = \bar{s} \bar{t}^* \bar{r}$, where $(\bar{b}a, \bar{r})$ is a walk from b to a, $(\bar{c}b, \bar{s})$ is a walk from a to b, and $(\bar{a}^*, \bar{t}^*)$ is a tour starting at b. Write $b_0 = b$. Now applying Lemma 5.4, we see that $\text{isth}\langle \bar{r}, \text{branch}_1\langle f, t_1, t_2, \bar{t}_0, \bar{t}_1 \rangle, \bar{s} \rangle(b_0) \in \Delta$. Further, applying Lemma 3.6 to the tour $(\bar{a}^*, \bar{t}^*)$, we may write $\bar{a}^* = \{\bar{b}_i b_i\}_{i=0}^{m-1} \bar{b}_m$ and $\bar{t}^* = \{\bar{r}_i r_i\}_{i=0}^{m-1} \bar{r}_m$ such that $r_i(b_i, b_{i+1}) \in \Delta$ for all i $(0 \le i < m)$, and $(\bar{b}_i, \bar{r}_i)$ is an acyclic tour starting at b_i for all i $(0 \le i \le m)$. By repeated applications of Lemma 5.1, $\text{fan}\langle \bar{r}_i \rangle(b_i) \in \Delta$ for all i $(0 \le i \le m)$. This yields the conditions required in the lemma, with disjunct (a) realized in condition (i). If $(\bar{a}_1, \bar{t}_1)$ is mixed, then we obtain disjunct (b), and if $(\bar{a}_0, \bar{t}_0)$ is mixed, disjunct (c), by similar arguments.

Conversely, suppose that there exists a decomposition as described by the lemma, with disjunct (a) realized in condition (i). By (i)(a) and Lemma 5.4, there exist an element $a \in A$, a walk $[\bar{b}a, \bar{r}]$ from b_0 to a, and a walk $[\bar{c}b_0, \bar{s}]$ from a to b_0, such that $(\bar{b}\bar{c}; \bar{r}\bar{s})$ is an acyclic tour, and $\mathfrak{A} \models \text{branch}_1\langle f, t_1, t_2, \bar{t}_0, \bar{t}_1 \rangle(a)$. Write $a_0 = a$, so that, by Lemma 5.2, there exist elements $a_1, a_2 \in A$ such that a_0, a_1, a_2 are distinct, $\mathfrak{A} \models f(a_0, a_1)$, $\mathfrak{A} \models f(a_2, a_1)$, $\mathfrak{A} \models t_1(a_2, a_1)$, $\mathfrak{A} \models t_2(a_1, a_0)$ and for $i = 1, 2$, $\mathfrak{A} \models \text{fan}\langle \bar{t}_{2-i} \rangle(a_i)$. By Lemma 5.1, there is an $\bar{t}_{2-i}$-tour starting at a_i for $i = 1, 2$. Starting at the element $a = a_0$, and concatenating these various tours and single steps, we obtain the tour

$$(a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{c} \bar{b}, \ f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \bar{s} \bar{r})$$

passing through b_0. Note that b_0 is the initial element of $\bar{b}$. From condition (ii) and repeated applications of Lemma 5.1, there exists a tour $(\bar{b}_j, \bar{r}_j)$ starting at b_j for all j ($0 \leq j < m$) and a tour $(\bar{b}_m, \bar{r}_m)$ starting at b_0. Thus, from condition (iii), there exists a tour

$$(\bar{b}_0 b_0 \ldots \bar{b}_{m-1} b_{m-1} \bar{b}_m, \ r_0 \bar{r}_0 \ldots \bar{r}_{m-1} r_{m-1} \bar{r}_m)$$

starting at b_0. Inserting the second of these tours into the first at the element b_0, we obtain the tour

$$(a_0 a_1 \bar{a}_0 a_2 \bar{a}_1 a_1 \bar{c} \{\bar{b}_i b_i\}_{i=0}^{m-1} \bar{b}_m \bar{b}, \ f f^{-1} \bar{t}_0 t_1 \bar{t}_1 t_2 \bar{s} \{\bar{r}_i r_i\}_{i=0}^{m-1} \bar{r}_m \bar{r}).$$

But this is just an $\bar{\mathbf{r}}_\kappa$-tour whose first three elements, a_0, a_1, a_2, are distinct, so that κ has a critical violation. The disjuncts (b) and (c) in condition (i) are dealt with similarly. $\qquad \square$

This leaves just the final case of Fig. 6 to cover.

Lemma 6.4. *Suppose $\mathfrak{A} \models F_\kappa \cup B_\kappa \cup I_\kappa$. If $\mathfrak{A}$ is $|\bar{\mathbf{r}}_\kappa|$-quasi-acyclic, but contains a mixed $\bar{\mathbf{r}}_\kappa$-tour whose three initial elements are all active, then there exist a decomposition $\bar{\mathbf{r}}_\kappa = r_0 r_1 \{\bar{r}_i r_i\}_{i=2}^m \bar{r}_m$ and a sequence of active elements $b_0 \cdots b_m$ with b_0, b_1, b_2 distinct and $b_m = b_0$, such that*

(i) *$r_i(b_i, b_{i+1}) \in \Delta$ for all i ($2 \leq i < m$); and*
(ii) *$\mathrm{fan}\langle \bar{r}_i \rangle(b_i) \in \Delta$ for all i ($0 \leq i \leq m$).*

Conversely, if such a decomposition exists, then $\mathfrak{A}$ contains a critical violation of κ.

Proof. Easy using by-now familiar reasoning from Lemmas 3.6, 4.8 and 5.1. $\qquad \square$

Lemmas 6.1–6.4 yield a reduction from the satisfiability and finite satisfiability problems for $\mathcal{GC}^2 \mathcal{DK}$ to the corresponding problems for $\mathcal{GC}^2 \mathcal{D}$. For definiteness, we consider satisfiability; the reduction for finite satisfiability is identical. By Lemma 2.1, we may confine attention to $\mathcal{GC}^2$-formulas of the form $\psi \wedge \Gamma \wedge K$, where ψ is a $\mathcal{GC}^2$-formula, Γ a database and K a set of unary and binary path-functional dependencies. We may assume that Γ is consistent, since if not, ψ is certainly unsatisfiable. Furthermore, we may suppose that every $\kappa \in K$ is binary, since unary path-functional dependencies can be easily eliminated. Let F_K be the conjunction of all the F_κ for $\kappa \in K$; similarly for B_K and I_K. Let N_K be the conjunction of all the formulas $\neg V_\kappa$, for $\kappa \in K$. Finally, let the signature σ_K^3 be the union of all the σ_κ^3 for $\kappa \in K$. Consider each of the exponentially many consistent completions $\Delta \supseteq \Gamma$ over σ_K^3, in turn. For each such Δ, check the $\mathcal{GC}^2 \mathcal{D}$-formula

$$\varphi \wedge B_K \wedge F_K \wedge I_K \wedge N_K,$$

is satisfiable, and that none of the ten types of violations of κ depicted in Fig. 6(II)–(X) arises, by checking that Δ satisfies the relevant conditions in Lemmas 6.2–6.4. (Note that the truth of formulas N_K rules out violations of κ of the type depicted in Fig. 6(I), by Lemma 6.1.) If, for some Δ, these checks succeed, we may report that φ is satisfiable. Otherwise, we may report that it is unsatisfiable.

Anticipating Theorems 9.5 and 9.6, stating that the satisfiability and finite satisfiability problems for $\mathcal{GC}^2 \mathcal{D}$ are in EXPTIME, we have our main result:

Theorem 6.5. *The satisfiability and finite satisfiability problems for $\mathcal{GC}^2 \mathcal{DK}$ are in EXPTIME.*

7. Complexity of $\mathcal{GC}^2\mathcal{D}$: preliminaries

It remains only to establish that the satisfiability and finite satisfiability problems for $\mathcal{GC}^2\mathcal{D}$ are in ExpTime (Theorems 9.5 and 9.6). The proof given here is a modification of the proof in [PH07] that the finite satisfiability problem for $\mathcal{GC}^2$ is in ExpTime. Unfortunately, guarded formulas cannot straightforwardly be used to express databases, and the authors know of no natural reduction from (finite) $\mathcal{GC}^2\mathcal{D}$-satisfiability to (finite) $\mathcal{GC}^2$-satisfiability: the proof, even though its broad outlines are the same, has to be re-written from scratch. Many of the following lemmas are taken—modulo inessential reformulation—directly from [PH07]. We begin with a normal-form lemma for $\mathcal{GC}^2$-formulas.

Lemma 7.1 ([PH07], Lemma 2). *Let ψ be a $\mathcal{GC}^2$-formula. We can compute in time polynomial in $\|\psi\|$, a sentence*

$$\varphi := \forall x\, \alpha \wedge \bigwedge_{1 \leq j \leq n} \forall x \forall y (e_j(x,y) \rightarrow (\beta_j \vee x = y)) \ \wedge$$

$$\bigwedge_{1 \leq i \leq m} \forall x \exists_{=C_i}\, y(o_i(x,y) \wedge x \neq y), \tag{7.1}$$

such that: (i) α is a quantifier- and equality-free formula in one variable, x; (ii) n, m are positive integers; (iii) e_j is a binary predicate different from $=$, for all j $(1 \leq j \leq n)$; (iv) β_j is a quantifier- and equality-free formula; (v) C_i is a positive integer; (vi) o_i is a binary predicate other than $=$, for all i $(1 \leq i \leq m)$; (vii) φ entails ψ; and (viii) any model of ψ can be expanded to a model of φ.

In the sequel, we fix a $\mathcal{GC}^2\mathcal{D}$-formula $\varphi \wedge \Delta$, where φ is of the form (7.1), and Δ a database, over some signature σ. As we are employing the unique names assumption, we continue to write c instead of $c^{\mathfrak{A}}$ where the interpretation $\mathfrak{A}$ is clear from context. We assume that σ contains, for each individual constant c, a unary predicate with the same name, and that Δ contains all the formulas $c(c)$ and $\neg c(d)$ where c and d are distinct individual constants. We refer to these unary predicates as *naming predicates* and to the formulas added to Δ as *naming formulas*. Elements realizing a naming predicate $c(\cdot)$ are to be viewed as candidates for the constant c. When building a model of $\varphi \wedge \Delta$, one of these candidates is chosen to realize the actual constant. Note that the addition of naming predicates and formulas requires at most a polynomial increase in the size of σ and Δ. We further assume, again for technical reasons, that σ contains $\lceil \log((mC)^2 + 1) \rceil$ unary predicates not occurring in φ or Δ. We refer to these unary predicates as *spare predicates*. Notice that the number of spare predicates is bounded by a polynomial function of $\|\varphi\|$.

A 1-*type* is a maximal consistent set of non-ground, equality-free literals (over σ) with x as the only free variable; a 2-*type* is a maximal consistent set of non-ground, equality-free literals (over σ) with x and y as the only free variables. In this article, expressions involving equality do not feature in 1- or 2-types: this includes, in particular, formulas of the forms $x = c$ and $x \neq c$. On the other hand, every individual constant c in σ is also a naming predicate, and literals of the forms $c(x)$ and $\neg c(x)$ do occur in 1- and 2-types. Since Δ is assumed to contain all the naming formulas, if $\mathfrak{A} \models \varphi \wedge \Delta$, then c will be the unique database element of $\mathfrak{A}$ whose 1-type contains the literal $c(x)$. We stress however that, if $\mathfrak{A} \models \varphi \wedge \Delta$, there can be multiple other (non-database) elements of $\mathfrak{A}$ whose 1-type contains the literal $c(x)$. (These elements can be viewed as candidates that where not chosen to realize c when $\mathfrak{A}$ was built.) We use the letters π and τ, possibly with subscripts, to range over 1- and

2-types respectively. Where convenient, we treat a k-type ($k = 1, 2$) as the conjunction of the formulas constituting it.

