Declarative Smart Contracts: Specifications and Optimizations

Anonymous Author(s)*

1 COMPLETE PROOF

In this section, we establish key properties of the selective view materialization algorithm, including the uniqueness of the minimal form of a view materialization plan, the termination guarantee, and completeness which together form the correctness of the algorithm.

1.0.1 Minimal form uniqueness. We first define the minimal form of a materialization plan, and claim the uniqueness of the minimal form, as formulated by Lemma 1. We denote a valid view materialization plan as a valid set for conciseness.

Definition 1. *A relation set S' is a minimal form of another relation set S if and only if:*

(1) $S' \subseteq S$,
(2) S' *is a minimal view materialization plan (Definition* **??***).*

Lemma 1. *For any valid materialization plan S, it must have one and only one minimal form S'.*

Proof sketch. We proceed by establishing contradictions. Assume that a valid materialization plan S has two distinct minimal forms, denoted as X and Y. To demonstrate the impossibility of such a scenario, we break down the proof by examining various overlapping cases of X and Y.

First, suppose $X \subset Y$, and let $D = Y \setminus X$. Since X is a valid relation set, we can obtain another valid set by removing relations in D from Y, which contradicts to the assumption that Y is in a minimal form. Similarly, the case where $Y \subset X$ also leads to contradiction.

The remaining case is where sets X and Y have at least one element that is unique to each set. Let A be the set of unique elements in X ($X \setminus Y$), and let B be the set of unique elements in Y ($Y \setminus X$), then we have $A \neq \emptyset$ and $B \neq \emptyset$.

Next, we want to show that for each relation a in A, the execution of queries to a depends on at least one relation in B. To see this, consider the fact that S is a valid materialization plan, and that Y is a minimal form of S. Consequently, relational queries to a should be executable given Y. Moreover, if there exists a relation a in A such that a is executable solely using relations in $X \cap Y$, then we can obtain another valid set by removing a from X, which contradicts the assumption that X is in a minimal form. Therefore, queries to each relation a in A must rely on at least one relation in B.

Similarly, queries to each relation b in B must also rely on at least one relation in A. This yields mutual dependencies between relations in set A and B. This contradicts the assumption that any contract has no circular dependencies among relations.

Therefore, we successfully establish the uniqueness of the minimal form of a given valid view materialization plan.

1.0.2 Termination. The enumeration algorithm (Algorithm **??**) traverses the replacement graph. Given the assumption that the replacement graph has no cycle, the traversal is guaranteed to terminate at nodes without any outgoing edges.

1.0.3 Completeness. We first introduce a few definitions before establishing the completeness theorem.

We denote *Replace* methods on a set of view materialization plans Σ as follows:

$$Replace(\Sigma) \triangleq \{Replace(S) \mid S \in \Sigma\}$$

Similarly,

$$GetMinimal(\Sigma) \triangleq \{GetMinimal(S) \mid S \in \Sigma\}$$

Definition 2. *View materialization plan space. Given a declarative smart contract, the set of all possible view materialization plans obtained by applying the Replace method for k times is defined as:*

$$\Sigma_k \triangleq \bigcup_{i=0}^{k} Replace^i(BasePlan)$$

Here $Replace^i(BasePlan)$ denotes the set of view materialization plans obtained by applying the *Replace* method (Algorithm **??**) to the base materialization plan i times. Since circular dependencies among relations are prohibited, the corresponding replacement graph G, has no circle. Let k denote the length of the longest path in G, the *Replace* method can be applied for at most k times.

Definition 3. *MinReplace. We combine the Replace and GetMinimal operations performed at every iteration of Algorithm* **??** *as the following operator:*

$$MinReplace(\Sigma) \triangleq GetMinimal(Replace(\Sigma))$$

where the input Σ represents a set of view materialization plans. With this definition, the output of Algorithm **??** is effectively:

$$\bigcup_{i=0}^{K} MinReplace^i(B')$$

where B' is the minimal form of the base view materialization plan.

