A. PROOF OF THEOREM 3.1(4)

**Theorem 3.1:** (4) RCDP(\(\mathcal{L}_Q, \mathcal{L}_C\)) is undecidable when \(\mathcal{L}_C\) is FP and \(\mathcal{L}_Q\) consists of a fixed query in FP even for fixed master data and containment constraints. □

**Proof.** Given a 2-head DFA \(A = (Q, \Sigma, \delta, q_0, q_{acc})\), we define schemas \(\mathcal{R}\) and \(\mathcal{R}_m\), a database instance \(D\) of \(\mathcal{R}\) and master data instance \(D_m\) of \(\mathcal{R}_m\), a set \(V\) of CCs expressed in \(\mathcal{L}_C\), and a query \(Q \in \mathcal{L}_Q\). We show that \(D\) is complete for \(Q\) relative to \((D_m, V)\) iff \(L(A)\) is nonempty.

(a) We define the relational schema \(\mathcal{R}\) to consist of the three relations \(P, \bar{P}\) and \(F\) described in the proof for case (3). Moreover, we add an extra 6-ary relation \(R_\Delta\) whose instances are to encode (a subset of) the transitions in \(\Delta\) of the DFA \(A\). We define \(\mathcal{R}_m = (R_m^m)\), where \(R_m^m\) is a unary relation.

(b) We define \(D = (I_1 = \emptyset, I_2 = \emptyset, I_3 = \emptyset, I_4 = \emptyset)\) and \(D_m = (I_1^m = \emptyset)\).

(c) The set \(V\) of CCs includes \(V_1, V_2\) and \(V_3\) given in the proof for case (3), which assure that instances of \(P, \bar{P}\) and \(F\) are well-formed. We additionally define a CC \(V_\Delta\), which assures that each instance \(I_\Delta\) of \(R_\Delta\) is a subset of \(\bigvee_{\Delta \in \Delta} \varphi_\Delta(q, x, y, q', x', y')\), where \(\varphi_\Delta\) is defined in the previous proof. We express this CC as follows:

\[
V_\Delta = \exists p, v, w, p', v', w' \exists f, f', g, g'(R_\Delta(p, v, w, p', v', w') \land F(v, f) \land F(w, g) \\
\land F(v', f') \land F(w', g') \land \bigwedge_{\Delta \in \Delta} \varphi_\Delta(p, v, w, p', v', w') \subseteq 0,
\]

where for a particular \(\Delta \in \Delta\) of the form \(\Delta = (q, \{n_1, n_2\}) \rightarrow (q', \{m_1, m_2\})\), the formula \(\varphi_\Delta(p, v, w, p', v', w')\) is given by

\[
\forall p \neq q \lor p' \neq q' \lor \alpha_1(v) \lor \beta_3(w) \lor \beta_2(w, w').
\]

Here \(\alpha_i(x) = F(x, x) \lor (\exists y(F(x, y) \land x \neq y)) \land \bar{P}(x)\) if \(n_i = 1\); \(\alpha_i(x) = F(x, x) \lor (\exists y(F(x, y) \land x \neq y)) \land P(x)\) if \(n_i = 0\); and \(\alpha_i(x) = \exists y(F(x, y) \land x \neq y)\) if \(n_i = \varepsilon\).

Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, the ACM copyright/server notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.

© 20YY ACM 0362-5915/20YY/0000-0000 $5.00
Moreover, \(\beta_i(x, y) = (x = y)\) if \(\text{move}_i = +1\) and \(\beta_i(x, y) = F(x, y)\) if \(\text{move}_i = 0\). Observe that \(V_\Delta\) can be expressed in FP although we did not use FP syntax.

(d) We define the query \(Q = \exists y_1 \exists y_2 [\text{TC}_{x, y, z, x', y', z'} \cdot Q_F(R_\Delta)] [q_0, 0, 0, q_{acc}, y_1, y_2] \land \exists F(0, x) \land \exists F(x, x), \) where \(Q_F(R_\Delta) = \exists f, f', g, g' (R_\Delta(p, v, w, p', v', w') \land F(v, f) \land F(w, g) \land F(v', f') \land F(w', g'))\). That is, it only considers elements with the proper domain range (i.e., elements in \(F\)). Note that \(Q\) is fixed.

We show that \(D\) is complete for \(Q\) relative to \((D_m, V)\) iff \(L(A) = \emptyset\). Indeed, suppose that \(L(A) = \emptyset\). Then, \(Q(D') = \emptyset\) for each \(D'\) such that \((D', D_m) \models V\). Hence \(D\) is complete. Conversely, suppose that \(L(A) \neq \emptyset\). Then there exists an instance \(D'\) on which \(Q(D') \neq \emptyset\). Hence \(D\) is not complete.