For a given structure $\mathfrak{A}$ interpreting σ and element $a \in A$, we denote by $\mathrm{tp}_{\mathfrak{A}}[a]$ the unique 1-type τ such that $\mathfrak{A} \models \tau(a)$. In this case, we say that τ is *realized* by a or that a *realizes* τ. If a and b are distinct elements of A, we speak similarly of the 2-type $\mathrm{tp}_{\mathfrak{A}}[a, b]$ *realized* by a and b. Let τ be a 2-type; we denote by τ^{-1} the 2-type which is the result of transposing the variables x and y in τ. Further, we denote by $\mathrm{tp}_1(\tau)$ the result of deleting all literals from τ involving the variable y, and we define $\mathrm{tp}_2(\tau) = \mathrm{tp}_1(\tau^{-1})$. We are to think of $\mathrm{tp}_1(\tau)$ as the 'starting point' of τ and of $\mathrm{tp}_2(\tau)$ as the 'endpoint' of τ. Evidently, if τ is a 2-type, $\mathfrak{A}$ a structure, and a, b distinct elements of A such that $\mathrm{tp}_{\mathfrak{A}}[a, b] = \tau$, then $\mathrm{tp}_{\mathfrak{A}}[b, a] = \tau^{-1}$, $\mathrm{tp}_{\mathfrak{A}}[a] = \mathrm{tp}_1(\tau)$ and $\mathrm{tp}_{\mathfrak{A}}[b] = \mathrm{tp}_2(\tau)$.

By a *vector* we understand an m-dimensional vector over $\mathbb{N}$. We denote the vector $(C_1, \ldots, C_m)$ of counting subscripts occurring in (7.1) by $\boldsymbol{C}$ and the m-dimensional zero vector by $\boldsymbol{0}$. We use $\leq$ for the pointwise ordering on vectors: $\boldsymbol{v} \leq \boldsymbol{w}$ if every component of $\boldsymbol{v}$ is less than or equal to the corresponding component of $\boldsymbol{w}$, and $\boldsymbol{v} < \boldsymbol{w}$ if $\boldsymbol{v} \leq \boldsymbol{w}$ and $\boldsymbol{v} \neq \boldsymbol{w}$; similarly for $\geq$ and $>$. Let $C = \max_{1 \leq i \leq m}\{C_i\}$. The number of vectors $\boldsymbol{u}$ such that $\boldsymbol{u} \leq \boldsymbol{C}$ is evidently bounded by $(C + 1)^m$, and thus by an exponential function of $\|\varphi\|$.

The following notions make specific reference to our fixed formula φ given in (7.1), and in particular to the binary predicates $q_1, \ldots, q_m$ appearing in it. Let τ be a 2-type over σ. We say that τ is a *message-type* if $\models \tau \to o_i(x, y)$, for some i ($1 \leq i \leq m$). For τ a message-type, if τ^{-1} is also a message-type, we say that τ is *invertible*; otherwise, τ is *non-invertible*. Finally, if τ is a 2-type such that neither τ nor τ^{-1} is a message-type, we say that τ is *silent*.

Let $\mathfrak{A}$ be a structure interpreting σ. If a, b are distinct elements of A such that $\mathrm{tp}_{\mathfrak{A}}[a, b] = \tau$ is a message-type, we say, informally, that *a sends a message (of type τ) to b*. We call $\mathfrak{A}$ *chromatic* if the following are true:

- For all $a, b \in \mathfrak{A}$ such that $a \neq b$ and $\mathrm{tp}_{\mathfrak{A}}[a, b]$ is an invertible message-type, we have $\mathrm{tp}_{\mathfrak{A}}[a] \neq \mathrm{tp}_{\mathfrak{A}}[b]$.
- For all pairwise distinct $a, b, c \in \mathfrak{A}$ such that $\mathrm{tp}_{\mathfrak{A}}[a, b]$ and $\mathrm{tp}_{\mathfrak{A}}[b, c]$ are invertible message-types, we have $\mathrm{tp}_{\mathfrak{A}}[a] \neq \mathrm{tp}_{\mathfrak{A}}[c]$.

In other words, a structure $\mathfrak{A}$ is chromatic if no element sends an invertible message to an element with the same 1-type as itself, and no element sends invertible messages to two different elements with the same 1-type as each other. The following lemma shows that for our fixed formula $\varphi \wedge \Delta$ and signature σ, we may restrict attention to chromatic models. This lemma relies on the fact that σ contains the $\lceil \log((mC)^2 + 1) \rceil$ 'spare' predicates not occurring in φ or Δ.

Lemma 7.2 [PH07, Lemma 3]. *If $\varphi \wedge \Delta$ has a model interpreting σ, then it has a chromatic model interpreting σ over the same domain.*

Assume, then, that $\mathfrak{A}$ is a chromatic model of $\varphi \wedge \Delta$ over σ. By adding to Δ all ground literals over σ that are true in $\mathfrak{A}$, we may assume that Δ is complete over σ. Notice that, for a complete database Δ, all models of Δ assign the same 1-type to c, and so we may denote this 1-type by $\mathrm{tp}_{\Delta}[c]$. If c and d are distinct individual constants, we define $\mathrm{tp}_{\Delta}[c, d]$ analogously.

Let $\Pi = \pi_0, \ldots, \pi_{P-1}$ be an enumeration of the 1-types over σ. It is clear that P is a power of 2, thus $p = \log P$ is an integer. We will need to index certain sub-sequences of Π using bit-strings. Let ϵ denote the empty string and define $\Pi_\epsilon = \Pi$ to be the whole sequence.

Now suppose $\Pi_{\mathbf{s}} = \pi_j, \ldots \pi_{k-1}$ has been defined, where $\mathbf{s}$ is a bit-string of length less than p, and $k - j$ is a power of 2. We define $\Pi_{\mathbf{s}0}$ and $\Pi_{\mathbf{s}1}$ to be the first and second halves of $\Pi_{\mathbf{s}}$, respectively:

$$\Pi_{\mathbf{s}0} = \pi_j, \ldots, \pi_{(j+k)/2-1} \quad \text{and} \quad \Pi_{\mathbf{s}1} = \pi_{(j+k)/2}, \ldots, \pi_{k-1}.$$

When $|\mathbf{s}| = p$, it is clear that $\Pi_{\mathbf{s}}$ corresponds to exactly one 1-type π_j: we sometimes denote this type by $\pi_{\mathbf{s}}$. Note that, in this case, $\mathbf{s}$ is the binary representation of the subscript j.

We now index (sequences of) invertible message-types according to their terminal 1-types using bit-strings as follows. Let Λ be the set of all invertible message-types (over σ). Fix any 1-type π, let $\mathbf{s}$ be any bit-string with $0 \leq |\mathbf{s}| \leq p$, and define

$$\Lambda_{\pi,\mathbf{s}} = \{\lambda \in \Lambda \mid \mathrm{tp}_1(\lambda) = \pi \text{ and } \mathrm{tp}_2(\lambda) \in \Pi_{\mathbf{s}}\},$$

i.e. $\Lambda_{\pi,\mathbf{s}}$ is the set of all invertible message-types 'starting' from an element of 1-type π and 'ending' at an element whose 1-type is in $\Pi_{\mathbf{s}}$. Notice that each of the sets $\Lambda_{\pi,\mathbf{s}}$ will usually contain more than one 2-type. This is true even when $|\mathbf{s}| = p$, as there are several ways one could 'connect' an element of 1-type π with another element of 1-type π'. For a *chromatic* model, $\mathfrak{A}$, however, we are guaranteed that no element sends an invertible message-type to an element of the same 1-type and any element a of 1-type π can send an invertible message to no more than one element b of type $\pi' (\neq \pi)$. Thus, when $|\mathbf{s}| = p$, there can be at most one element $b \in A \setminus \{a\}$ such that $\mathrm{tp}_{\mathfrak{A}}[a, b] \in \Lambda_{\pi,\mathbf{s}}$.

In a similar way, we use bit-strings to index sequences of 2-types that are not invertible message-types. Fix any 1-type π and let

$$M_\pi \; = \; \mu_{\pi,0}, \ldots, \mu_{\pi,Q-1}$$

be an enumeration of all 2-types τ with $\mathrm{tp}_1(\tau) = \pi$ that are either non-invertible message-types or silent 2-types. In other words, M_π is an enumeration of all 2-types τ with $\mathrm{tp}_1(\tau) = \pi$ such that τ^{-1} is not a message-type. It follows, then, that Q is a power of 2, thus $q = \log Q$ is an integer. Define $M_{\pi,\epsilon} = M_\pi$. Now suppose $M_{\pi,\mathbf{t}} = \mu_j, \ldots \mu_{k-1}$ has been defined, where $\mathbf{t}$ is a bit-string of length less than q, and $k - j$ is a power of 2. We recursively define

$$M_{\pi,\mathbf{t}0} = \mu_j, \ldots, \mu_{(j+k)/2-1} \quad \text{and} \quad M_{\pi,\mathbf{t}1} = \mu_{(j+k)/2}, \ldots, \mu_{k-1}.$$

Note that if $|\mathbf{t}| = q$, then $M_{\pi,\mathbf{t}}$ contains a single 2-type $\mu_{\pi,j}$, which we often write as $\mu_{\pi,\mathbf{t}}$ for convenience.

8. Reduction to linear programming

In this section, we show how to transform our given $\mathcal{GC}^2\mathcal{D}$-formula, $\varphi \wedge \Delta$, into a system of exponentially many linear inequalities. The variables in these inequalities represent the frequency with which certain (local) configurations are realized in a putative model. We shall show that this system of inequalities has a non-negative integer solution if and only if $\varphi \wedge \Delta$ is finitely satisfiable.

Let $\mathfrak{A}$ be a model of φ interpreting the signature σ. Let $a \in A$ be an element with $\mathrm{tp}_{\mathfrak{A}}[a] = \pi$, and $\mathbf{s}$ a bit-string of length at most p. Define the $\mathbf{s}$-*spectrum* of a in $\mathfrak{A}$, denoted $\mathrm{sp}_{\mathbf{s}}^{\mathfrak{A}}[a]$, to be the m-element vector $\boldsymbol{v} = (v_1, \ldots, v_m)$ where, for $1 \leq i \leq m$,

$$v_i = |\{b \in A \setminus \{a\} : \mathfrak{A} \models o_i(a, b) \text{ and } \mathrm{tp}_{\mathfrak{A}}[a, b] \in \Lambda_{\pi,\mathbf{s}}\}|.$$

Likewise, if $\mathtt{t}$ is a bit-string with $|\mathtt{t}| < q$, define the $\mathtt{t}$-*tally* of a, denoted $\mathrm{tl}_{\mathtt{t}}^{\mathfrak{A}}[a]$, to be the m-element vector $\boldsymbol{w} = (w_1, \ldots, w_m)$ where, for $1 \le i \le m$,

$$w_i = |\{b \in A \setminus \{a\} : \mathfrak{A} \models o_i(a, b) \text{ and } \mathrm{tp}_{\mathfrak{A}}[a, b] \in M_{\pi, \mathtt{t}}\}|.$$

Informally, $\mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[a]$ records the number of o_i-messages ($1 \le i \le m$) sent by the element a, and whose types belong to $\Lambda_{\pi, \mathtt{s}}$. (Note that these message-types are all invertible.) In particular, $\mathrm{sp}_{\epsilon}^{\mathfrak{A}}[a]$ records the number of invertible o_i-messages ($1 \le i \le m$) sent by the element a. Similarly, $\mathrm{tl}_{\mathtt{t}}^{\mathfrak{A}}[a]$ records the number of o_i-messages ($1 \le i \le m$) sent by the element a, and whose types belong to $M_{\pi, \mathtt{t}}$. (Note that these message-types are all non-invertible.) In particular, $\mathrm{tl}_{\epsilon}^{\mathfrak{A}}[a]$ records the number of non-invertible o_i-messages ($1 \le i \le m$) sent by the element a.

