Formalizing and Checking Multilevel Consistency*

Ahmed Bouajjani[1], Constantin Enea[1], Madhavan Mukund[2,3], Gautham Shenoy R[2], and S P Suresh[2,3]

[1] Université Paris Diderot, France {abou,cenea}@irif.fr
[2] Chennai Mathematical Institute, India {madhavan,gautshen,spsuresh}@cmi.ac.in
[3] CNRS UMI 2000 ReLaX

Abstract. Developers of distributed data-stores must trade consistency for performance and availability. Such systems may in fact implement weak consistency models, e.g., causal consistency or eventual consistency, corresponding to different costs and guarantees to the clients. We consider the case of distributed systems that offer not just one level of consistency but multiple levels of consistency to the clients. This corresponds to many practical situations. For instance, popular data-stores such as Amazon DynamoDB and Apache's Cassandra allow applications to tag each query within the same session with a separate consistency level. In this paper, we provide a formal framework for the specification of multilevel consistency, and we address the problem of checking the conformance of a computation to such a specification. We provide a principled algorithmic approach to this problem and apply it to several instances of models with multilevel consistency.

1 Introduction

To achieve availability and scalability, modern data-stores (key-value stores) rely on optimistic replication, allowing multiple clients to issue operations on shared data on a number of replicas, which communicate changes to each other using message passing. One benefit of such architectures is that the replicas remain locally available to clients even when network connections fail. Unfortunately, the famous CAP theorem [14] shows that such high Availability and tolerance to network Partitions are incompatible with strong Consistency, i.e., the illusion of a single centralized replica handling all operations. For this reason, modern replicated data-stores often provide weaker forms of consistency such as eventual consistency [22] or causal consistency [18], which have been formalized only recently [6, 7, 9, 21].

Programming applications on top of weakly-consistent data-stores is difficult. Some form of synchronization is often unavoidable to preserve correctness. Therefore, popular data-stores such as Amazon DynamoDB and Apache's Cassandra provide different levels of consistencies, ranging from weaker forms to

* Partially supported by CEFIPRA DST-Inria-CNRS Project 2014-1, AVeCSo.

strong consistency. Applications can tag queries to the data-store with a suitable level of consistency depending on their needs.

Implementations of large-scale data-stores are difficult to build and test. For instance, they must account for partial failures, where some components or the network can fail and produce incomplete results. Ensuring fault-tolerance relies on intricate protocols which are difficult to design and reason about. The black-box testing framework Jepsen[4] found a remarkably large number of subtle problems in many production distributed data-stores.

Testing a data-store raises two issues: (1) deriving a suitable set of testing scenarios, e.g., faults to inject into the system and the set of operations to be executed, and (2) efficient algorithms for checking whether a given execution satisfies the considered consistency models. The Jepsen framework shows that the first issue can be solved using randomization, e.g., introducing faults at random and choosing the operations randomly. The effectiveness of this solution has been proved formally in recent work [20]. The second issue is dependent on a suitable formalization of the consistency models.

In this work, we consider the problem of specifying data-stores which provide multiple levels of consistency and derive algorithms to check whether a given execution adheres to such a multilevel consistency specification.

We build on the specification framework in [9] which formalizes consistency models using two auxiliary relations: (i) a *visibility* relation, which specifies the set of operations observed by each operation, and (ii) an *arbitration order*, which specifies the order in which concurrent operations should be viewed by all replicas. An execution is said to satisfy a consistency criterion if there exists a visibility relation and an arbitration order that obey an associated set of constraints. For the case of a data-store providing multiple levels of consistency, we consider multiple visibility relations and arbitration orders, one for each level of consistency. Then, we consider a set of formulas which specifies each consistency level in isolation, and also, how visibility relations and arbitration orders of different consistency levels are related.

Based on this formalization, we investigate the problem of checking whether a given execution satisfies a certain multilevel consistency specification. In general, this problem is known to be NP-COMPLETE [6]. However, we show that for executions where each value is written at most once to a key, this problem is polynomial time for many practically-interesting multilevel consistency specifications. Since practical data-store implementations are data-independent [23], i.e., their behaviour doesn't depend on the concrete values read or written in the transactions, it suffices to consider executions where each value is written at most once. This complexity result uses the idea of *bad patterns* introduced in [6] for the case of causal consistency. Intuitively, a bad pattern is a set of operations occurring in a particular order corresponding to a consistency violation. In this paper, we provide a *systematic methodology* for deriving bad patterns characterizing a wide range of consistency models and combinations thereof.

[4] Available at `http://jepsen.io`

Our contributions form an effective algorithmic framework for the verification of modern data-stores providing multiple levels of consistency. To the best of our knowledge, we are the first to investigate the asymptotic complexity for such a wide class of consistency models and their combinations, despite their prevalence in practice.

The paper is organized as follows. We begin with some real-life examples of multilevel consistency. In Section 3, we present a formal model for specifying and reasoning about multilevel consistency. Section 4 describes algorithms for verifying multilevel consistency. We conclude with a discussion of related work. Some details and proofs are presented in an Appendix.

2 Multilevel consistency in the wild

In this section we present some real-world instances of multilevel consistency. We restrict our attention to distributed read-write key-value data-stores (henceforth referred to as read-write stores), consisting of unique memory locations addressed by *keys* or *variables*. We use *keys* and *variables* interchangeably in this work. The contents of these memory locations come from a domain, called *values*.

The read-write data-store provides two APIs to access and modify the contents of a particular memory location. The API to read the content of a particular memory location is typically named *Read* or *Get*, and the API to store a value into a particular memory location is typically named *Write* or *Put*. In this paper, we refer to these two methods as Read and Write respectively. The Read method does not update the state of the data-store and only reveals part of the state to the application session which invokes the method. The Write method on the other hand modifies the state of the data-store.

Typically, an application reads a location of the data-store, performs some local computation and writes a value back to the data-store. A sequence of related read and write operations performed by an application is called a *session*.

Applications expect some sort of consistency guarantee from the data-store in terms of how *fresh* or *stale* the data value is that they read from the data-store. They also seek some guarantees pertaining to monotonicity of the results that are presented to them. These guarantees provided by the data-store to the applications are called *consistency criterion*. Some of the popular consistency criteria include:

– **Read-Your-Writes**: The effects of prior operations in the session will be visible to later operations in the same session.
– **Monotonic Reads**: Once the effect of an operation becomes visible within a session, it remains visible to all subsequent operations in that session.
– **Monotonic Writes**: If the effect of a remote operation is visible in a session, then the effects of all prior operations in the session of the remote operation are also visible.
– **Causal consistency**: Effects of prior operations in a session are always visible to later operations. Further, if the effect of an operation is visible

to another operation, then every operation that has seen the effects of the latter would have seen the effects of the former.

– **Sequential Consistency**: Effects of the operations can be explained from a single sequential execution obtained by interleaving the reads and writes performed at individual sessions.

Most of the existing literature on testing the behaviour of read-write stores focuses on testing the correctness with respect to specific consistency criteria [6, 7, 13]. However, there are cases where data-stores such as DynamoDB and Cassandra offer to applications the choice of specifying the consistency level per read-operation [10]. There are distributed data-store libraries that allow consistency rationing [17] and also allow incremental consistency guarantees for the read operations [16]. In each of these cases we need to reason about the correctness of the behaviour of the data-store with respect to more than one consistency criterion.