B. PROOF OF THEOREM 4.1(3) AND (4)

**Theorem 4.1:** \(\text{RCQP}(\mathcal{L}_Q, \mathcal{L}_C)\) is undecidable when (3) \(\mathcal{L}_Q\) is FP and \(\mathcal{L}_C\) consists of fixed queries in FP, or (4) \(\mathcal{L}_C\) is FP and \(\mathcal{L}_Q\) is CQ. When \(\mathcal{L}_Q\) is FP, it remains undecidable for fixed master data and fixed containment constraints.

**Proof.** The undecidability of both cases is proved by reduction from the emptiness problem for 2-head DFA’s.

(3) When \(\mathcal{L}_Q\) is FP and \(\mathcal{L}_C\) consists of fixed FP queries. Given a 2-head DFA \(A = (Q, \Sigma, \delta, q_0, q_{acc})\), we define \(R, R_m, D_m, V\) and \(Q\) as follows.

(a) We define the relational schema \(\mathcal{R} = (P, \bar{P}, F, R_\Delta, R^1_b, R^2_b, R^3_b, R^4_b)\), where \(P, \bar{P}, F\) and \(R_\Delta\) are the same as what we have seen in the proof for case (1), and for each \(i \in [1, 4]\), \(R^i_b\) is a unary relation schema. We define \(R_m = (R^m_1, R^m_2)\), where both \(R^m_1\) and \(R^m_2\) are unary relation schemas.

(b) The master data instance is \(D_m = (I^m_1 = \emptyset, I^m_2 = \{(1)\})\).

(c) The set \(V\) of CCS consists of \(V_1, V_2, V_3, V_4\) given in the proof for case (1), and in addition, three CCSs given below:

- \(\neg V^1_b : \exists x F(0, x) \land \exists x F(x, x) \subseteq R^m_2\), which states that instances \(I^2_b\) of \(R^1_b\) are bounded by \(I^m_2\), provided that the instance \(I_F\) of \(F\) contains an initial position;

- \(\neg V^2_b : \exists x F(x, x) \land \exists x F^2(x, x) \subseteq R^m_2\), which constrains instances \(I^2_b\) of \(R^2_b\) when the instance \(I_F\) of \(F\) contains a final position; and

- \(\neg V^3_b : \exists x \exists y_1 [\text{TC}_{x, y, z, x', y', z'} \cdot (q_0, 0, 0, q_{acc}, y_1, y_2) \land \exists x F(x, x) \subseteq \emptyset]\), which places \(I^m_2\) as an upper bound of instances \(I^2_b\) of \(R^2_b\) when the instance \(I_{\Delta}\) of \(R_{\Delta}\) contains a run from the initial state to the final state.

(d) The query \(Q\) is defined as \(Q(z) = Q_1(z) \cup Q_2(z)\), where

\[ Q_1(z) = (\bigwedge_{\delta \in \Delta} R_{\Delta}(t_\delta)) \times (R^1_b \cup R^2_b \cup R^3_b), \]

and \(t_\delta\) is the tuple encoding the transition \(\delta \in \Delta\) as described in the proof of Theorem 3.1; and \(Q_2(z) = (\exists s \in (\bigwedge_{\delta \in \Delta} R_{\Delta}(s)) \land (t = \bigvee_{\delta \in \Delta} t_\delta) \land s[q, x, y] = t[q, x, y] \land s[q', x', y'] \neq t[q', x', y']) \times R^4_b\). Here \(q, x\) and \(y\) are the key attributes of \(R_{\Delta}\), and \(q', x'\) and \(y'\) are its remaining attributes. In other words, \(Q_2\) returns \(R^4_b\) if there exist a tuple \(s\) in \(R_{\Delta}\) and a valid

ACM Transactions on Database Systems, Vol. V, No. N, Month 20YY.
transition tuple $t_δ$ encoding some $δ ∈ Δ$ such that $s$ and $t_δ$ violate the key.

Note that FP (DATALOG) can simulate disjunction by means of multiple rules for the same IDB predicate. Hence $Q(z)$ is indeed expressible in FP.