Let $\mathtt{s}$ be any bit-string with $|\mathtt{s}| < p$ and fix a 1-type π. For a given structure $\mathfrak{A}$, each vector $\boldsymbol{v}$ specifies a set of elements of $\mathfrak{A}$, i.e. the set of elements of type π that have $\mathtt{s}$-spectrum $\boldsymbol{v}$. The following lemma encapsulates the observation that this set can also be characterized as the union of sets of elements of type π whose $\mathtt{s0}$- and $\mathtt{s1}$-spectra add up to $\boldsymbol{v}$. The same idea applies to tallies.

Lemma 8.1. *Let $\mathfrak{A}$ be a finite model of φ, and let $a \in A$ with $\mathrm{tp}_{\mathfrak{A}}[a] = \pi$. Let $\mathtt{s}$, $\mathtt{t}$ be any bit-strings with $|\mathtt{s}| < p$ and $|\mathtt{t}| < q$. Then, the following equations hold:*

$$\mathrm{sp}_{\epsilon}^{\mathfrak{A}}[a] + \mathrm{tl}_{\epsilon}^{\mathfrak{A}}[a] = \boldsymbol{C} \tag{8.1}$$

$$\mathrm{sp}_{\mathtt{s0}}^{\mathfrak{A}}[a] + \mathrm{sp}_{\mathtt{s1}}^{\mathfrak{A}}[a] = \mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[a] \tag{8.2}$$

$$\mathrm{tl}_{\mathtt{t0}}^{\mathfrak{A}}[a] + \mathrm{tl}_{\mathtt{t1}}^{\mathfrak{A}}[a] = \mathrm{tl}_{\mathtt{t}}^{\mathfrak{A}}[a] \tag{8.3}$$

Proof. Equation (8.1) is immediate from the definition of spectra and tallies and the normal form in Lemma 7.1. For Equation (8.2), notice that the set $\Lambda_{\pi, \mathtt{s}}$ can be partitioned into two subsets $\Lambda_{\pi, \mathtt{s0}}$ and $\Lambda_{\pi, \mathtt{s1}}$, and this induces a partition of the outgoing message-types from a ($1 \le i \le m$); Equation (8.2) is then evident. Likewise for Equation (8.3). $\qquad\square$

Let τ be any 2-type. With τ we associate an m-dimensional vector $\boldsymbol{C}_{\tau}$, whose ith component is given by

$$(\boldsymbol{C}_{\tau})_i = \begin{cases} 1 & \text{if } \models \tau \to o_i(x, y), \\ 0 & \text{otherwise.} \end{cases}$$

Let $\mathfrak{A}$ be a finite model of φ, $a \in A$, and $\mathtt{t}$ a bit-string with $|\mathtt{t}| = q$. Now, consider the (only) 2-type $\mu_{\pi, \mathtt{t}}$ in $M_{\pi, \mathtt{t}}$ and, if $\mu_{\pi, \mathtt{t}}$ is non-silent, let n be the number of messages of type $\mu_{\pi, \mathtt{t}}$ sent by a, i.e. n is the number of elements $b \in A \setminus \{a\}$ such that $\mathrm{tp}_{\mathfrak{A}}[a, b] = \mu_{\pi, \mathtt{t}}$. It is clear, then, that $\mathrm{tl}_{\mathtt{t}}^{\mathfrak{A}}[a] = n \cdot \boldsymbol{C}_{\mu_{\pi, \mathtt{t}}}$. On the other hand, if $\mu_{\pi, \mathtt{t}}$ is silent, we have $\mathrm{tl}_{\mathtt{t}}^{\mathfrak{A}}[a] = \boldsymbol{C}_{\mu_{\pi, \mathtt{t}}} = \boldsymbol{0}$. Let $\mathtt{s}$ be a bit-string with $|\mathtt{s}| = p$, and suppose further that $\mathfrak{A}$ is chromatic. For each element $a \in A$ with 1-type π and non-zero $\mathtt{s}$-spectrum, there is a unique $\lambda \in \Lambda_{\pi, \mathtt{s}}$ (depending on a) such that a sends a message of type λ, and hence has $\mathtt{s}$-spectrum $\boldsymbol{C}_{\lambda}$.

Lemma 8.2. *Let $\mathfrak{A}$ be a chromatic model of φ, $a \in A$, π a 1-type, and $\mathtt{s}$ a bit-string with $|\mathtt{s}| = p$. If $\mathrm{tp}_{\mathfrak{A}}[a] = \pi$ and $\mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[a] \ne \boldsymbol{0}$, then there exists $\lambda \in \Lambda_{\pi, s}$ with $\mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[a] = \boldsymbol{C}_{\lambda}$ such that a sends a message of type λ to some $b \in A \setminus \{a\}$. Conversely, if there exists $\lambda \in \Lambda_{\pi, s}$ such that a sends a message of type λ to some $b \in A \setminus \{a\}$, then $\mathrm{tp}_{\mathfrak{A}}[a] = \pi$ and $\mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[a] = \boldsymbol{C}_{\lambda}$.*

Proof. Suppose $\mathrm{tp}_{\mathfrak{A}}[a] = \pi$ and $\mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[a] \ne \boldsymbol{0}$. As discussed previously, there exists a unique $b \in A \setminus \{a\}$ such that $\lambda = \mathrm{tp}_{\mathfrak{A}}[a, b] \in \Lambda_{\pi, \mathtt{s}}$. Clearly, then, $\mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[a] = \boldsymbol{C}_{\lambda}$. Conversely, suppose

Variable	Intended meaning of its value
x_λ	$|\{a \in A : \text{for some } b \in A \setminus \{a\}, \text{tp}_\mathfrak{A}[a, b] = \lambda\}|$
$y_{\pi, \mathbf{s}, \boldsymbol{u}}$	$|\{a \in A_\pi : \text{sp}_\mathbf{s}^\mathfrak{A}[a] = \boldsymbol{u}\}|$
$z_{\pi, \mathbf{t}, \boldsymbol{u}}$	$|\{a \in A_\pi : \text{tl}_\mathbf{s}^\mathfrak{A}[a] = \boldsymbol{u}\}|$
$\hat{y}_{\pi, \mathbf{s}, \boldsymbol{v}, \boldsymbol{w}}$	$|\{a \in A_\pi : \text{sp}_{\mathbf{s}0}^\mathfrak{A}[a] = \boldsymbol{v} \text{ and } \text{sp}_{\mathbf{s}1}^\mathfrak{A}[a] = \boldsymbol{w}, \text{ whenever } |\mathbf{s}| < p\}|$
$\hat{z}_{\pi, \mathbf{t}, \boldsymbol{v}, \boldsymbol{w}}$	$|\{a \in A_\pi : \text{tl}_{\mathbf{t}0}^\mathfrak{A}[a] = \boldsymbol{v} \text{ and } \text{tl}_{\mathbf{t}1}^\mathfrak{A}[a] = \boldsymbol{w}, \text{ whenever } |\mathbf{t}| < q\}|$

Table 1. Variables and their intended meanings, for a finite model $\mathfrak{A}$ of $\Delta \cup \{\varphi\}$. Recall, $A_\pi = \{a \in A \mid \text{tp}_\mathfrak{A}[a] = \pi\}$.

a sends a message of type $\lambda \in \Lambda_{\pi, \mathbf{s}}$ to some element $b \in A \setminus \{a\}$. Evidently, then, $\text{tp}_\mathfrak{A}[a] = \pi$ and b is unique, thus $\text{sp}_\mathbf{s}^\mathfrak{A}[a] = \boldsymbol{C}_\lambda$. $\qquad \square$

Henceforth, given a structure $\mathfrak{A}$, we will denote the set of elements in the universe A of $\mathfrak{A}$ having 1-type π by A_π, that is $A_\pi = \{a \in A \mid \text{tp}_\mathfrak{A}[a] = \pi\}$. We now proceed to transform the question of whether $\varphi \wedge \Delta$ is finitely satisfiable into the question of whether a certain system of linear equations/inequalities has a solution over $\mathbb{N}$. The solutions of these equations count how often various local configurations appear in a model. These configurations are:
- realizations of each invertible message-type λ;
- elements of 1-type π having $\mathbf{s}$-spectrum $\boldsymbol{u}$, for all vectors $\boldsymbol{u} \leq \boldsymbol{C}$;
- elements of 1-type π having $\mathbf{t}$-tally $\boldsymbol{u}$, for all vectors $\boldsymbol{u} \leq \boldsymbol{C}$;
- elements of 1-type π whose $\mathbf{s}$-spectrum $\boldsymbol{u}$ is obtained as the sum of an $\mathbf{s}0$-spectrum $\boldsymbol{v}$ and an $\mathbf{s}1$-spectrum $\boldsymbol{w}$, for $\boldsymbol{v}, \boldsymbol{u}, \boldsymbol{w} \leq \boldsymbol{C}$ and for all $\mathbf{s}$ with $|\mathbf{s}| < p$;
- elements of 1-type π whose $\mathbf{t}$-tally $\boldsymbol{u}$ is obtained as the sum of a $\mathbf{t}0$-tally $\boldsymbol{v}$ and a $\mathbf{t}1$-tally $\boldsymbol{w}$, for $\boldsymbol{v}, \boldsymbol{u}, \boldsymbol{w} \leq \boldsymbol{C}$ and for all $\mathbf{t}$ with $|\mathbf{t}| < q$.

To each of those configurations we associate a variable which is intended to capture how many times it appears in a model. These variables and the properties that they capture can be seen in Table 1. Unless specified otherwise, the ranges of the subscripts of these variables are as follows: π ranges over all 1-types in Π, λ ranges over all invertible message-types, $\mathbf{s}$ ranges over all bit-strings with $|\mathbf{s}| \leq p$, $\mathbf{t}$ ranges over all bit-strings with $|\mathbf{t}| \leq q$, and $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ vary over all vectors $\leq \boldsymbol{C}$. (Similarly for their primed counterparts $\pi', \lambda', \mathbf{s}', \mathbf{t}', \boldsymbol{u}', \boldsymbol{v}'$ and $\boldsymbol{w}'$.) That is, we have one variable x_λ for each invertible message-type λ, one variable $y_{\pi, \mathbf{s}, \boldsymbol{u}}$ for each possible combination of $\pi \in \Pi$, $\mathbf{s}$ with $|\mathbf{s}| \leq p$ and $\boldsymbol{u} \leq \boldsymbol{C}$, and so on.