We now look at some examples of multilevel consistency in the real world. In this work, we assume that the Read and the Write APIs are as follows.

Definition 1 (Read and Write APIs) *Let x be a key/variable, val denote a value read-from/written-to the data-store and $level$ denote the consistency level.*

– Write(x, val) *: Updates the content of the memory location addressed by the key/variable x with the value val.*

– Read$(x, val, level)$ *: The content of the memory location whose key is x is val with respect to the consistency level $level$.*

Read-Write Stores with strong and weak reads

Consider the case of the data-store Cassandra, which allows the application a more fine grained choice of consistency levels, such as `ANY`, `ONE`, `QUORUM`, `ALL`. It achieves this by ensuring that when the Read is executed with `ANY`, the return value is provided by consulting any available replica of the data store. Similarly, if the Read operation is submitted with `ONE`, the return value is provided by consulting a replica that is known to contain at least one value for that key. On the other hand, if the Read is executed with `QUORUM`, the data-store returns the value after consulting a majority of the replicas. Finally, if Read is executed with `ALL`, then all the replicas are consulted before returning the response. Clearly, `ANY` is the weakest consistency criterion while `ALL` is the strongest consistency criterion. In general, a data-store offers responses pertaining to different consistency criteria by consulting the required subset of replicas to answer the query.

Typically a read operation under the stronger consistency criterion will take more time, since it might have to wait for all pending operations to become visible, or run a consensus protocol before returning the result. In certain cases, applications may be satisfied with Read operations that return values that are correct with respect to some weaker consistency criterion. Consider a web-application that displays the available seats in a movie theater. The application can choose to read the available seats based on a weaker consistency criterion, since:

– The number of users attempting to book seats is usually more than the seats available. Waiting for a consensus or a quorum can slow down the reads for everyone. So a quicker response is desirable.
– There is a lag between the time the user gets to see available seats and the time when the user decides to book particular seats. Since concurrent bookings are ongoing, the data displayed can become stale by the time the user books the seat.
– Users can change their minds before finally settling on a set of seats, and paying for them.

Thus, the web-application can opt for a read satisfying a weaker consistency criterion while allowing the user to pick a seat, and then perform a read satisfying a stronger consistency criterion only when the user pays for it.

Consider the example in Figure 1 where all write requests are processed at the same replica. For each session, there is a (potentially different) designated replica from which the responses to the weak reads are returned.

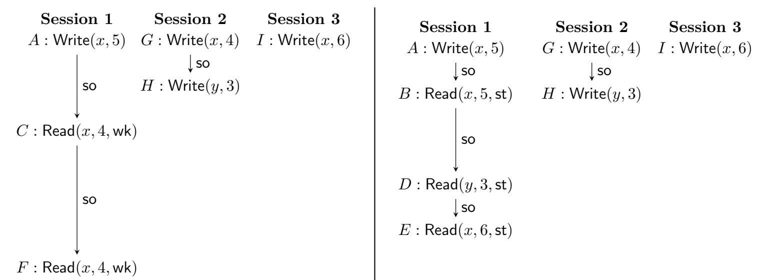
$$
\begin{tikzcd}[row sep=small]
\textbf{Session 1} & \textbf{Session 2} & \textbf{Session 3} \\
A : \mathsf{Write}(x, 5) \arrow[d, "\mathsf{so}"] & G : \mathsf{Write}(x, 4) \arrow[d, "\mathsf{so}"] & I : \mathsf{Write}(x, 6) \\
B : \mathsf{Read}(x, 5, \mathsf{st}) \arrow[d, "\mathsf{so}"] & H : \mathsf{Write}(y, 3) & \\
C : \mathsf{Read}(x, 4, \mathsf{wk}) \arrow[d, "\mathsf{so}"] & & \\
D : \mathsf{Read}(y, 3, \mathsf{st}) \arrow[d, "\mathsf{so}"] & & \\
E : \mathsf{Read}(x, 6, \mathsf{st}) \arrow[d, "\mathsf{so}"] & & \\
F : \mathsf{Read}(x, 4, \mathsf{wk}) & &
\end{tikzcd}
$$

Fig. 1: An example of a read-write store behaviour with strong and weak reads. The so relation relates read and write operations from the same session in the order in which they happened in that session.

In this scenario, the strong reads (corresponding to the consistency level ALL) satisfy sequential consistency while the weak reads obey monotonic reads consistency. Hence, the fragment consisting of all the writes and the weak reads should be correct with respect to monotonic reads. Similarly, the fragment consisting of all the writes and the strong reads should be correct with respect to sequential consistency.

The weak fragment corresponding to the example in Figure 1 can be seen in Figure 2(a). This fragment is correct with respect to monotonic reads; once the write G is visible at session 1 to the read C, it remains visible throughout the session. The write I is not visible to any of the other sessions yet.

The strong fragment is represented in Figure 2(b). This is correct with respect to sequential consistency, where the order of the operations obtained by consensus is $A \to B \to G \to H \to I \to D \to E$.



(a) Weak Fragment from Figure 1 | (b) Strong Fragment from Figure 1

Fig. 2: Strong and Weak fragments of the hybrid behaviour

However, since the strong reads correspond to the level `ALL` where all the replicas have seen the prior writes and have agreed on the order of the concurrent writes, it behooves a weak read following a strong read to take into consideration the effects seen by the earlier strong read. Thus the data-store imposes an additional constraint. Once a write is visible to a strong read in a session, it is visible to all the subsequent weak reads in that session. This ensures that the weaker reads do incorporate the prior results seen by the session. Similarly, a write visible to a weak read is made from a replica which participates in the subsequent strong reads corresponding to the level `ALL`. Thus the effects visible to the prior weak reads in a session are also visible to the subsequent strong reads.

With these additional constraints, we can no longer explain the read operation F, since the effects of writes G and I are both visible at read F. The strong consistency criterion has already guaranteed that write I has happened after write G, thereby effectively overwriting the value 4 with the value 6. Hence this behaviour is incorrect in the multilevel setting.

Now consider the behaviour of Cassandra where writes are performed at one of the replicas (corresponds to the level `ONE`), weak reads are performed at one of the replicas (corresponds to the level `ONE`) and strong reads are performed at a quorum of replicas (corresponds to the level `QUORUM`). In this situation, it is not necessary that the effects of writes visible to prior weaker reads are visible at subsequent stronger reads, since the replica from which the weaker read is performed may be missing from the quorum of replicas from which the stronger read is made. Similarly, the effects of writes visible to prior stronger reads of a session need not be visible to the subsequent weaker reads in the session, as the writes from the quorum may not have reached the replica from which the weaker read is performed. Thus, the stronger and weaker reads can be independent of each other.

Finally consider the case of Amazon DynamoDB Accelerator (DAX) [1], which contains a write-through cache sitting between the application and the DynamoDB backend. Every write made by the application is first submitted to the DynamoDB backend and also updated at the cache. By default, the reads are eventually consistent, i.e., the reads are performed from the cache. If the item does not exist in the cache, then it is fetched from the backend data-store and the cache is updated with the item before the value is returned to the application. However, the application can also request strongly consistent reads by invoking `ConsistentRead`. In this case, the value is read from the backend and returned to the application, without caching the results. Any subsequent eventually consistent reads made by the application may not reflect the value returned by the prior strongly consistent read. In the case of DAX, it can be observed that the effects of the writes visible to the weak eventually consistent reads are also visible to the subsequent strongly consistent reads as those writes are also present in the DynamoDB backend. However, it is not necessary that the effects of writes visible to the strongly consistent reads are visible to the subsequent weak eventually consistent reads.