Theorem 1. *Completeness. Given a set of relational queries, let B be the base materialization plan, and B' be the minimal form of B (GetMinimal(B)), Algorithm* **??** *outputs all minimal forms of view materialization plans obtainable from Σ:*

$$\bigcup_{i=0}^{K} MinReplace^i(B') = GetMinimal(\Sigma_K)$$

This theorem asserts that the application of the *GetMinimal* method during each iteration of the algorithm does not compromise the final correctness of the algorithm. Instead, it enhances efficiency by early pruning of non-minimal materialization plans.

To prove Theorem 1, we first establish the following lemma.

Lemma 2. *For all valid materialization plan S, let S' be the minimal form of S:*

$$MinReplace(\{S\}) = MinReplace(\{S'\})$$

Proof sketch. We first prove $MinReplace(\{S'\}) \subseteq MinReplace(\{S\})$. $\forall$ relation $r \in S$, (1) if $r \in S'$, then r can be executed by any replacement choices of S' including the reduced one v; (2) if $r \notin S'$, then r must be calculated using some relations $p \in P$ where $P \subseteq S'$. In this case, $\forall p \in P$, we have $p \in S'$, and therefore p can be executed by any replacement choices of S' including the reduced one v, and thus r can also be executed by v. Therefore, v is also a reduced alternative choice of S.

We next prove $MinReplace(\{S\}) \subseteq MinReplace(\{S'\})$. $\forall$ relation $r \in S'$, since $S' \subseteq S$, we have $r \in S$, and therefore r can be executed by any replacement choices of S including the reduced one u. So u is also a reduced alternative choice of S'.

Following Lemma 2, we further deduce the following lemma:

Lemma 3. *Let Σ be a set of valid materialization plan, We have:*

$$MinReplace(\Sigma) = MinReplace(GetMinimal(\Sigma))$$

Given Lemma 3, we proceed to prove completeness (Theorem 1) of the materialization enumeration algorithm. The proof is conducted via mathematical induction on the number of the application for *MinReplace* method i.

For the base case, where $i = 0$, and $B' = GetMinimal(B)$, which is true following our assumption.

For induction case where $i = n$, the induction hypothesis is as follow:

$$\bigcup_{i=0}^{n} MinReplace^i(B') = \{GetMinimal(S) | S \in \Sigma_n\}$$

We then analyze the case where $i = n + 1$, the output of the enumeration algorithm after $n + 1$ applications of the *minReplace* method can be rewritten as follows:

$$\bigcup_{i=0}^{n+1} MinReplace^i(B') = \bigcup_{i=0}^{n} MinReplace^i(B') \cup minReplace^{n+1}(B')$$

Substituting the right hand side of the above eqaution with the induction hypothesis, we have:

$$\bigcup_{i=0}^{n+1} MinReplace^i(B') = \{GetMinimal(S) | S \in \Sigma_n\} \cup minReplace^{n+1}(B') \tag{1}$$

Next, we expand $minReplace^{n+1}(B')$:

$$minReplace^{n+1}(B') = minReplace^n(minReplace(B'))$$

By lemma 3, we can further rewrite the right-hand-side:

$$\begin{aligned} minReplace^{n+1}(B') &= minReplace^n(minReplace(B)) \\ &= minReplace^n(GetMinimal(Replace(B))) \\ &= minReplace^n(Replace(B)) \\ &\dots \\ &= minReplace(Replace^n(B)) \\ &= GetMinimal(Replace^{n+1}(B)) \\ &= \{GetMinimal(S) \mid S \in Replace^{n+1}(B)\} \end{aligned}$$

Substituting the right-hand-side of the above equation back to Equation 1, we have:

$$\bigcup_{i=0}^{n+1} MinReplace^i(B') = \{GetMinimal(S) \mid S \in \Sigma_{n+1}\}$$

which establishes the validity for $i = n + 1$.

Here, we have successfully completed the induction step and prove the completeness property as defined in Theorem 1.