We now write a collection of constraints that a given structure $\mathfrak{A}$ has to satisfy for it to be a model of $\varphi \wedge \Delta$. For ease of reading, we divide these constraints into four groups: the first three are imposed by the formula φ, and the fourth, by the database Δ. Let $\mathcal{E}_1$ be the following set of constraints, where indices vary over their standard ranges, but with $\mathbf{s}$ and $\mathbf{t}$

subject to the additional constraint that $|\mathbf{s}| < p$ and $|\mathbf{t}| < q$:

$$y_{\pi,\epsilon,\boldsymbol{u}} = z_{\pi,\epsilon,\boldsymbol{C}-\boldsymbol{u}} \tag{8.4}$$

$$y_{\pi,\mathbf{s},\boldsymbol{u}} = \sum\{\hat{y}_{\pi,\mathbf{s},\boldsymbol{v}',\boldsymbol{w}'} \mid \boldsymbol{v}' + \boldsymbol{w}' = \boldsymbol{u}\} \tag{8.5}$$

$$z_{\pi,\mathbf{t},\boldsymbol{u}} = \sum\{\hat{z}_{\pi,\mathbf{t},\boldsymbol{v}',\boldsymbol{w}'} \mid \boldsymbol{v}' + \boldsymbol{w}' = \boldsymbol{u}\} \tag{8.6}$$

$$y_{\pi,\mathbf{s}0,\boldsymbol{v}} = \sum\{\hat{y}_{\pi,\mathbf{s},\boldsymbol{v},\boldsymbol{w}'} \mid \boldsymbol{v} + \boldsymbol{w}' \leq \boldsymbol{C}\} \tag{8.7}$$

$$y_{\pi,\mathbf{s}1,\boldsymbol{w}} = \sum\{\hat{y}_{\pi,\mathbf{s},\boldsymbol{v}',\boldsymbol{w}} \mid \boldsymbol{v}' + \boldsymbol{w} \leq \boldsymbol{C}\} \tag{8.8}$$

$$z_{\pi,\mathbf{t}0,\boldsymbol{v}} = \sum\{\hat{z}_{\pi,\mathbf{t},\boldsymbol{v},\boldsymbol{w}'} \mid \boldsymbol{v} + \boldsymbol{w}' \leq \boldsymbol{C}\} \tag{8.9}$$

$$z_{\pi,\mathbf{t}1,\boldsymbol{w}} = \sum\{\hat{z}_{\pi,\mathbf{t},\boldsymbol{v}',\boldsymbol{w}} \mid \boldsymbol{v}' + \boldsymbol{w} \leq \boldsymbol{C}\} \tag{8.10}$$

$$1 \leq \sum\{y_{\pi',\epsilon,\boldsymbol{u}'} \mid \pi' \text{ a 1-type}, \boldsymbol{u}' \leq \boldsymbol{C}\} \tag{8.11}$$

Lemma 8.3 ([PH07], Lemma 7). *Let $\mathfrak{A}$ be a finite model of φ. The constraints $\mathcal{E}_1$ are satisfied when the variables take the values specified in Table 1.*

Proof. For (8.4), we see from (8.1) that every element contributing to the count $y_{\pi,\epsilon,\boldsymbol{u}}$ contributes to $z_{\pi,\epsilon,\boldsymbol{C}-\boldsymbol{u}}$, and vice versa. For (8.5), we see from (8.2) that every element contributing to the count $y_{\pi,\mathbf{s},\boldsymbol{u}}$ must contribute to $\hat{y}_{\pi,\mathbf{s},\boldsymbol{v}',\boldsymbol{w}'}$ for some pair of vectors $\boldsymbol{v}'$, $\boldsymbol{w}'$ summing to $\boldsymbol{u}$, and vice versa. The other equations are similar. The inequality (8.11) in effect states that A is non-empty. $\square$

Recalling our fixed formula φ given in (7.1), let τ be any 2-type. We say that τ is *forbidden* if the following formula is unsatisfiable:

$$\bigwedge \tau \wedge \alpha(x) \wedge \alpha(y) \wedge$$
$$\bigwedge_{1 \leq j \leq n} \big((e_j(x,y) \to \beta_j(x,y)) \wedge (e_j(y,x) \to \beta_j(y,x))\big).$$

Evidently, no forbidden 2-type can be realized in any model of φ.

Let $\mathcal{E}_2$ be the following set of constraints, where indices vary over their standard ranges, but with $\mathbf{s}$ and $\mathbf{t}$ subject to the additional constraint that $|\mathbf{s}| = p$ and $|\mathbf{t}| = q$:

$$y_{\pi,\mathbf{s},\boldsymbol{u}} = \sum\{x_{\lambda'} \mid \lambda' \in \Lambda_{\pi,\mathbf{s}} \text{ and } \boldsymbol{C}_{\lambda'} = \boldsymbol{u}\} \tag{8.12}$$

$$x_{\lambda^{-1}} = x_\lambda \tag{8.13}$$

$$x_\lambda = 0, \quad \text{whenever } \mathrm{tp}_1(\lambda) = \mathrm{tp}_2(\lambda) \tag{8.14}$$

$$x_\lambda = 0, \quad \text{whenever } \lambda \text{ is forbidden} \tag{8.15}$$

$$z_{\pi,\mathbf{t},\boldsymbol{u}} = 0, \quad \text{whenever } \mu_{\pi,\mathbf{t}} \text{ is forbidden} \tag{8.16}$$

$$z_{\pi,\mathbf{t},\boldsymbol{u}} = 0, \quad \text{whenever } \boldsymbol{u} \text{ is not a scalar multiple of } \boldsymbol{C}_{\mu_{\pi,\mathbf{t}}} \tag{8.17}$$

Lemma 8.4 ([PH07], Lemma 8). *Let $\mathfrak{A}$ be a finite, chromatic model of φ. The constraints $\mathcal{E}_2$ are satisfied when the variables take the values specified in Table 1.*

Proof. For (8.12), we observed in Lemma 8.2 that every element a contributing to the count $y_{\pi,\mathbf{s},\boldsymbol{u}}$ emits exactly one invertible message whose type λ lies in $\Lambda_{\pi,\mathbf{s}}$, and that $\boldsymbol{u} = \mathrm{sp}_{\mathbf{s}}^{\mathfrak{A}}[a] = \boldsymbol{C}_\lambda$. The other equations are obvious. $\square$

Let $\mathcal{E}_3$ be the following set of constraints, where indices vary over their standard ranges, but with $\mathtt{t}$ and $\boldsymbol{u}$ subject to the additional constraint that $|\mathtt{t}| = q$, and $\boldsymbol{u} \neq \boldsymbol{0}$:

$$z_{\pi,\mathtt{t},\boldsymbol{u}} > 0 \Rightarrow \sum \{y_{\pi',\epsilon,\boldsymbol{u}'} \mid \pi' = \mathrm{tp}_2(\mu_{\pi,\mathtt{t}}) \text{ and } \boldsymbol{u}' \leq \boldsymbol{C}\} > 0 \qquad (8.18)$$

In effect, this inequality states that, if a non-invertible message-type is realized in $\mathfrak{A}$, then it must have a 'landing-site' of the appropriate 1-type. Thus, we have:

Lemma 8.5 [PH07, Lemma 9]. *Let $\mathfrak{A}$ be a finite model of φ. The constraints $\mathcal{E}_3$ are satisfied when the variables take the values specified in Table 1.*

We now turn our attention to the constraints that Δ enforces. (Note that these constraints do not occur at all in [PH07].) These constraints feature the spectra and tallies of our database elements. The difficulty is that, since we do not know how elements in the database are related to those outside it in some putative model $\mathfrak{A}$, these quantities are unknown. Moreover, although the number of database elements is bounded by the size of Δ, the number of values $\mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[c]$ and $\mathrm{tl}_{\mathtt{t}}^{\mathfrak{A}}[c]$ (for each constant $c \in D$) is exponential in the size of φ, and thus too large to simply guess in order to obtain the desired ExpTime-complexity bound. We therefore need to perform this guessing judiciously.

Suppose $\mathfrak{A} \models \varphi \wedge \Delta$ is chromatic: for each constant $c \in D$, let

- S_c be the set of strings $\mathtt{s}$ of length p such that c sends an invertible message in $\mathfrak{A}$ to some database element b of 1-type $\pi_{\mathtt{s}}$; and
- T_c be the set of strings $\mathtt{t}$ of length q such that c sends a non-invertible message in $\mathfrak{A}$ of type $\mu_{\mathtt{t}}$ to some database element.

Thus, S_c is the set of indices of (terminal 1-types of) invertible messages sent by c within the database, and T_c is the set of indices of non-invertible messages sent by c within the database. Obviously, the cardinalities of both sets are bounded by the size of the database.

If $\mathtt{x}$ is any bit-string we denote by $\mathrm{seg}(\mathtt{x})$ the set of all proper initial segments of $\mathtt{x}$ (thus, $\epsilon \in \mathrm{seg}(\mathtt{x})$, but $x \notin \mathrm{seg}(\mathtt{x})$), and we write

$$\mathrm{CC}_{\mathtt{x}} = \{\epsilon\} \cup \{\mathtt{yb} \mid \mathtt{y} \in \mathrm{seg}(\mathtt{x}) \text{ and } \mathtt{b} \in \{\mathtt{0},\mathtt{1}\}\}.$$

Alternatively, $\mathrm{CC}_{\mathtt{x}}$ is the set of all proper initial segments of $\mathtt{x}$ together with their extensions by a single bit. For example,

$$\mathrm{CC}_{\mathtt{0101}} = \{\epsilon, \mathtt{0}, \mathtt{1}, \mathtt{00}, \mathtt{01}, \mathtt{010}, \mathtt{011}, \mathtt{0100}, \mathtt{0101}\}. \qquad (8.19)$$

(The notation is an allusion to the similar notion of C-command in transformational linguistics.) Notice that $|\mathrm{CC}_{\mathtt{x}}| = 2|\mathtt{x}| + 1$. Now define, for any individual constant c,

$$\mathrm{CCS}_c = \{\epsilon\} \cup \bigcup \{\mathrm{CC}_{\mathtt{s}} \mid \mathtt{s} \in S_c\}$$

$$\mathrm{CCT}_c = \{\epsilon\} \cup \bigcup \{\mathrm{CC}_{\mathtt{t}} \mid \mathtt{t} \in T_c\}.$$

The idea is simple: for an individual constant c of type π, CCS_c collects every string $\mathtt{s}$ of length p such that c sends an invertible message of type $\lambda \in \Lambda_{\pi,\mathtt{s}}$ to a database element, together with all of $\mathtt{s}$'s proper prefixes and the extensions of those proper prefixes by a single bit. Notice that CCS_c is always taken to include the empty string even if c sends no invertible messages to any database element: this stipulation avoids some otherwise tedious special cases. Similarly, CCT_c collects every string $\mathtt{t}$ of length q such that c sends a non-invertible message of type $\mu \in M_{\pi,\mathtt{t}}$ to a database element, together with all of $\mathtt{t}$'s proper prefixes and the extensions of those proper prefixes by a single bit; again, we always

add the string ϵ. Both of these sets are polynomially bounded in the size of $\varphi \wedge \Delta$; we shall need to guess $\mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[c]$ only for $\mathtt{s} \in \mathrm{CCS}_c$ and $\mathrm{tl}_{\mathtt{t}}^{\mathfrak{A}}[c]$ only for $\mathtt{t} \in \mathrm{CCT}_c$. We remark that, by construction, for any $\mathtt{s}$ with $0 \leq |\mathtt{s}| < p$, $\mathtt{s}0 \in \mathrm{CCS}_c$ if and only if $\mathtt{s}1 \in \mathrm{CCS}_c$, and $\mathtt{s}0 \in \mathrm{CCS}_c$ implies $\mathtt{s} \in \mathrm{CCS}_c$; similarly with CCT_c.