From these examples of multilevel consistency, we can see that the presence of another consistency criterion can impose additional constraints on the choice of the visibility and arbitration relations chosen to explain the correctness of the history. In the next section, we provide a formal framework for modelling behaviours of read-write data-stores with multiple consistency levels.

3 Formalizing Multilevel Consistency

We extend the formal framework provided in [8] for modelling the behaviours of read-write stores. Each operation submitted to the data-store by the application is either a Read or a Write operation whose signature is given in Definition 1.

We denote the set of all variables in the read-write store by *Vars* and assume that each value written to the read-write store is a natural number $val \in \mathbb{N}$. We assume that all variables are initially undefined, with value $\perp$.

For simplicity, we assume only two consistency levels, weak and strong, denoted by wk and st, respectively, where the consistency criterion corresponding to wk-level is strictly weaker than then the consistency criterion corresponding to the st-level. Comparison between consistency criteria is formally defined in Definition 7.

The behaviour of the read-write data-store as observed by an application is the sequence of reads and writes that it performs on the stores. The sequence of related read and write operations is termed a *session*. Thus the behaviour of the read-write store seen by each session is a total order of read/write operations performed in that session.

The behaviour of the read-write store is the collection of behaviours seen by all the sessions. In Figure 1 we saw the behaviour of the data-store as observed by the three sessions accessing the data-store. We call such a behaviour a *hybrid history*, formally defined as follows:

Definition 2 (Hybrid History) *A hybrid history of a read-write store is a pair* $H = (\mathcal{O}, \mathsf{so})$ *where* $\mathcal{O}$ *is the set of read-write operations and* so *is a collection of total orders called session orders.*

For a history H*, we define the following subsets of* $\mathcal{O}$*.*

- $\mathcal{O}_{\mathsf{Read}}$ *is the set of read operations occurring in* H*.*
- $\mathcal{O}_{\mathsf{Write}}$ *is the set of write operations occurring in* H*.*
- $\mathcal{O}_{\mathsf{wk}} = \mathcal{O}_{\mathsf{Write}} \cup \{\mathsf{Read}(x, val, level) \in \mathcal{O}_{\mathsf{Read}} \mid level = \mathsf{wk}\}$ *(the set of weak operations occurring in* H*).*
- $\mathcal{O}_{\mathsf{st}} = \mathcal{O}_{\mathsf{Write}} \cup \{\mathsf{Read}(x, val, level) \in \mathcal{O}_{\mathsf{Read}} \mid level = \mathsf{st}\}$ *(the set of strong operations occurring in* H*).*

The weak fragment of the history H *is denoted* H_{wk} *and defined to be* $(\mathcal{O}_{\mathsf{wk}}, \mathsf{so} \cap (\mathcal{O}_{\mathsf{wk}} \times \mathcal{O}_{\mathsf{wk}}))$*. Similarly the strong fragment of the history* H *is denoted* H_{st} *and is defined to be* $(\mathcal{O}_{\mathsf{st}}, \mathsf{so} \cap (\mathcal{O}_{\mathsf{st}} \times \mathcal{O}_{\mathsf{st}}))$*. Note that we take the write operations to be part of both the strong and weak fragments.*

- For $X \subseteq \mathcal{O} \times \mathcal{O}$ and $\ell \in \{\mathsf{Read}, \mathsf{Write}, \mathsf{wk}, \mathsf{st}\}$, $X \upharpoonright_{\ell} = X \cap (\mathcal{O}_{\ell} \times \mathcal{O}_{\ell})$.
- For $X, Y \subseteq \mathcal{O} \times \mathcal{O}$, $X; Y$ denotes composition of X and Y, i.e., $\{(x, y) \mid \exists z : (x, z) \in X \text{ and } (z, y) \in Y\}$.
- For $X \subseteq \mathcal{O} \times \mathcal{O}$, $\mathsf{total}(X)$ is used to mean that X is a total order.

When a replica of the read-write store receives an operation from an application, it decides how the effects of the older operations known to the replica (either received from applications, or from other replicas of the data-store) should be made visible to the new operation. A visibility relation over a history specifies the set of operations visible to an operation.

Definition 3 (Visibility Relation) *A visibility relation* vis *over a history* $H = (\mathcal{O}, \mathsf{so})$ *is an acyclic relation over* $\mathcal{O}$*. For* $o, o' \in \mathcal{O}$*, we write* $o \xrightarrow{\mathsf{vis}} o'$ *to indicate that the effects of the operation* o *are visible to the operation* o'*.*

If a pair of operations o, o' *are not related by* vis*, we term them concurrent operations, denoted by* $o \parallel_{\mathsf{vis}} o'$*.*

We define the view *of an operation* o *with respect to a visibility relation* vis*, denoted* $\mathsf{View}_{\mathsf{vis}}(o)$ *to be the set of all the* Write *operations visible to it.*

For the history in Figure 1, we can define a visibility relation to be

$$\{A \xrightarrow{\mathsf{vis}} B, G \xrightarrow{\mathsf{vis}} C, G \xrightarrow{\mathsf{vis}} D, H \xrightarrow{\mathsf{vis}} D, G \xrightarrow{\mathsf{vis}} E, H \xrightarrow{\mathsf{vis}} E, I \xrightarrow{\mathsf{vis}} E, G \xrightarrow{\mathsf{vis}} F\}$$

When the replicas communicate with each other, they need to reconcile the effects of concurrent write operations in order to converge to the same state eventually. In case of convergent data-stores this is done using a rule such as *Last Writer Wins* which totally orders all write operations. This is abstracted by an arbitration relation, which is a total order over all write operations in the history. We will denote the arbitration relation by arb. We assume that the arbitration relation is consistent with the visibility relation, in the sense that for a pair of writes o and o', if o is visible to o' then o is before o' in arb.

Definition 4 (Arbitration Relation) *An arbitration relation* arb *over a hybrid history* $H = (\mathcal{O}, \mathsf{so})$ *is a total order over* $\mathcal{O}_{\mathsf{Write}}$*. For* $o_i, o_j \in \mathcal{O}$*, we say* $o_i \xrightarrow{\mathsf{arb}} o_j$ *to indicate that operation* o_i *has been ordered before the operation operation* o_j*.*

For the history in Figure 1 the arbitration relation can be the total order

$$A \xrightarrow{\mathsf{arb}} G \xrightarrow{\mathsf{arb}} H \xrightarrow{\mathsf{arb}} I$$

We define the correctness of a hybrid history in terms of the *functional specification* of read-write stores.

Let H be a hybrid history. Let vis and arb be visibility and arbitration relations over H.

We say that a write operation o' is a *related-write* of a read operation o iff o' is in the view of o and both o and o' operate on the same variable. The set of all related writes of o, denoted as $RelWrites_{\mathsf{vis}}(o)$ is defined to be $\{o' \in \mathsf{View}_{\mathsf{vis}}(o) \mid o$ and o' operate on the same variable$\}$.