We show that $RCQ(Q, D_m, V)$ is nonempty iff $L(A) ≠ \emptyset$. First, suppose that $L(A) ≠ \emptyset$. Let $D = (I_P, I_P, I_F, I_A, I_c, I_q) = \{(1)\}, I_b = \{(1)\}, I_c = \{(1)\}, I_q = \emptyset$ such that (i) $I_P, I_P$, and $I_F$ encode an input string that is accepted by $A$; and (ii) $I_D$ contains all valid transitions of $Δ$. Then clearly $Q(D) = \{(1)\}$. We show that $D$ is complete for $Q$ relative to $(D_m, V)$. Indeed, no more tuples can be added to any of the instances $I^*_b = \{(1)\}, I^*_c = \{(1)\}$ and $I^*_q = \{(1)\}$ as they are constrained by $V$, and moreover, adding tuples to the other relations of $D$ does not change the query result. In particular, since $D$ contains all valid transitions, no tuples can be added to $I_A$ that satisfy the condition of $Q_2(D)$ and make $I^*_q$ part of the query result. In other words, $I^*_q$ does not affect the query result in this case. Therefore, $D$ is complete for $Q$ relative to $(D_m, V)$ and as a result, $RCQ(Q, D_m, V)$ is nonempty.

Conversely, assume that $L(A) = \emptyset$ but by contradiction, that $RCQ(Q, D_m, V)$ is nonempty. Let $D = (I_P, I_P, I_F, I_A, I_c, I_q, I_b, I_c, I_q)$ be complete for $Q$ relative to $(D_m, V)$. It suffices to consider two cases, when $Q(D) = \emptyset$ and when $Q(D) ≠ \emptyset$.

Assume first that $Q(D) = \emptyset$. It is easy to verify that $I_A$ and $\bigvee_{δ ∈ Δ} t_δ$ do not contain tuples that, when put together, violate the key. Indeed, otherwise we could add elements to $I^*_b$, showing that $D$ is not complete. Next we examine various cases of $Q_1$ that make $Q(D) = \emptyset$. (a) Suppose that $\bigwedge_{δ ∈ Δ} R_δ(t_δ)$ is not satisfied. Then we can simply add all tuples $t_δ$ to $I_A$, resulting in $J_A$. Let $D' = (I_P, I_P, I_F, I_A, I_c, I_q, I_b, I_c, I_q)$. Then $Q(D') ≠ \emptyset$ and hence $D$ is not complete. Observe that we can add these tuples to $I_A$ since we know that $I_A$ and $\bigvee_{δ ∈ Δ} t_δ$, when taken together, do not violate the key. (b) Assume that $Q(D) = \emptyset$ because $I^*_b = \emptyset$, but $\bigwedge_{δ ∈ Δ} R_δ(t_δ)$ is satisfied. Then we can define $I^*_b = \{(1)\}$, $J^*_b = \{(1)\}$ and $J^*_q = \{(1)\}$. Let $D' = (I_P, I_P, I_F, I_A, J_b, J_c, J_q)$. Then $Q(D') = \emptyset$ and hence $D$ is not complete. (c) Similarly, we can show that $D$ is not complete when both parts of $Q_1$ considered in (a) and (b) are not satisfied. These tell us that when $Q(D) = \emptyset$, $D$ cannot possibly be complete for $Q$ relative to $(D_m, V)$.

Now assume that $Q(D) ≠ \emptyset$. The argument above tells us that if $D$ is complete, then $\{(1)\} ⊆ I^*_b$, $\{(1)\} ⊆ I^*_c$, and $\{(1)\} ⊆ I^*_q$. Observe the following: (i) $I_P$ must contain a tuple of the form $(0, x)$ since otherwise we can change the query result by expanding $I^*_b$; and similarly, (ii) $I_P$ must contain a tuple of the form $(x, 0)$ because otherwise we can change the query result by expanding $I^*_c$. Both (i) and (ii) can be guaranteed without violating the CCs. In addition, for $D$ to be complete it is necessary to ensure that no more tuples can be added to $I^*_b$. This is only guaranteed when $I^*_b$ is bounded by $I^*_a$ via the CC $V_0^a$, and when the transitive closure of $I_A$ contains a tuple of the form $(q_0, 0, q_{acc}, y_1, y_2)$. However, since $I_A \models \bigwedge_{δ ∈ Δ} R_δ(t_δ)$ and $(D, D_m) \models V_4$ (stating the keys in the relation $R_δ$), putting these together we would have got an accepting run of $A$ on the input string encoded by $D$. This contradicts the assumption that $L(A) = \emptyset$. Hence $D$ is not complete for $Q$ relative to $(D_m, V)$. That is, $RCQ(Q, D_m, V)$ is empty.