Keeping our chromatic structure $\mathfrak{A}$ fixed for the moment, for any individual constant c, with $\mathrm{tp}_{\mathfrak{A}}[c] = \pi$, and any $\mathtt{s} \in \mathrm{CCS}_c$, $\mathtt{t} \in \mathrm{CCT}_c$, define:

$$\gamma_{\pi,\mathtt{s}}^c = \mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[c] \qquad\qquad \delta_{\pi,\mathtt{t}}^c = \mathrm{tl}_{\mathtt{t}}^{\mathfrak{A}}[c]. \qquad (8.20)$$

That is, $\gamma_{\pi,\mathtt{s}}^c$ is the $\mathtt{s}$-spectrum in $\mathfrak{A}$ of the database element named c, and $\delta_{\pi,\mathtt{t}}^c$ is its $\mathtt{t}$-tally. As a special case of (8.1)–(8.3), for all bit-strings $\mathtt{s}$, $\mathtt{t}$ with $\mathtt{s}0 \in \mathrm{CCS}_c$ and $\mathtt{t}0 \in \mathrm{CCT}_c$:

$$\gamma_{\pi,\epsilon}^c + \delta_{\pi,\epsilon}^c = \boldsymbol{C} \qquad (8.21)$$

$$\gamma_{\pi,\mathtt{s}0}^c + \gamma_{\pi,\mathtt{s}1}^c = \gamma_{\pi,\mathtt{s}}^c \qquad (8.22)$$

$$\delta_{\pi,\mathtt{t}0}^c + \delta_{\pi,\mathtt{t}1}^c = \delta_{\pi,\mathtt{t}}^c. \qquad (8.23)$$

A moment's thought shows that Δ imposes additional constraints on the quantities $\gamma_{\pi,\mathtt{s}}^c$, $\delta_{\pi,\mathtt{t}}^c$. For consider an individual constant c and bit-string $\mathtt{s}$ of length p, and suppose there exists an individual constant d such that $\mathrm{tp}_{\Delta}[c,d] = \lambda \in \Lambda_{\pi,\mathtt{s}}$. (Since $\mathfrak{A}$ is chromatic, there can be at most one such d.) Then $\gamma_{\pi,\mathtt{s}}^c = \mathrm{sp}_{\mathtt{s}}^{\mathfrak{A}}[c] = \boldsymbol{C}_\lambda$. Similarly, if, for an individual constant c and bit-string $\mathtt{t}$ of length q, where $\mu = \mu_{\pi,\mathtt{t}}$ is a non-invertible message-type, there exist k individual constants d, distinct from c, with $\mathrm{tp}_{\Delta}[c,d] = \mu_{\pi,\mathtt{t}}$, then, writing μ for $\mu_{\pi,\mathtt{t}}$, we have $\delta_{\pi,\mathtt{t}}^c = \mathrm{tl}_{\mathtt{t}}^{\mathfrak{A}}[c] \geq k \cdot \boldsymbol{C}_\mu$. As we might say, Δ has to be *compatible* with the systems $\gamma_{\pi,\mathtt{s}}^c$ and $\delta_{\pi,\mathtt{t}}^c$, in the obvious sense of not requiring that c sends more messages than these vectors allow.

We need one further group of constraints. For each invertible type λ, let η_λ be the number of elements in Δ that send an invertible message of type λ to another database element. Of course, these numbers can be read directly from Δ. Now let $\mathcal{E}_4$ be the following classes of constraints, where λ varies as usual, π ranges over the set of 1-types realized in the database, c ranges over the set of individual constants, $\mathtt{s}$ ranges over the set of strings $\mathtt{s}$ such that $\mathtt{s}0 \in \mathrm{CCS}_c$ and $\mathtt{t}$ over the set of strings $\mathtt{t}$ such that $\mathtt{t}0 \in \mathrm{CCT}_c$:

$$x_\lambda \geq \eta_\lambda \qquad (8.24)$$

$$y_{\pi,\epsilon,\boldsymbol{u}} \geq 1, \quad \text{when } \boldsymbol{u} = \gamma_{\pi,\epsilon}^c \text{ and } \pi = \mathrm{tp}_{\Delta}[c] \qquad (8.25)$$

$$z_{\pi,\epsilon,\boldsymbol{u}} \geq 1, \quad \text{when } \boldsymbol{u} = \delta_{\pi,\epsilon}^c \text{ and } \pi = \mathrm{tp}_{\Delta}[c] \qquad (8.26)$$

$$\hat{y}_{\pi,\mathtt{s},\boldsymbol{v},\boldsymbol{w}} \geq 1, \quad \text{when } \boldsymbol{v} = \gamma_{\pi,\mathtt{s}0}^c, \boldsymbol{w} = \gamma_{\pi,\mathtt{s}1}^c, \text{ and } \pi = \mathrm{tp}_{\Delta}[c] \qquad (8.27)$$

$$\hat{z}_{\pi,\mathtt{t},\boldsymbol{v},\boldsymbol{w}} \geq 1, \quad \text{when } \boldsymbol{v} = \delta_{\pi,\mathtt{t}0}^c, \boldsymbol{w} = \delta_{\pi,\mathtt{t}1}^c, \text{ and } \pi = \mathrm{tp}_{\Delta}[c] \qquad (8.28)$$

Lemma 8.6. *Let $\mathfrak{A}$ be a finite model of $\varphi \wedge \Delta$. The constraints $\mathcal{E}_4$ are satisfied when the variables take the values specified in Table 1.*

Proof. The constraints (8.24), (8.25) and (8.26) are evident. For (8.27), let $c \in D$ be a constant, with 1-type $\pi = \mathrm{tp}_{\Delta}[c]$. Pick any $\mathtt{s}$ such that $\mathtt{s}0 \in \mathrm{CCS}_c$. (Hence, also, $\mathtt{s}1, \mathtt{s} \in \mathrm{CCS}_c$.) By definition, $\gamma_{\pi,\mathtt{s}0}^c$ and $\gamma_{\pi,\mathtt{s}1}^c$ are the $\mathtt{s}0$- and $\mathtt{s}1$-spectrum (respectively) of the (database) element $c \in A$. But then, referring to Table 1, c is one of the elements 'recorded' by the variable $\hat{y}_{\pi,\mathtt{s},\boldsymbol{v},\boldsymbol{w}}$, for $\boldsymbol{v} = \gamma_{\pi,\mathtt{s}0}^c$ and $\boldsymbol{w} = \gamma_{\pi,\mathtt{s}1}^c$. Consequently, $\hat{y}_{\pi,\mathtt{s},\boldsymbol{v},\boldsymbol{w}} \geq 1$, when $\boldsymbol{v} = \gamma_{\pi,\mathtt{s}0}^c$ and $\boldsymbol{w} = \gamma_{\pi,\mathtt{s}1}^c$. Thus, the constraints (8.27) are satisfied. Similarly for (8.28). $\qquad \square$

Let $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2 \cup \mathcal{E}_3 \cup \mathcal{E}_4$. Note that the size $\|\mathcal{E}\|$ of $\mathcal{E}$ is bounded above by an exponential function of $\|\varphi\|$.

Lemma 8.7. *Let φ and Δ be as above. If $\varphi \wedge \Delta$ is finitely satisfiable, then $\mathcal{E}$ has a solution over $\mathbb{N}$.*

Proof. Lemmas 8.3–8.6. $\qquad\square$

9. Obtaining a Model from the Solutions

In the previous section, we established a system of constants $\gamma^c_{\pi,\mathbf{s}}$, $\delta^c_{\pi,\mathbf{t}}$ and η_λ, and constructed a system $\mathcal{E}$ of linear equations and inequalities featuring these constants. We showed that, if $\varphi \wedge \Delta$ has a finite model $\mathfrak{A}$, then, by interpreting the constants $\gamma^c_{\pi,\mathbf{s}}$, $\delta^c_{\pi,\mathbf{t}}$ and η_λ, with respect to Δ and $\mathfrak{A}$ as suggested, we find that the $\gamma^c_{\pi,\mathbf{s}}$, $\delta^c_{\pi,\mathbf{t}}$ satisfy (8.21)–(8.23), and, moreover, that $\mathcal{E}$ has a solution over $\mathbb{N}$. In this section, we establish a converse result. Assuming that the constants $\gamma^c_{\pi,\mathbf{s}}$ and $\delta^c_{\pi,\mathbf{t}}$ satisfy (8.21)–(8.23), and that the $\gamma^c_{\pi,\mathbf{s}}$, $\delta^c_{\pi,\mathbf{t}}$ and η_λ are compatible with Δ as described, we construct, from any solution of $\mathcal{E}$ over $\mathbb{N}$, a finite model of $\varphi \wedge \Delta$.

Our strategy is as follows. We start with sets of elements A_π of the right cardinality, for each 1-type π, and gradually build the message-types that those elements 'want' to send. Fix some 1-type π and let A_π be a set with cardinality

$$|A_\pi| = \sum \{y_{\pi,\epsilon,\boldsymbol{u}'} \mid \boldsymbol{u}' \leq \boldsymbol{C}\}.$$

Think of A_π as the set of elements that 'want' to have 1-type π. We shall define functions $\mathbf{f}_{\pi,\mathbf{s}}$ and $\mathbf{g}_{\pi,\mathbf{t}}$ that give us the spectra and tallies for each element $a \in A_\pi$. Think of $\mathbf{f}_{\pi,\mathbf{s}}(a)$ as the $\mathbf{s}$-spectrum that a 'wants' to have and $\mathbf{g}_{\pi,\mathbf{t}}(a)$ as the $\mathbf{t}$-tally that a 'wants' to have (when a model is eventually built). For those functions to agree with the solutions of the previous system of constraints, we shall ensure that

$$|\mathbf{f}^{-1}_{\pi,\mathbf{s}}(\boldsymbol{u})| = y_{\pi,\mathbf{s},\boldsymbol{u}} \tag{9.1}$$

$$|\mathbf{g}^{-1}_{\pi,\mathbf{t}}(\boldsymbol{u})| = z_{\pi,\mathbf{t},\boldsymbol{u}}. \tag{9.2}$$

Furthermore, we shall ensure that, for all $a \in A_\pi$,

$$\mathbf{f}_{\pi,\epsilon}(a) + \mathbf{g}_{\pi,\epsilon}(a) = \boldsymbol{C} \tag{9.3}$$

$$\mathbf{f}_{\pi,\mathbf{s}0}(a) + \mathbf{f}_{\pi,\mathbf{s}1}(a) = \mathbf{f}_{\pi,\mathbf{s}}(a) \tag{9.4}$$

$$\mathbf{g}_{\pi,\mathbf{t}0}(a) + \mathbf{g}_{\pi,\mathbf{t}1}(a) = \mathbf{g}_{\pi,\mathbf{t}}(a). \tag{9.5}$$

Finally, for each individual constant c, setting $\pi = \mathrm{tp}_\Delta[c]$, we shall ensure the existence of an element $b_c \in A_\pi$ such that, for all $\mathbf{s} \in \mathrm{CCS}_c$ and all $\mathbf{t} \in \mathrm{CCT}_c$,

$$\mathbf{f}_{\pi,\mathbf{s}}(b_c) = \gamma^c_{\pi,\mathbf{s}} \tag{9.6}$$

$$\mathbf{g}_{\pi,\mathbf{t}}(b_c) = \delta^c_{\pi,\mathbf{t}}. \tag{9.7}$$

The idea is that such an element b_c can be used to realize the constant c, when a model is eventually built.

The following lemma guarantees that the above requirements can be satisfied.