$MaxRelWrites_{\mathsf{vis}}(o)$, the set of maximal elements among these related writes with respect to vis, is defined to be

$$\{o' \in RelWrites_{\mathsf{vis}}(o) \mid \forall o'' \in RelWrites_{\mathsf{vis}}(o) : o'' \xrightarrow{\mathsf{vis}} o' \vee o'' \parallel_{\mathsf{vis}} o'\}$$

The effective write of a read-operation o, denoted by $EffWrite^{\mathsf{arb}}_{\mathsf{vis}}(o)$ is defined to be the maximum write operation from the set of maximal related writes of o as per the arbitration relation.

$$EffWrite^{\mathsf{arb}}_{\mathsf{vis}}(o) = \begin{cases} max(\mathsf{arb}\restriction_{MaxRelWrites_{\mathsf{vis}}(o)}) & \text{if } MaxRelWrites_{\mathsf{vis}}(o) \neq \emptyset \\ \perp & \text{otherwise} \end{cases}$$

Definition 5 (Functional Correctness for Read-Write Stores) *Let* $H = (\mathcal{O}, \mathsf{so})$ *be a hybrid history of a read-write data store with visibility relation* vis *and arbitration relation* arb*. We say that* $(H, \mathsf{vis}, \mathsf{arb})$ *is* functionally correct *iff for every read operation* $o = \mathsf{Read}(x, val, level)$*, the following conditions hold.*

- $EffWrite^{\mathsf{arb}}_{\mathsf{vis}}(o) = \perp$ *iff* $val = \perp$ *(i.e., there was no write operation on* x *when* o *happened).*
- *If* $o' = EffWrite^{\mathsf{arb}}_{\mathsf{vis}}(o)$ *then* o' *wrote the value* val*.*

Next, we formally define *consistency criteria* in terms of a set of formulas. Our definition is adapted from the definitions of constraints in [12].

Definition 6 (Consistency Criteria) *A* relation term τ *is a composition of the form* $r_1; \cdots ; r_k$ $(k \geq 1)$*, where each* $r_i \in \{so, vis\}$*. A* consistency criterion *is a subset of*

$$\{\tau \subseteq vis \mid \tau \text{ is a relation term}\} \cup \{\mathsf{total}(vis)\}.$$

Thus a consistency criterion is a possibly empty collection of visibility constraints and an optional totality constraint. For simplicity of notation, we usually write a constraint as a conjunction.

Note that so and vis are variables which are usually interpreted as restrictions of the so *and* vis *relations in a history. As we will see below, we always require an additional constraint that* $\mathsf{vis}\restriction_{\mathsf{Write}} \subseteq \mathsf{arb}$ *(and hence it is not explicitly included in the consistency criteria).*

For a consistency criterion α, $RelTerms(\alpha)$ is the set of all relation terms occurring in α, and $VisBasic(\alpha)$ is the collection of all visibility constraints in α excluding the totality constraint $\mathsf{total}(vis)$

Definition 7 (Consistency Criterion in a history) *Let $H = (\mathcal{O}, \mathsf{so})$ be a hybrid history, let vis and arb be a visibility and arbitration relation over H, and let α be a consistency criterion. We say that $H, \mathsf{vis} \models \alpha$ iff:*

1. *for every $\tau \subseteq vis$ in α, $\tau[so := \mathsf{so}, vis := \mathsf{vis}] \subseteq \mathsf{vis}$, and*
2. *if $\mathsf{total}(vis) \in \alpha$, then $\mathsf{total}(\mathsf{vis})$ holds.*

Further we say that $H, \mathsf{vis}, \mathsf{arb} \models \alpha$ iff $H, \mathsf{vis} \models \alpha$ and $\mathsf{vis}\restriction_{\mathsf{Write}} \subseteq \mathsf{arb}$.

Some well known consistency criteria are given in Table 1.

Name	Description
Basic Eventual Consistency (BEC)	$\top$
Read Your Writes (RYW)	$so \subseteq vis$
Monotonic Reads (MR)	$vis; so \subseteq vis$
Monotonic Writes (MW)	$so; vis \subseteq vis$
Strong Eventual Consistency (SEC)	$so \subseteq vis \ \wedge \ vis; so \subseteq vis$
FIFO Consistency (FIFO)	$so \subseteq vis \ \wedge \ vis; so \subseteq vis \ \wedge \ so; vis \subseteq vis$
Causal Consistency (CC)	$so \subseteq vis \ \wedge \ vis; vis \subseteq vis$
Sequential Consistency (SEQ)	$so \subseteq vis \ \wedge \ vis; vis \subseteq vis \ \wedge \mathsf{total}(vis)$

Table 1: Well known consistency criteria

We say that a consistency criterion α is at least as *strong* as another consistency criterion α' if for every history H, visibility relation vis, and arbitration relation arb over H, if $H, \mathsf{vis}, \mathsf{arb} \models \alpha$ then $H, \mathsf{vis}, \mathsf{arb} \models \alpha'$.

Suppose $H = (\mathcal{O}, \mathsf{so})$ is a hybrid history. Let α_w and α_s respectively be the wk and st consistency criteria. We then want to choose wk and st visibility relations $\mathsf{vis}_{\mathsf{wk}}$, $\mathsf{vis}_{\mathsf{st}}$, respectively, and an arbitration relations arb such that $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{arb} \models \alpha_w$ and $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}}, \mathsf{arb} \models \alpha_s$.

As we had noted in the previous section, in a multilevel setting, it is not sufficient to separately satisfy the constraints corresponding to the wk and st consistency criteria. We now proceed to modelling multilevel consistency constraints.

Modelling Multilevel Consistency

Taking inspiration from DAX [1] and the cache-hierarchy in modern processors, we can model multilevel consistency as a series of data-stores arranged in increasing order of the consistency they guarantee, such that the data-store offering the weakest level of consistency is closest to the application, and the data-store offering the strongest level of consistency is farthest away from the application. We shall further assume that these data-stores use the same arbitration strategy to order concurrent write operations and every weaker data-store has the capability to update its state to match that of a stronger data-store.

For the purpose of this paper, since we are restricting ourselves to only two levels, namely wk and st, this will reduce to having just two data-stores, where the data-store corresponding to the weaker consistency criterion sits as a cache between the application and the data-store corresponding to the stronger consistency criterion.

All the wk-reads are performed from the wk data-store.

There are two possible ways in which the writes can be performed.

1. Write-Through: The write is first performed at the st-data-store and eventually will be propagated to the wk-data-store.
2. Write-Back: The write is first performed at the wk-data-store and eventually will be propagated to the st-data-store.

There are two possible ways in which st-reads can be performed.

(a) Read-Through: The result of the st-read performed at the st-data-store is directly sent to the application bypassing the wk-data-store.
(b) Read-Back: The result of the st-read is updated at the wk-data-store before it is propagated to the application.

Thus, the system picks one of two ways to perform the write, and one of the two ways to perform the st-read.

Note that a system which picks the Write-Through strategy for performing the write will ensure that any write visible at the wk data-store will also be visible to the st data-store, as all the writes are first performed at the st data-store before they are propagated to the wk one. Hence, the effects of write operations visible to a wk-read operation are also visible to the subsequent st-operations in the session.

Similarly a system which picks the Read-Back strategy for performing the st-reads will ensure that any write that is visible to a *strong*-read will also be visible at a subsequent wk-read in the session as before returning the result of the st-read to the application, the result is merged into the wk data-store.