(4) When $L_C$ is FP and $L_Q$ is CQ.
Given a 2-head DFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, q_{\text{acc}})$, we define $\mathcal{R}, \mathcal{R}_m, D_m, V$ and $Q$ as follows.

(a) We define the relational schema $\mathcal{R} = (P, \bar{P}, F_e)$, where $P$ and $\bar{P}$ are unary relations specified in the proof for case (1). Instead of using binary relation $F$ given there, we use a ternary relation schema $F_e$. We define $\mathcal{R}_m = (R_1^m, R_2^m)$, where $R_1^m$ and $R_2^m$ are both unary relations.

(b) The master data instance is $D_m = (I_1^m = \emptyset, I_2^m = \{(1)\})$, the same as in case (3).

(c) The query $Q$ is defined to be $Q(z) = \exists y F_e(0, y, z)$.

(d) The set $V$ of CCs consists of $V_3$ given in the proof for case (1), and the following:

$-V_3 : \exists x \exists y \exists u (F_e(x, y, u) \land F_e(x, y, v) \land u \neq v) \subseteq \emptyset$, ensuring that instances of $F_e$ are functions on the first two attributes;

$-V_4 : \exists x \exists y \exists u (F_e(x, x, u) \land F_e(x, y, v) \land x \neq y) \subseteq \emptyset$, asserting that there exists at most one tuple of the form $(k, k)$ in an instance of $F_e$;

$-V_5 : q(z) \subseteq R_2^m$, where $q(z) = Q'$, and $Q'$ is a minor variation of the FP query specified in the proof for Theorem 3.1 (4) that tests whether there exists an accepting run, using $P$, $\bar{P}$ and $\exists z F_e(x, y, z)$ as input. This CC ensures that if there is an accepting run, then tuples of the form $(0, y, z)$ in $F_e$ are $(0, y, 1)$. Observe that in this case, there exists only a single such tuple because $F_e$ is a function on the first two attributes.

We show that $\text{RCQ}(Q, D_m, V)$ is nonempty iff $L(\mathcal{A}) \neq \emptyset$. First, suppose that $L(\mathcal{A}) = \emptyset$ but $\text{RCQ}(Q, D_m, V)$ is nonempty. Then there exists $D$ in $\text{RCQ}(Q, D_m, V)$, where $D = (I_1, I_2, I_F)$. Let $Q(D) = \{c_1, \ldots, c_k\}$, and $c'$ be a value distinct from $c_i$ for $i \in [1, k]$. Let $J_{F_e} = I_{F_e} \cup \{(0, s_0, c')\}$, where $s_0$ is the unique value such that $(0, s_0, u)$ is in $I_{F_e}$. Since $L(\mathcal{A}) = \emptyset$, the CC $q(z) \subseteq R_2^m$ will be vacuously satisfied, i.e., $Q'$ is false and thus the $z$-values are not bounded. Let $D' = (I_1, I_2, I_{F_e})$. Then $(D', D_m) \models V$ but $Q(D') = Q(D) \cup \{c'\}$. This contradicts the assumption that $D$ is complete. Hence $\text{RCQ}(Q, D_m, V)$ must be empty.

Conversely, suppose that $L(\mathcal{A}) \neq \emptyset$. Let $D = (I_1, I_2, I_{F_e})$ be the instance of $\mathcal{R}$ that encodes an input string that is accepted by $\mathcal{A}$. Then $Q'$ in $V_4$ is satisfied. As a consequence, $q(z) \subseteq R_2^m$ holds, and the $z$-values of tuples $(0, s_0, z)$ in $F_e$ are bounded by $I_2^m$, i.e., these values have to be 1. Hence $Q(D) = \{(1)\}$. Moreover, it is easily verified that for each $D' = (I_1', I_2', I_{F_e})$ such that $D \subseteq D'$ and $(D', D_m) \models V$, we have that $Q(D') = \{(1)\}$. Indeed, the extra tuples in $D'$ do not contribute to the run of the DFA. More specifically, since $D \subseteq D'$, the query $Q'$ will evaluate to true on $D'$, due to the monotonicity of FP. Hence $V_4$ enforces the $z$-values in tuples $(0, s_0, z)$ of $F_e$ to be 1. In addition, since $F_e$ is a function on the first two attributes, no extra tuples of the form $(0, s', z')$ can be added to $I_{F_e}$. Hence $D$ is complete for $Q$ relative to $(D_m, V)$ and therefore, $\text{RCQ}(Q, D_m, V)$ is nonempty.

Observe that the proofs only fix $D_m, V$ when $L_Q$ is FO or FP, and thus verify the undecidability for fixed $D_m$ and $V$. □