Lemma 9.1. *Suppose that x_λ, $y_{\pi,s,\boldsymbol{u}}$, $z_{\pi,t,\boldsymbol{u}}$, $\hat{y}_{\pi,s,\boldsymbol{v},\boldsymbol{w}}$, $\hat{z}_{\pi,t,\boldsymbol{v},\boldsymbol{w}}$ are (classes of) natural numbers satisfying the constraints $\mathcal{E}$. Fix any 1-type $\pi \in \Pi$, and let A_π be a set of cardinality $\sum\{y_{\pi,\epsilon,\boldsymbol{u}'} \mid \boldsymbol{u}' \leq \boldsymbol{C}\}$. Then there exists a system of functions on A_π*

$$\mathbf{f}_{\pi,s} : A_\pi \to \{\boldsymbol{u} \mid \boldsymbol{u} \leq \boldsymbol{C}\} \quad and \quad \mathbf{g}_{\pi,t} : A_\pi \to \{\boldsymbol{u} \mid \boldsymbol{u} \leq \boldsymbol{C}\},$$

for each bit-string s with $|s| \leq p$ and t with $|t| \leq q$, satisfying the following conditions:

(i) *Equations (9.1) and (9.2) hold for all vectors $\boldsymbol{u} \leq \boldsymbol{C}$;*

(ii) *Equations (9.3)–(9.5) hold for all $a \in A_\pi$ and all s, t with $|s| < p$ and $|t| < q$.*

Furthermore, if $\pi = \mathrm{tp}_\Delta[c]$ for some individual constant c, then there exists in addition an element $b_c \in A_\pi$ such that

(iii) *Equations (9.6)–(9.7) hold for all $s \in CCS_c$ and $t \in CCT_c$.*

Proof. Decompose A_π into pairwise disjoint sets $A_{\boldsymbol{u}}$ of cardinality $|A_{\boldsymbol{u}}| = y_{\pi,\epsilon,\boldsymbol{u}}$, for each vector $\boldsymbol{u} \leq \boldsymbol{C}$. Note that, since $y_{\pi,\epsilon,\boldsymbol{u}}$ might be zero, some of these sets may be empty.

We construct the functions $\mathbf{f}_{\pi,s}$, where $0 < |s| \leq p$ by induction on s. Suppose $s = \epsilon$; for each $\boldsymbol{u} \leq \boldsymbol{C}$, and for all $a \in A_{\boldsymbol{u}}$, set $\mathbf{f}_{\pi,\epsilon}(a) = \boldsymbol{u}$ and $\mathbf{g}_{\pi,\epsilon}(a) = \boldsymbol{C} - \boldsymbol{u}$. These assignments clearly satisfy (9.1) and (9.2), keeping in mind the constraints (8.4). Now suppose $\pi = \mathrm{tp}_\Delta[c]$, for some individual constant c. By (8.25), and setting $\boldsymbol{u}$ to be the vector $\gamma_{\pi,\epsilon}^c$, we have $A_{\boldsymbol{u}} \neq \emptyset$. Choose b_c to be any element in $A_{\boldsymbol{u}}$. This immediately secures (9.6) for $s = \epsilon$, and given the condition (8.21) relating $\gamma_{\pi,\epsilon}^c$ and $\delta_{\pi,\epsilon}^c$, also (9.7).

Now, assume that $\mathbf{f}_{\pi,s}$ ($0 \leq |s| < p$) has been defined and satisfies (9.1) and (9.6). For every vector $\boldsymbol{u} \leq \boldsymbol{C}$, decompose the set $\mathbf{f}_{\pi,s}^{-1}(\boldsymbol{u})$ into subsets $A_{\boldsymbol{v},\boldsymbol{w}}$ with cardinality $|A_{\boldsymbol{v},\boldsymbol{w}}| = \hat{y}_{\pi,s,\boldsymbol{v},\boldsymbol{w}}$, where $\boldsymbol{v}$, $\boldsymbol{w}$ range over all vectors $\leq \boldsymbol{C}$ such that $\boldsymbol{v} + \boldsymbol{w} = \boldsymbol{u}$. This is possible from the constraints (8.5) and Equation (9.1). Set

$$\mathbf{f}_{\pi,s0}(a) = \boldsymbol{v} \quad and \quad \mathbf{f}_{\pi,s1}(a) = \boldsymbol{w},$$

for all $a \in A_{\boldsymbol{v},\boldsymbol{w}}$. Notice that Equation (9.4) holds as required.

To see that $\mathbf{f}_{\pi,s0}$ and $\mathbf{f}_{\pi,s1}$ both satisfy Equation (9.1), note that $\mathbf{f}_{\pi,s0}(a) = \boldsymbol{v}$ if and only if, for some vector $\boldsymbol{w}'$ such that $\boldsymbol{v} + \boldsymbol{w}' \leq \boldsymbol{C}$, $a \in A_{\boldsymbol{v},\boldsymbol{w}'}$. Similarly, $\mathbf{f}_{\pi,s1}(a) = \boldsymbol{w}$ if and only if, for some vector $\boldsymbol{v}'$ such that $\boldsymbol{v}' + \boldsymbol{w} \leq \boldsymbol{C}$, $a \in A_{\boldsymbol{v}',\boldsymbol{w}}$. That is,

$$\mathbf{f}_{\pi,s0}^{-1}(\boldsymbol{v}) = \bigcup\{A_{\boldsymbol{v},\boldsymbol{w}'} \mid \boldsymbol{v} + \boldsymbol{w}' \leq \boldsymbol{C}\}$$

$$\mathbf{f}_{\pi,s1}^{-1}(\boldsymbol{w}) = \bigcup\{A_{\boldsymbol{v}',\boldsymbol{w}} \mid \boldsymbol{v}' + \boldsymbol{w} \leq \boldsymbol{C}\},$$

with the collections of sets on the respective right-hand sides being pairwise disjoint. By the constraints (8.7)–(8.8), together with the fact that $|A_{\boldsymbol{v},\boldsymbol{w}}| = \hat{y}_{\pi,s,\boldsymbol{v},\boldsymbol{w}}$ for all $\boldsymbol{v}$, $\boldsymbol{w}$, we have:

$$|\mathbf{f}_{\pi,s0}^{-1}(\boldsymbol{v})| = y_{\pi,s0,\boldsymbol{v}}$$

$$|\mathbf{f}_{\pi,s1}^{-1}(\boldsymbol{w})| = y_{\pi,s1,\boldsymbol{w}},$$

which establishes (9.1) for the functions $\mathbf{f}_{\pi,s0}$ and $\mathbf{f}_{\pi,s1}$.

It remains only to secure (9.6) for the functions $\mathbf{f}_{\pi,s0}$ and $\mathbf{f}_{\pi,s1}$ in the case where $\pi = \mathrm{tp}_\Delta[c]$, for some individual constant c, and $s0$ (and hence $s1$ and s) is in CCS_c. By inductive hypothesis, $\mathbf{f}_{\pi,s}(b_c) = \gamma_{\pi,s}^c$, i.e., $b_c \in \mathbf{f}_{\pi,s}^{-1}(\boldsymbol{u})$ where $\boldsymbol{u}$ is the vector $\gamma_{\pi,s}^c$. Let $\boldsymbol{v} = \gamma_{\pi,s0}^c$ and $\boldsymbol{w} = \gamma_{\pi,s1}^c$. Then, by (8.22), these vectors satisfy the condition $\boldsymbol{v} + \boldsymbol{w} = \boldsymbol{u}$, so that, in the decomposition of $\mathbf{f}_{\pi,s}^{-1}(\boldsymbol{u})$, the set $A_{\boldsymbol{v},\boldsymbol{w}}$ will have cardinality $\hat{y}_{\pi,\boldsymbol{v},\boldsymbol{w}}$. Thus, by (8.27), $A_{\boldsymbol{v},\boldsymbol{w}} \neq \emptyset$, so that we may ensure that b_c is contained in this set when we perform the decomposition of $\mathbf{f}_{\pi,s}^{-1}(\boldsymbol{u})$. But then we will have set $\mathbf{f}_{\pi,s0}(b_c) = \boldsymbol{v} = \gamma_{\pi,s0}^c$ and

$\mathbf{f}_{\pi,\mathbf{s1}}(b_c) = \boldsymbol{w} = \gamma^c_{\pi,\mathbf{s1}}$, so that (9.6) holds with $\mathbf{s}$ replaced by $\mathbf{s0}$ and $\mathbf{s1}$. This completes the induction.

The construction of the functions $\mathbf{g}_{\pi,\mathbf{t}}$ is completely analogous. $\qquad\square$

Lemma 9.2 [PH07, Lemma 12]. *Let the functions $\mathbf{f}$ and $\mathbf{g}$ be constructed as in Lemma 9.1. Then, for all $a \in A_\pi$, we have*

$$\sum\{\mathbf{f}_{\pi,s'}(a) : |s'| = p\} + \sum\{\mathbf{g}_{\pi,t'}(a) : |t'| = q\} = \boldsymbol{C}.$$

Proof. We prove the stronger result that, for all $a \in A_\pi$, j $(0 \le j \le p)$ and k $(0 \le k \le q)$,

$$\sum\{\mathbf{f}_{\pi,s'}(a) : |s'| = j\} + \sum\{\mathbf{g}_{\pi,t'}(a) : |t'| = k\} \quad = \quad \boldsymbol{C}, \qquad (9.8)$$

using a double induction on j and k. If $j = k = 0$, then the left-hand side of (9.8) is simply $\mathbf{f}_{\pi,\epsilon}(a) + \mathbf{g}_{\pi,\epsilon}(a)$, which is equal to $\boldsymbol{C}$ by (9.3). Suppose now that the result holds for the pair j, k, with $j < p$. Then

$$\sum\{\mathbf{f}_{\pi,s'}(a) : |s'| = (j+1)\} + \sum\{\mathbf{g}_{\pi,t'}(a) : |t'| = k\}$$
$$= \sum\{\mathbf{f}_{\pi,s'0}(a) + \mathbf{f}_{\pi,s'1}(a) : |s'| = j\} + \sum\{\mathbf{g}_{\pi,t'}(a) : |t'| = k\}$$
$$= \sum\{\mathbf{f}_{\pi,s'}(a) : |s'| = j\} + \sum\{\mathbf{g}_{\pi,t'}(a) : |t'| = k\} \qquad \text{by (9.4)}$$
$$= \boldsymbol{C} \qquad \text{by inductive hypothesis.}$$

This establishes the result for the pair $j + 1, k$. An analogous argument using (9.5) applies when $k < m$, completing the induction. $\qquad\square$

We are now ready to prove the converse of Lemma 8.7.

Lemma 9.3. *Let Δ, φ and $\mathcal{E}$ be as given above. If $\mathcal{E}$ has a solution over $\mathbb{N}$, then $\varphi \wedge \Delta$ is finitely satisfiable.*

Proof. Let x_λ, $y_{\pi,\mathbf{s},\boldsymbol{u}}$, $z_{\pi,\mathbf{t},\boldsymbol{u}}$, $\hat{y}_{\pi,\mathbf{s},\boldsymbol{v},\boldsymbol{w}}$, $\hat{z}_{\pi,\mathbf{t},\boldsymbol{v},\boldsymbol{w}}$ be natural numbers satisfying $\mathcal{E}$ (with the indices π, $\mathbf{s}$, $\mathbf{t}$, $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$ varying as usual). Notice that, for all positive integers k, the (sets of) natural numbers kx_λ, $ky_{\pi,\mathbf{s},\boldsymbol{u}}$, $kz_{\pi,\mathbf{t},\boldsymbol{u}}$, $k\hat{y}_{\pi,\mathbf{s},\boldsymbol{v},\boldsymbol{w}}$, $k\hat{z}_{\pi,\mathbf{t},\boldsymbol{v},\boldsymbol{w}}$ also satisfy $\mathcal{E}$. Thus, we may assume that all values in the sought-after solution are either 0 or $\ge 3mC$.