However, the Write-Back and Read-Through strategies do not provide any guarantees between the effects of writes visible to wk (resp. st) reads in relation to the subsequent st (resp. wk) reads in that session.

We now define the guarantees provided by each of these four strategies in the form of a constraint.

Definition 8 (Multilevel Constraints) *We define the following formulas:*

- $\psi^{write}_{thru} := (vis^{wk}; so)\restriction_{\mathsf{st}} \subseteq vis^{st}$
- $\psi^{write}_{back} := \top$
- $\psi^{read}_{thru} := \top$

– $\psi^{read}_{back} := (vis^{st}; so) \restriction_{\mathsf{wk}} \subseteq vis^{wk}$

A multilevel constraint φ *is a conjunction* $\psi^{read} \wedge \psi^{write}$*, where* $\psi^{read} \in \{\psi^{read}_{thru}, \psi^{read}_{back}\}$ *and* $\psi^{write} \in \{\psi^{write}_{thru}, \psi^{write}_{back}\}$.

Suppose $H = (\mathcal{O}, \mathsf{so})$ is a history, and $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$ are two visibility relations respectively over $\mathcal{O}_{\mathsf{wk}}$ and $\mathcal{O}_{\mathsf{st}}$. Let φ be a multilevel constraint. We say that $H, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}} \models \varphi$ iff $\varphi[so := \mathsf{so}, vis^{wk} := \mathsf{vis}_{\mathsf{wk}}, vis^{st} := \mathsf{vis}_{\mathsf{st}}]$ is true.

The formula ψ^{write}_{thru} imposes the constraint that the strong operations see the effects seen by the prior weak operations in the session. Similarly, the formula ψ^{read}_{back} imposes the constraint that the weak operations see the effects seen by the prior strong operations in the session. These two guarantee that the effect seen by reads of one consistency level remain monotonically visible to the subsequent reads of another consistency level.

Consider Cassandra's multilevel consistency with writes performed at level `ONE`, weak-reads at level `ONE` and strong-reads at level `ALL` which ensure that weaker reads see the effects visible to prior stronger reads and vice-versa. This can be modelled using $\psi^{write}_{thru} \wedge \psi^{read}_{back}$.

On the other hand, Cassandra's multilevel consistency with writes performed at level `ONE`, weak-reads at level `ONE` and strong-reads at level `QUORUM` neither ensures that weaker reads see the effects visible to prior stronger reads nor the converse. This can be modelled using $\psi^{write}_{back} \wedge \psi^{read}_{thru}$.

The DynamoDB's DAX case can be modelled using $\psi^{write}_{thru} \wedge \psi^{read}_{thru}$ which only allows for the effects of prior weak reads to be visible to subsequent stronger reads, but not the converse.

We now formally define when a hybrid history is correct.

Definition 9 (Multilevel Correctness of a Hybrid History) *A hybrid history* $H = (\mathcal{O}, \mathsf{so})$ *of a read-write store is said to be* multilevel correct *with respect to a* wk*-consistency criterion* α_w*,* st*-consistency criterion* α_s *and multilevel consistency constraint* φ*, iff there exists visibility relations* $\mathsf{vis}_{\mathsf{wk}}$ *and* $\mathsf{vis}_{\mathsf{st}}$ *over* H_{wk} *and* H_{st} *respectively and arbitration relation* arb *such that*

– $(H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{arb})$ *and* $(H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}}, \mathsf{arb})$ *are functionally correct,*
– $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{arb} \models \alpha_w$*,*
– $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}}, \mathsf{arb} \models \alpha_s$*, and*
– $H, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}} \models \varphi$*.*

4 Testing Multilevel Correctness of a Hybrid History

Given a read-write hybrid history $H = (\mathcal{O}, \mathsf{so})$, we want to test it for multi-level correctness with respect to weak and strong consistency criteria α_w and α_s and multilevel constraints given by φ.

We note that for the history to be correct, for every read operation that returns a value that is not $\perp$, there should exist a write operation writing the same value to the variable that was read. The *reads-from* relation associates a write operation to the read that reads its effect. Our strategy for testing the multilevel correctness of H is to enumerate all such *reads-from* relations rf, for each rf we find visibility relations $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$, respectively, containing $\mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{rf}_{\mathsf{st}}$, such that they satisfy the visibility constraints imposed by the individual consistency criteria, as well as the multilevel constraints, i.e., $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}} \models \alpha_w$, $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}} \models \alpha_s$ and $H, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}} \models \varphi$. We then check for the presence of a finite number of *bad-patterns* in these visibility relations. The presence of a bad-pattern implies that for every arbitration relation arb, there is some level $\ell \in \{\mathsf{wk}, \mathsf{st}\}$ such that either the arbitration constraint $\mathsf{vis}_\ell \upharpoonright_{\mathsf{Write}} \subseteq \mathsf{arb}$ is not satisfied, or the history $(H_\ell, \mathsf{vis}_\ell, \mathsf{arb})$ is not functionally correct.

If the history is multi-level correct, then we will find a witness consisting of a *reads-from* relation rf and visibility relations $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$ extending $\mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{rf}_{\mathsf{st}}$ such that all the constraints are satisfied and there are no bad-patterns. If the history is not multi-level correct, then for every pair of weak and strong visibility relation extending every *reads-from* relation, either some constraint is not satisfied or there exists a bad-pattern.

We present the bad-pattern characterization for multilevel correctness of a hybrid history in the next subsection. In the following subsection, we provide a procedure for computing the minimal visibility relations $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$ for a given *reads-from* relation rf that satisfies α_w, α_s and φ.

4.1 Bad Pattern characterization for multilevel correctness

We now characterize the correctness of hybrid histories based on the non-existence of certain bad patterns. This is a generalization of the bad-pattern characterization for causal consistency in [6].

Given a hybrid history, we can associate each Read with a unique write operation from the history whose effect the Read operation reads from. We call this the *reads-from* relation.

Definition 10 (Reads-From) *A* reads-from *relation* rf *over a history* $H = (\mathcal{O}, \mathsf{so})$ *is a binary relation such that*

1. $(o_i, o_j) \in \mathsf{rf} \implies o_i$ *is a* Write, o_j *is a* Read, *both on the same variable, such that the value returned by* o_j *is the value written by* o_i.
2. $(o_i, o_j) \in \mathsf{rf} \wedge (o_k, o_j) \in \mathsf{rf} \implies o_i = o_k$.
3. *For all* $o_j = \mathsf{Read}(x, val, level) \in \mathcal{O}_{\mathsf{Read}}$

$$[\exists\ o \in \mathcal{O}_{\mathsf{Write}} \text{ which writes } val \text{ to } x \implies \exists\ o_i \in \mathcal{O}_{\mathsf{Write}} : (o_i, o_j) \in \mathsf{rf}.]$$

Condition 1 associates a read operation with a write operation only if they operate on the same variable and that the return value of the read operation matches the argument of the write operation.

Condition 2 ensures that a read operation is associated with at most one write operation.

Finally, Condition 3 insists that if a Read is not related to any Write via rf, it is only because there is no matching Write in the hybrid history (i.e. a write of the same value to the same variable).

Let rf be a *reads-from* relation on a hybrid history $H = (\mathcal{O}, \mathsf{so})$. For a Read operation $o \in \mathcal{O}$, if there exists a Write operation o' such that $(o', o) \in \mathsf{rf}$, then we say that $\mathsf{rf}^{-1}(o) = o'$. If no such o' exists, we set $\mathsf{rf}^{-1}(o) = \bot$.

Further, we denote by $\mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{rf}_{\mathsf{st}}$ the *reads-from* relation restricted to H_{wk} and H_{st} respectively.

Suppose rf_ℓ is a *reads-from* relation over H_ℓ. We say that a visiblity relation vis_ℓ over H_ℓ *extends* rf_ℓ iff $\mathsf{rf}_\ell \subseteq \mathsf{vis}_\ell$. Suppose arb is an arbitration relation over H_ℓ. Then, we say that $(\mathsf{vis}_\ell, \mathsf{arb})$ *realize* rf_ℓ iff for all read operations $o \in \mathcal{O}_\ell$, $\mathsf{rf}_\ell^{-1}(o) = \mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o)$.

Given a *reads-from* relation rf_ℓ and a visibility relation vis_ℓ that extends it, we can define a conflict relation that orders all the remaining maximal related writes in $\mathit{MaxRelWrites}_{\mathsf{vis}_\ell}(o)$ of a read-operation o before the write-operation $\mathsf{rf}_\ell^{-1}(o)$. The conflict relation captures the essence of the arbitration relation for a given *reads-from* relation and a visibility relation extending it.

Definition 11 (Conflict Relation) *Let $H_\ell = (\mathcal{O}_\ell, \mathsf{so}_\ell)$ be a history. Let rf_ℓ be a* reads-from *relation over H_ℓ. Let $\mathsf{vis}_\ell \supseteq \mathsf{rf}_\ell$ be a visiblity relation over H_ℓ. We define the conflict relation for rf_ℓ and vis_ℓ, denoted $\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell)$, as the set*

$$\{(o'', o') \mid \exists o \in \mathcal{O}_\ell\restriction_{\mathsf{Read}} : o'', o' \in \mathit{MaxRelWrites}_{\mathsf{vis}}(o) \wedge o' = \mathsf{rf}_\ell^{-1}(o)\}.$$

We now define the bad patterns that characterize the correctness of the hybrid history.

Definition 12 (Bad Patterns for a hybrid history) *Let $H = (\mathcal{O}, \mathsf{so})$ be a hybrid history with weak and strong consistency criteria α_w and α_s respectively and multilevel constraints φ. Let rf be a* reads-from *relation over H. For $\ell \in \{\mathsf{wk}, \mathsf{st}\}$, let vis_ℓ be a relation over $\mathcal{O}_\ell$ with $\mathsf{vis}_\ell \supseteq \mathsf{rf}_\ell$ such that $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}} \models \alpha_w$, $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}} \models \alpha_s$ and $H, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}} \models \varphi$. We define the following bad patterns for $(H, \mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}})$. For some $\ell \in \{\mathsf{wk}, \mathsf{st}\}$:*

- $\mathsf{BADVISIBILITY}$: $\mathsf{Cyclic}(\mathsf{vis}_\ell)$
- $\mathsf{THINAIR}$: $\exists o \in \mathcal{O}_{\mathsf{Read}}\restriction_\ell$: *$o$ returns a value that is not $\bot$, but* $\mathsf{rf}_\ell^{-1}(o) = \bot$
- $\mathsf{BADINITREAD}$: $\exists o \in \mathcal{O}_{\mathsf{Read}}\restriction_\ell$: *$o$ returns $\bot$ but* $\mathit{RelWrites}_{\mathsf{vis}_\ell}(o) \neq \emptyset$
- $\mathsf{BADREAD}$: $\exists o \in \mathcal{O}_{\mathsf{Read}}\restriction_\ell$: $\mathsf{rf}_\ell^{-1}(o) \notin \mathit{MaxRelWrites}_{\mathsf{vis}_\ell}(o)$
- BADARB: $\mathsf{Cyclic}(\bigcup_{\ell \in \{\mathsf{wk}, \mathsf{st}\}} (\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell) \cup (\mathsf{vis}_\ell)_{\mathsf{Write}}))$

$\mathsf{BADVISIBILITY}$ says that one of the visibility relations has a cycle.

$\mathsf{THINAIR}$ says that there exists a read in the history which reads a non-initial value which is not written by any write operation in the hybrid history.

$\mathsf{BADINITREAD}$ says that there is a read operation on a variable which reads the initial value despite having a non-initial write to that variable in its view.

$\mathsf{BADREAD}$ says that the write operation from which the read-operation reads is not a maximal write, and there are other writes in the view of the read operation that would have overwritten the value written by that write.

BADARB says that the union of the conflict relations along visibility relation restricted to only the Write operations has a cycle indicating that there exists no total-order arb over $\mathcal{O}_{\mathsf{Write}}$, such that $(\mathsf{vis}_\ell, \mathsf{arb})$ realizes rf_ℓ.

Multi-level correctness of a hybrid history can be characterized in terms of non-existence of these bad patterns. We prove this in Appendix A.

Theorem 13 (Bad patterns characterization). *A hybrid history $H = (\mathcal{O}, \mathsf{so})$ is said to be multilevel correct with respect to weak and strong consistency criteria α_w, α_s and multilevel constraint φ iff there exists a* reads-from *relation* rf*, and relations $\mathsf{vis}_{\mathsf{wk}} \supseteq \mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}} \supseteq \mathsf{rf}_{\mathsf{st}}$ respectively over $\mathcal{O}_{\mathsf{wk}}$ and $\mathcal{O}_{\mathsf{st}}$ such that $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}} \models \alpha_w$, $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}} \models \alpha_s$ and $H, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}} \models \varphi$ and no bad pattern exists in $(H, \mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}})$.*

4.2 Constructing Minimal Visibility Relations

Suppose $H = (\mathcal{O}, \mathsf{so})$ is a hybrid history. Let α_w and α_s be the formulas defining the weak and strong consistency criteria, and let φ be the formula defining the multilevel constraints. Let $\alpha'_w = \mathit{VisBasic}(\alpha_w)$ and $\alpha'_s = \mathit{VisBasic}(\alpha_s)$.

We provide a procedure that iterates over all the possible *reads-from* relations and constructs a minimal visibility relation extending the *reads-from* relation such that it satisfies α_w, α_s and φ. The pseudo-code for the procedure is presented in Algorithm 1 and 2.