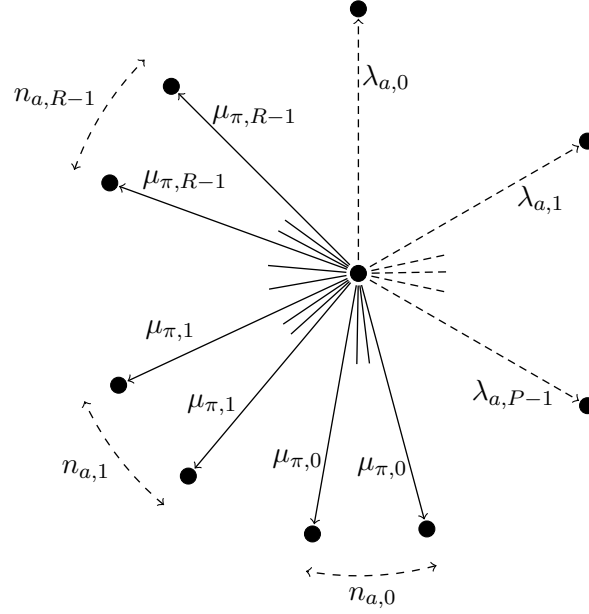We start by defining the universe A of our model $\mathfrak{A}$. Let

$$A = \bigcup\{A_\pi \mid \pi \text{ is any 1-type over } \sigma\},$$

where each set A_π has cardinality

$$|A_\pi| = \sum\{y_{\pi,\epsilon,\boldsymbol{u}'} \mid \boldsymbol{u}' \le \boldsymbol{C}\}$$

and the sets A_π are pairwise disjoint. Think of A_π as the elements of A that 'want' to have 1-type π. Note that $A \ne \emptyset$, by the constraint (8.11).

Let the functions $\mathbf{f}_{\pi,\mathbf{s}}$, $\mathbf{g}_{\pi,\mathbf{t}}$ and elements b_c be as constructed in Lemma 9.1. Think of $\mathbf{f}_{\pi,\mathbf{s}}(a)$ as the $\mathbf{s}$-spectrum that a 'wants' to have, $\mathbf{g}_{\pi,\mathbf{t}}(a)$ as the $\mathbf{t}$-tally that a 'wants' to have, and b_c the database element that 'wants' to be the denotation of c. We are only interested in the values of these functions when $|\mathbf{s}| = p$ and $|\mathbf{t}| = q$. Fix some such $\mathbf{s}$. Decompose each A_π into sets $\mathbf{f}_{\pi,\mathbf{s}}^{-1}(\boldsymbol{u})$ (with $\boldsymbol{u}$ varying), for each vector $\boldsymbol{u}$ with $\boldsymbol{0} < \boldsymbol{u} \le \boldsymbol{C}$. By the constraints (8.12) and Equation (9.1), decompose each of those $\mathbf{f}_{\pi,\mathbf{s}}^{-1}(\boldsymbol{u})$ into pairwise disjoint (possibly empty) sets A_λ with $|A_\lambda| = x_\lambda$, for all invertible message-types $\lambda \in \Lambda_{\pi,\mathbf{s}}$ with $C_\lambda = \boldsymbol{u}$. When performing this decomposition, for each individual constant c, if Δ specifies

FIGURE 9. The messages sent by $a \in A_\pi$.

that c sends an invertible message of type λ to some other database element, then make sure that b_c is assigned to the set A_λ: by (8.24), this set is at least as numerous as the number of database elements sending a message of type λ, so that we never run out of elements. Think of A_λ as the set of elements in A_π that 'want' to send a single invertible message of type λ. The above process is repeated for all possible different values of $\mathbf{s}$ (with $|\mathbf{s}| = p$), and each decomposition should be thought of as independent of each other. Analogously, for each $\mathbf{t}$ with $|\mathbf{t}| = q$, A_π is decomposed into pairwise disjoint sets $\mathbf{g}_{\pi,\mathbf{t}}^{-1}(\boldsymbol{u})$ (with $\boldsymbol{u}$ varying); and, again, those decompositions should be thought of as independent of each other.

Based on the above decompositions, we specify for each $a \in A_\pi$ a 'mosaic piece' and show how to assemble these pieces into a model of φ. A mosaic piece is, informally, a collection of the messages that a 'wants' to send. This collection might contain more than one message of each type (or zero for that matter). Let $a \in A_\pi$; the mosaic piece corresponding to a contains:

(i) a single message labelled $\lambda_{a,\mathbf{s}}$ for each bit-string $\mathbf{s}$ with $|\mathbf{s}| = p$ if $\mathbf{f}_{\pi,\mathbf{s}}(a) \neq \mathbf{0}$, where $\lambda_{a,\mathbf{s}}$ is the (unique) 2-type λ for which $a \in A_\lambda$;

(ii) $n_{a,\mathbf{t}}$ messages labelled $\mu_{\pi,\mathbf{t}}$ for each bit-string $\mathbf{t}$ with $|\mathbf{t}| = q$, where $n_{a,\mathbf{t}}$ is the (unique) natural number such that $\mathbf{g}_{\pi,\mathbf{t}}(a) = n_{a,\mathbf{t}} \cdot \boldsymbol{C}_{\mu_{\pi,\mathbf{t}}}$. Note that if $\mathbf{g}_{\pi,\mathbf{t}}(a) = 0$ then $n_{a,\mathbf{t}} = 0$, otherwise $n_{a,\mathbf{t}}$ exists by the constraints (8.17) and Equation (9.2).

The mosaic piece corresponding to a is depicted in Fig. 9. Here, for legibility, we have identified the strings $\mathbf{s}$ of length p with the numbers $0, \ldots, P-1$, and the strings $\mathbf{t}$ of length q for which $\mu_{\pi,\mathbf{t}}$ is a (non-invertible) message-type with the numbers $0, \ldots, R-1$, for suitable P, R. The dashed radial arrows indicate that a may or may not send a message of invertible type $\lambda_{a,\mathbf{s}} \in \Lambda_{\pi,\mathbf{s}}$; the solid radial arrows indicate that a sends $n_{a,\mathbf{t}}$ (possibly zero) messages of non-invertible type $\mu_{\pi,\mathbf{t}}$.

Let $a \in A$ and define $\boldsymbol{C}_a$ to be the vector $(C_{a,1}, \ldots, C_{a,m})$ whose ith coordinate $C_{a,i}$ records the number of messages in the mosaic piece of a containing an outgoing o_i arrow— i.e. messages having label ν for which $o_i(x, y) \in \nu$. Clearly (see Fig. 9),

$$\boldsymbol{C}_a = \sum \{\mathbf{f}_{\pi,\mathbf{s}'}(a) : |\mathbf{s}'| = p\} + \sum \{\mathbf{g}_{\pi,\mathbf{t}'} : |\mathbf{t}'| = q\}$$

and, by Lemma 9.2,

$$\boldsymbol{C}_a = \boldsymbol{C}. \tag{9.9}$$

We now build $\mathfrak{A}$ in four steps as follows.

Step 1 (Fixing the 1-types)   For all 1-types π and all $a \in A_\pi$, set $\mathrm{tp}_{\mathfrak{A}}[a] = \pi$. Since 1-types do not contain equality literals such as $x = c$ or $x \neq d$, this is meaningful. Moreover, since the sets A_π are pairwise disjoint, no clashes arise. For each individual constant c, let $c^{\mathfrak{A}} = b_c$. Since Δ contains naming formulas $c(c)$ and $\neg c(d)$, where c and d are distinct, it is obvious that the unique names assumption is respected: that is, $c^{\mathfrak{A}} \neq d^{\mathfrak{A}}$ for c and d distinct. Therefore, we may, as before, write c instead of the more correct $c^{\mathfrak{A}}$, since no confusion arises.

Step 2 (Fixing the invertible message-types)   We first assign the invertible message-types for all pairs $c, d \in A$, where c and d are distinct individual constants, as dictated by Δ. To see that this is possible, suppose $\Delta \models \lambda(c, d)$, where $\lambda \in \Lambda_{\pi,\mathbf{s}}$, say. Then by Lemma 9.1, c will 'want' to have $\mathbf{s}$-spectrum $\gamma^c_{\pi,\mathbf{s}}$, which, by the assumed compatibility of Δ with these constants, implies that $\gamma^c_{\pi,\mathbf{s}} = \boldsymbol{C}_\lambda$. That is: the mosaic piece for c will send an invertible message of type λ. Similarly, the mosaic piece for d will send an invertible message of type λ^{-1}; and these two messages may be paired up with each other. We then put, by the constraints (8.13), all other λ-labelled messages and all λ^{-1}-labelled messages in one-to-one correspondence, for each invertible message of type λ. Thus, if a sends a λ-labelled message and b 'wants to receive it' (i.e. sends a λ^{-1}-labelled message), we set $\mathrm{tp}_{\mathfrak{A}}[a, b] = \lambda$. To ensure that each assignment $\mathrm{tp}_{\mathfrak{A}}[a, b]$ $(a, b \in A)$ is valid, we need only check that a and b are distinct. But, since $x_\lambda > 0$, by the constraints (8.14) we must have $\mathrm{tp}_{\mathfrak{A}}[a] \neq \mathrm{tp}_{\mathfrak{A}}[b]$ hence, by construction, a and b belong to the disjoint sets $A_{\mathrm{tp}_{\mathfrak{A}}[a]}$ and $A_{\mathrm{tp}_{\mathfrak{A}}[b]}$. Moreover, since every element sends at most one invertible message of each type, no conflicts with the present assignment will arise in future assignments.

Step 3 (Fixing the non-invertible message-types)   Start by decomposing each non-empty set A_π into three pairwise disjoint (possibly empty) sets $A_{\pi,0}$, $A_{\pi,1}$ and $A_{\pi,2}$ having at least mC elements each, since $|A_\pi| \geq 3mC$, and with the following restriction: if $c^{\mathfrak{A}} \in A_\pi$, for some constant $c \in D$ (i.e. $\pi = \mathrm{tp}_\Delta[c]$), pick these three sets such that $c^{\mathfrak{A}} \in A_{\pi,0}$. This is possible by our choise of solution of $\mathcal{E}$.

Let $\mu_{\pi,\mathbf{t}}$ be any non-invertible message type, with $\pi = \mathrm{tp}_1(\mu_{\pi,\mathbf{t}})$ and $\rho = \mathrm{tp}_2(\mu_{\pi,\mathbf{t}})$ being its starting and terminal 1-types. Let $a \in A$ be an element that sends $n_{a,\mathbf{t}} > 0$ messages of type $\mu_{\pi,\mathbf{t}}$. Clearly, then, $a \in A_\pi$ and there is a vector $\boldsymbol{u} > \boldsymbol{0}$ such that $\mathbf{g}_{\pi,\mathbf{t}}(a) = \boldsymbol{u}$, hence $\mathbf{g}_{\pi,\mathbf{t}}^{-1}(\boldsymbol{u})$ is non-empty. As a result, $z_{\pi,\mathbf{t},\boldsymbol{u}} = |\mathbf{g}_{\pi,\mathbf{t}}^{-1}(\boldsymbol{u})|$ is positive, thus, by the constraints (8.18), $\sum \{y_{\rho,\epsilon,\boldsymbol{u}'} \mid \boldsymbol{u}' \leq \boldsymbol{C}\}$ is also positive. This implies that A_ρ is non-empty since, clearly, $|A_\rho| = \sum \{y_{\rho,\epsilon,\boldsymbol{u}'} \mid \boldsymbol{u}' \leq \boldsymbol{C}\}$ and hence has been partitioned into three sets $A_{\rho,0}$, $A_{\rho,1}$ and $A_{\rho,2}$ having at least mC elements each.