Algorithm 1 Constructing minimal visibility relations

```
1   MinVisOne(O_ℓ, so_ℓ, vis_ℓ, α_ℓ):
2       Let vis_o := vis_ℓ;
3
4       while (True):
5           Let vis_p := vis_o;
6           for τ ∈ RelTerms(α_ℓ)):
7               vis_n := vis_p ∪ τ[so_ℓ, vis_p];
8               vis_p := vis_n;
9           if (vis_n == vis_o)
10              return vis_n
11          vis_o := vis_n
12
13
14  ComputeVisSet(O_ℓ, so_ℓ, vis_ℓ, α_ℓ)
15      if total(vis) is a subformula in α_ℓ:
16          visSet_ℓ := {totvis|totvis
                         is a
                         total order over
                          O_ℓ such that
                          vis_ℓ ⊆ totvis}
17      else :
18          visSet_ℓ := {vis_ℓ}
19
20      return visSet_ℓ
21  MinVisMulti(O, so, vis_wk, vis_st, ψ)
22      if ψ ∈ {ψ^write_back, ψ^read_thru}:
23          return (vis_wk, vis_st)
24      else if ψ ∈ {ψ^write_thru}:
25          Let ℓ = st, ℓ' = wk;
26      else if ψ ∈ {ψ^read_back}:
27          Let ℓ = wk, ℓ' = st;
28
29      Let vis^o_ℓ := vis_ℓ;
30      Let vis^o_ℓ' := vis_ℓ';
31
32      if ψ ∈ {ψ^write_thru, ψ^read_back}:
33          Let vis^n_ℓ := vis^o_ℓ ∪ (vis^o_ℓ'; so) ↾ℓ;
34          Let vis^n_ℓ' := vis^o_ℓ';
35
36       if ψ ∈ {ψ^read_back}:
37          return (vis^n_ℓ, vis^n_ℓ')
38      else if ψ ∈ {ψ^write_thru}:
39          return (vis^n_ℓ', vis^n_ℓ)
40
41
42
```

In Lines 1-12 we have a method MinVisOne that takes as input a visiblity relation vis_ℓ for the history $(\mathcal{O}_\ell, \mathsf{so}_\ell)$ and constructs an extension $\mathsf{vis}_{\mathsf{n}}$ that satisfies the formula $VisBasic(\alpha_\ell)$. We achieve this by iterating over the $RelTerms$ appearing in $RelTerms(\alpha_\ell)$ (Line 6) and extending the previous visibility relation vis_p with the evaluation of the term (Line 7). We do this until we obtain a relation vis_n which we can no longer extend (Line 9). This final visibility relation vis_n extends vis_ℓ and satisfies the formula $VisBasic(\alpha_\ell)$.

In Lines 21-40, we have the procedure MinVisMulti which takes as inputs the hybrid history $(\mathcal{O}, \mathsf{so})$, visibility relations vis_wk and vis_st and an individual conjunct ψ appearing in the multilevel constraint φ. Since every visibility relation trivially satisfies ψ^{write}_{back} or ψ^{read}_{thru}, for these multilevel constraint, we simply return without modifying vis_wk or vis_st (Lines 22-23). In the remaining cases, when the multi-level constraint is either ψ^{write}_{back} or ψ^{read}_{thru}, for $\ell, \ell' \in \{\mathsf{wk}, \mathsf{st}\}$, the multilevel constraints relates the write operations visible to the operations of level ℓ in terms of the writes seen by operations of level ℓ' that have occured previously in the session. Depending on the conjunct ψ, we set ℓ and ℓ' appropriately(Lines 24-27). We then extend the visibility relation for level ℓ by relating each ℓ-operation to the Writes that have been seen by any of the ℓ'-operations prior to the ℓ-operation in its session (Line 33). The visibility relation for level ℓ' remains unchanged in this case (Line 34).

We return these extended visibility relations as a pair, where the wk visibility extension is followed by st visiblity extension (Lines 36-39).

Algorithm 2 Testing multilevel correctness of a hybrid history

43 ComputeStableExt$(\mathcal{O}, \mathsf{so}, \mathsf{vis}_\mathsf{wk}, \mathsf{vis}_\mathsf{st}, \alpha_w, \alpha_s, \varphi)$:
44   Let $\mathsf{vis}^\mathsf{o}_\mathsf{wk} := \mathsf{vis}_\mathsf{wk}$, $\mathsf{vis}^\mathsf{o}_\mathsf{st} := \mathsf{vis}_\mathsf{st}$
45
46   **while** (True):
47   Let $\mathsf{vis}^\mathsf{p}_\mathsf{wk} := \mathsf{vis}^\mathsf{o}_\mathsf{wk}$, $\mathsf{vis}^\mathsf{p}_\mathsf{st} := \mathsf{vis}^\mathsf{o}_\mathsf{st}$
48
49   Let $\mathsf{vis}^\mathsf{n}_\mathsf{wk} :=$ MinVisOne$(\mathcal{O}_\mathsf{wk}, \mathsf{so}_\mathsf{wk}, \mathsf{vis}^\mathsf{p}_\mathsf{wk}, \alpha_w)$;
50
51   Let $\mathsf{vis}^\mathsf{n}_\mathsf{st} :=$ MinVisOne$(\mathcal{O}_\mathsf{st}, \mathsf{so}_\mathsf{st}, \mathsf{vis}^\mathsf{p}_\mathsf{st}, \alpha_s)$;
52
53   **for** each subformula ψ_i in the conjunction φ:
54             $\mathsf{vis}^\mathsf{p}_\mathsf{wk} := \mathsf{vis}^\mathsf{n}_\mathsf{wk}$, $\mathsf{vis}^\mathsf{p}_\mathsf{st} := \mathsf{vis}^\mathsf{n}_\mathsf{st}$
55
56             $(\mathsf{vis}^\mathsf{n}_\mathsf{wk}, \mathsf{vis}^\mathsf{n}_\mathsf{st}) =$ MinVisMulti$(\mathcal{O}, \mathsf{so}, \mathsf{vis}^\mathsf{p}_\mathsf{wk}, \mathsf{vis}^\mathsf{p}_\mathsf{st}, \psi_i)$
57
58   **if** $\mathsf{vis}^\mathsf{n}_\mathsf{wk} = \mathsf{vis}^\mathsf{o}_\mathsf{wk}$ **and** $\mathsf{vis}^\mathsf{n}_\mathsf{st} = \mathsf{vis}^\mathsf{o}_\mathsf{st}$:
59   **return** $(\mathsf{vis}^\mathsf{n}_\mathsf{wk}, \mathsf{vis}^\mathsf{n}_\mathsf{st})$
60
61         $\mathsf{vis}^\mathsf{o}_\mathsf{wk} := \mathsf{vis}^\mathsf{n}_\mathsf{wk}$, $\mathsf{vis}^\mathsf{o}_\mathsf{st} := \mathsf{vis}^\mathsf{n}_\mathsf{st}$

62 TestMultiCorrect$(\mathcal{O}, \mathsf{so}, \alpha_w, \alpha_s, \varphi)$:
63   Let rfSet := $\{\mathsf{rf} | \mathsf{rf}$ is a reads-from relation over $(\mathcal{O}, \mathsf{so})\}$
64   **for** rf $\in$ rfSet:
65   Let $\mathsf{vis}^{\min}_\mathsf{wk} :=$ MinVisOne$(\mathcal{O}_\mathsf{wk}, \mathsf{so}_\mathsf{wk}, \mathsf{rf}_\mathsf{wk}, \alpha_w)$;
66   Let $\mathsf{visSet}_\mathsf{wk} =$ ComputeVisSet$(\mathcal{O}_\mathsf{wk}, \mathsf{so}_\mathsf{wk}, \mathsf{vis}^{\min}_\mathsf{wk}, \alpha_w)$;
67
68   Let $\mathsf{vis}^{\min}_\mathsf{st} :=$ MinVisOne$(\mathcal{O}_\mathsf{wk}, \mathsf{so}_\mathsf{st}, \mathsf{rf}_\mathsf{st}, \alpha_s)$;
69   Let $\mathsf{visSet}_\mathsf{st} :=$ ComputeVisSet$(\mathcal{O}_\mathsf{st}, \mathsf{so}_\mathsf{st}, \mathsf{vis}^{\min}_\mathsf{st}, \alpha_s)$;
70
71   **for** $\mathsf{vis}_\mathsf{wk} \in \mathsf{visSet}_\mathsf{wk}, \mathsf{vis}_\mathsf{st} \in \mathsf{visSet}_\mathsf{st}$:
72   Let $(\mathsf{vis}^\mathsf{stb}_\mathsf{wk}, \mathsf{vis}^\mathsf{stb}_\mathsf{st}) :=$ ComputeStableExt$(\mathcal{O}, \mathsf{so}, \mathsf{vis}_\mathsf{wk}, \mathsf{vis}_\mathsf{st}, \alpha_w, \alpha_s, \varphi)$;
73
74   **if** BadPatterns$(\mathcal{O}, \mathsf{so}, \mathsf{rf}, \mathsf{vis}^\mathsf{stb}_\mathsf{wk}, \mathsf{vis}^\mathsf{stb}_\mathsf{st}) ==$ NoBadPatterns:
75   **return** $(\mathsf{rf}, \mathsf{vis}^\mathsf{stb}_\mathsf{wk}, \mathsf{vis}^\mathsf{stb}_\mathsf{st})$
76
77   **return** BadHistory