Let us assume for the moment that a is not equal to the interpretation of any constant. We employ the standard 'circular witnessing' technique of [GKV97]. The non-empty set A_π

has been partitioned into $A_{\pi,0}$, $A_{\pi,1}$ and $A_{\pi,2}$; since $a \in A_\pi$, let j be such that $a \in A_{\pi,j}$, $0 \leq j \leq 2$. Now, let $k = j + 1 \, (\mathrm{mod} \ 3)$ and select $n_{a,\mathbf{t}}$ elements b from $A_{\rho,k}$ that have not already been chosen to receive any messages (invertible or non-invertible) and set, for each one of those, $\mathrm{tp}_\mathfrak{A}[a,b] = \mu_{\pi,\mathbf{t}}$. Note that there are enough elements in $A_{\rho,k}$ to choose from, as a can send at most mC messages (of invertible or non-invertible type). Suppose, on the other hand, a is (the interpretation of) some constant, say c, and that Δ specifies that c sends messages of type μ to k other database elements. Then by Lemma 9.1, c will 'want' to have $\mathbf{t}$-spectrum $\delta^c_{\pi,\mathbf{t}}$, which, by the assumed compatibility of Δ with these constants, implies that $\delta^c_{\pi,\mathbf{t}} \geq k \cdot \boldsymbol{C}_\mu$. That is: the mosaic piece for c will send at least k non-invertible messages of type λ. Now set $\mathrm{tp}_\mathfrak{A}[c,d] = \mu_{\pi,\mathbf{t}}$ for each of the k elements as required by Δ, subtract k from the value $n_{a,\mathbf{t}}$ to take account of the fact that these non-invertible message of type $\mu_{\pi,\mathbf{t}}$ have been dealt with, and proceed as before. Since μ is not an invertible 2-type, we can be sure that it has not already been set during Step 2. Moreover, 'circular witnessing' ensures that no clashes arise during Step 3.

Step 4 (Fixing the remaining 2-types)   If c and d are distinct individual constants for which $\mathrm{tp}_\mathfrak{A}[c,d]$ has not been defined, set $\mathrm{tp}_\mathfrak{A}[c,d] = \mathrm{tp}_\Delta[c,d]$; of course, this must a silent type, since all messages sent within the database have been accounted for. If $\mathrm{tp}_\mathfrak{A}[a,b]$ has not yet been defined, set it to be the 2-type

$$\pi \cup \rho[y/x] \cup \{\neg e \mid e \text{ is a guard-atom not involving } =\},$$

where $\pi = \mathrm{tp}_\mathfrak{A}[a]$, $\rho = \mathrm{tp}_\mathfrak{A}[b]$ and $\rho[y/x]$ is the result of replacing x by y in ρ. Note that, since $C_1, \ldots, C_m$ are by assumption positive integers, a and b certainly send some messages and, thus, the constraints (8.15) and (8.16) ensure that both $\alpha \wedge \pi$ and $\alpha \wedge \rho$ are satisfiable. Note also that this is the point in the proof where we make essential use of of the fact that φ is guarded. In particular, the conjuncts $\forall x \forall y (e_j(x,y) \rightarrow (\ldots))$ in (7.1) are satisfied by the assignments in this step, because the antecedents are false.

This completes the definition of $\mathfrak{A}$ and we now show that $\mathfrak{A} \models \varphi \wedge \Delta$. Referring to the normal form in Lemma 7.1, notice that none of the above steps violates the conjuncts

$$\forall x \, \alpha \wedge \bigwedge_{1 \leq j \leq n} \forall x \forall y (e_j(x,y) \rightarrow (\beta_j \vee x = y)).$$

Furthermore, the conjuncts

$$\bigwedge_{1 \leq i \leq m} \forall x \exists_{=C_i} y (o_i(x,y) \wedge x \neq y)$$

are all satisfied taking into account Equation (9.9) and the fact that none of the 2-types assigned in Step 4 is a message type. The database Δ is evidently satisfied by the above construction. $\qquad \square$

Now, observe that all the constraints in $\mathcal{E}$ have the forms

$$
\begin{aligned}
x_1 + \ldots + x_n &= x, \\
x_1 + \ldots + x_n &\geq c, \\
x &= 0, \\
x_1 &= x_2, \\
x &\geq c, \\
x > 0 \Rightarrow x_1 + \ldots + x_n &> 0,
\end{aligned}
$$

where x, $x_1, \ldots x_n$ are variables and c is a constant. Recall that the size $\|\mathcal{E}\|$ of $\mathcal{E}$ is exponential in $|\varphi \wedge \Delta|$. Our goal is to find a solution of $\mathcal{E}$ in $\mathbb{N}$. The following lemma shows that we can transform a system of the above form into an integer programming problem which, in turn, can be regarded as a linear programming problem. This is important because linear programming is in PTIME, whereas integer programming is in NPTIME.

Lemma 9.4 ([Cal96]; see also [PH07], Lemma 15). *Let Δ, φ and $\mathcal{E}$ be as above. An algorithm exists to determine whether $\mathcal{E}$ has a solution over $\mathbb{N}$ in time bounded by a polynomial function of $\|\mathcal{E}\|$.*

We have now established the required upper complexity-bound for finite satisfiability in $\mathcal{GC}^2\mathcal{D}$.

Theorem 9.5. *The finite satisfiability problem in $\mathcal{GC}^2\mathcal{D}$ is in EXPTIME.*

Proof. Let a $\mathcal{GC}^2\mathcal{D}$-formula $\psi \wedge \Gamma$ and ψ be given. By Lemma 7.1, convert ψ to a formula φ of the form (7.1), and let σ be the signature of φ together with additional unary predicates: all the naming predicates for the individual constants in Δ and $\lceil \log((mC)^2 + 1) \rceil$ spare predicates. If Γ is inconsistent, fail; otherwise, add all the naming formulas to Γ, and guess a consistent completion $\Delta \supseteq \Gamma$. Having fixed φ, σ and Δ, compute, for each individual constant c, the sets of strings CCS_c and CCT_c, and then guess the vectors $\gamma^c_{\pi,\mathbf{s}}$, for $\mathbf{s} \in \mathrm{CCS}_c$ and $\delta^c_{\pi,\mathbf{t}}$, for $\mathbf{t} \in \mathrm{CCT}_c$. Check that these vectors satisfy the conditions (8.21)–(8.23), and that they are consistent with Δ, failing if not. Compute the numbers η_λ from Δ. Now write the system of equations $\mathcal{E}$. Determine whether $\mathcal{E}$ has a solution over $\mathbb{N}$. By Lemma 9.4, this can be determined in exponential time. If $\mathcal{E}$ has a solution for any of these guesses, then succeed; otherwise fail. By Lemmas 8.6 and 9.3, the above procedure has a successful run (for some guess) if and only if $\psi \wedge \Gamma$ has a finite model. All guesses involve only a polynomial amount of data, and so may be exhaustively searched in exponential time. $\square$

Having dealt with the finite satisfiability problem for $\mathcal{GC}^2$, we turn now to the corresponding satisfiability problem. We employ 'extended arithmetic', over the set $\mathbb{N} \cup \{\aleph_0\}$, where addition and multiplication are extended in the obvious way, i.e. $\aleph_0 + \aleph_0 = \aleph_0 \cdot \aleph_0 = \aleph_0$, $n + \aleph_0 = \aleph_0 + n = \aleph_0$, for all $n \in \mathbb{N}$, etc. After all the gruel we have just chomped our way through, Theorem 9.6 is dessert.

Theorem 9.6. *The satisfiability problem in $\mathcal{GC}^2\mathcal{D}$ is in EXPTIME.*

Sketch proof. We proceed exactly as in the finite case, except that we seek solutions to $\mathcal{E}$ over extended arithmetic. It is evident that, if $\boldsymbol{v}$ is a solution of $\mathcal{E}$, then so is $\aleph_0 \boldsymbol{v}$. Thus, we may confine attention to solutions over the 2-element set $\{0, \aleph_0\}$. Such a system is essentially Boolean, so its constraints can be viewed as formulas of propositional logic; and, with a little care, can be written as Horn clauses. (See [PH07] for more details.) This establishes membership in EXPTIME. $\square$

It is well known that the satisfiability and finite satisfiability problems for $\mathcal{GC}^2$ are EXPTIME-hard. Thus, all the complexity bounds given above are tight.

REFERENCES

[ANvB98] Hajnal Andréka, István Németi, and Johan van Benthem. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27(3):217–274, 1998.

[BP00] J. Biskup and Torsten Polle. Decomposition of database classes under path functional dependencies and onto constraints. In *Proc. Foundations of Information and Knowledge Systems, Proceedings of First International Symposium, (FoIKS)*, volume 1762 of *Lecture Notes in Computer Science*, pages 31–49. Springer, 2000.

[Cal96] Diego Calvanese. Unrestricted and finite model reasoning in class-based representation formalisms. PhD thesis, Dipartimento di Informatica e Sistemistica, Università di Roma La Sapienza. In *In Proc. of the 4th Int. Conf. on Deductive and Object-Oriented Databases (DOOD-95), number 1013 in Lecture Notes in Computer Science*. Citeseer, 1996.

[GKV97] Erich Grädel, Phokion G Kolaitis, and Moshe Y Vardi. On the decision problem for two-variable first-order logic. *Bulletin of symbolic logic*, 3(01):53–69, 1997.

[Grä99a] Erich Grädel. On the restraining power of guards. *Journal of Symbolic Logic*, pages 1719–1742, 1999.

[Grä99b] Erich Grädel. Why are modal logics so robustly decidable? In *Bulletin EATCS*. Citeseer, 1999.

[IW94] M. Ito and G. Weddell. Implication problems for functional constraints on databases supporting complex objects. *Journal of Computer and System Sciences*, 49(3):726–768, 1994.

[Kaz04] Yevgeny Kazakov. A polynomial translation from the two-variable guarded fragment with number restrictions to the guarded fragment. In *Logics in Artificial Intelligence*, pages 372–384. Springer, 2004.

[PH07] Ian Pratt-Hartmann. Complexity of the guarded two-variable fragment with counting quantifiers. *J. Log. Comput.*, 17(1):133–155, 2007.

[PH09] Ian Pratt-Hartmann. Data-complexity of the two-variable fragment with counting quantifiers. *Information and Computation*, 207(8):867–888, 2009.

[TW05a] David Toman and Grant Weddell. On path-functional dependencies as first-class citizens in description logicsion between inverse features and path-functional dependencies in description logics. In *Proc. Description Logics (DL)*, 2005.

[TW05b] David Toman and Grant Weddell. On the interaction between inverse features and path-functional dependencies in description logics. In *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, pages 603–608, 2005.

[TW08] David Toman and Grant Weddell. On keys and functional dependencies as first-class citizens in description logics. *Journal of Automated Reasoning*, 40(2–3):117–132, 2008.

[Var96] Moshe Y Vardi. Why is modal logic so robustly decidable? *Descriptive complexity and finite models*, 31:149–184, 1996.

[Wed89] G. Weddell. A theory of functional dependencies for object oriented data models. In *Proc. International Conference on Deductive and Object-Oriented Databases*, pages 165–184, 1989.