In Lines 43-61 we have the procedure ComputeStableExt which takes history $(\mathcal{O}, \mathsf{so})$ a pair of visibility relations vis_wk and vis_st and extends it to $\mathsf{vis}^\mathsf{n}_\mathsf{wk}$ and $\mathsf{vis}^\mathsf{n}_\mathsf{st}$

such that they individually satisfy $\mathit{VisBasic}(\alpha_w)$ (Line 49) and $\mathit{VisBasic}(\alpha_s)$ (Line 51) respectively and jointly satisfy φ (Lines 53-56). We repeat this till we can extend these relations no longer, which implies that they have satisfied all the constraints (Lines 58-59).

The procedure $\mathsf{TestMultiCorrect}$ in Lines 62-77 takes as input a hybrid history $H = (\mathcal{O}, \mathsf{so})$ whose multilevel correctness we want to check with respect to formulas α_w , α_s and φ.

We first enumerate the set of possible *reads-from* relations on the history (line 63). We then iterate through each of the *reads-from* relations rf to see whether it can be extended to construct a minimal visibility relation satisfying all the constraints and having no bad-patterns (Lines 64-75). For each rf, we construct minimal visibility relations $\mathsf{vis}_{\mathsf{wk}}^{\min}$ and $\mathsf{vis}_{\mathsf{st}}^{\min}$ extending $\mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{rf}_{\mathsf{st}}$ respectively and satisfying the subformulas $\mathit{VisBasic}(\alpha_w)$ and $\mathit{VisBasic}(\alpha_s)$ respectively (Lines 65,68).

If α_w (resp. α_s) contains the subformula $\mathsf{total}(\mathit{vis})$, we enumerate the set of all the total orders extending $\mathsf{vis}_{\mathsf{wk}}^{\min}$ (resp. $\mathsf{vis}_{\mathsf{st}}^{\min}$) in the set $\mathsf{visSet}_{\mathsf{wk}}$ (resp. $\mathsf{visSet}_{\mathsf{st}}$) in Line 66 (resp. Line 69). If α_w (resp. α_s) does not contain the subformula $\mathsf{total}(\mathit{vis})$, then, $\mathsf{visSet}_{\mathsf{wk}}$ (resp. $\mathsf{visSet}_{\mathsf{st}}$) will contain the only minimum visibility relation extending $\mathsf{rf}_{\mathsf{wk}}$ (resp. $\mathsf{rf}_{\mathsf{st}}$), i.e., $\mathsf{vis}_{\mathsf{wk}}^{\min}$ (resp. $\mathsf{vis}_{\mathsf{st}}^{\min}$.).

For each pair of visibility relations from $\mathsf{visSet}_{\mathsf{wk}}$ and $\mathsf{visSet}_{\mathsf{st}}$ we compute their stable extensions $\mathsf{vis}_{\mathsf{wk}}^{\mathsf{stb}}$ and $\mathsf{vis}_{\mathsf{st}}^{\mathsf{stb}}$ which individually satisfy α_w and α_s, respectively, and jointly satisfy φ (line 72). We then check if this computed extension has a bad pattern (Line 74). If no bad patterns are found, we return the $(\mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}})$ as the witness.

If none of the rf can be extended to obtain the required visibility relation, we declare that the history is a bad history. We formally prove the correctness of $\mathsf{TestMultiCorrect}$ in Appendix B.

Theorem 14 (Correctness of $\mathsf{TestMultiCorrect}$ procedure). *For a hybrid read-write history $H = (\mathcal{O}, \mathsf{so})$ with weak and strong consistency criteria α_w and α_s respectively and multilevel constraints given by φ, the procedure* $\mathsf{TestMultiCorrect}$ *returns a witness* $(\mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}})$ *over H iff H is multi-level correct with respect to α_w, α_s and φ.*

4.3 Complexity

Suppose $H = (\mathcal{O}, \mathsf{so})$ is history with $|\mathcal{O}| = N$.

We note that in the procedure $\mathsf{ComputeStableExt}$, at the end of every iteration of the outer **while**-loop, the values of $\mathsf{vis}_{\mathsf{wk}}^{\mathsf{n}}$ and $\mathsf{vis}_{\mathsf{st}}^{\mathsf{n}}$ monotonically increase from the end of the previous iteration. Since they are binary relations over finite history $H = (\mathcal{O}, \mathsf{so})$ their size is upper bounded by $O(N^2)$. The time taken to evaluate each term in $\mathit{RelTerms}(\alpha_\ell)$ is again polynomial in N. Hence, the time-complexity of $\mathsf{ComputeStableExt}$ is polynomial in N, say $f(N)$.

We can observe from the procedure $\mathsf{TestMultiCorrect}$ that the main part that adds to the complexity is iterating through all the *reads-from* relation, as well as the total orders if α_w or α_s contain the subformula $\mathsf{total}(\mathit{vis})$. Suppose the number of read operations are k. Then the number of write operations is $N-k$, and there are $O((N-k)^k)$ *reads-from* relations. Since $k = O(N)$, this can be bound by $O(2^{N \log N})$. Furthermore, for a given rf, if any of the levels $\ell \in \{\mathsf{wk}, \mathsf{st}\}$ require that the visibility relation be a total order, then we iterate over all the total-orders containing the minimal visibility relation extending rf. Iterating through this requires time bounded by $O(2^{N \log N})$. Thus the worst case time complexity of the procedure is $O(f(N).2^{N \log N})$.

In general, the problem of testing the correctness of a hybrid history is in NP. We need to guess the *reads-from* relation, and then, extend it to obtain the minimal visibility relations satisfying the visibility constraints of the wk and the st consistency criteria. If the visibility relation is required to be a total order, we can guess the order. Extending this to derive fixed-point minimal visibility relations that satisfy all the visibility constraints via $\mathsf{ComputeStableExt}$ requires polynomial time. Subsequently checking for each of the bad-patterns requires polynomial time.

Note that we can reduce the testing of the correctness of a regular history (that contains only a single level of Read and Write operations) with respect to consistency criterion α to this procedure by defining the level of all the read operations to st. We set α_s to α, α_w to $\top$, and φ to $\psi^{write}_{back} \wedge \psi^{read}_{thru}$. For any *reads-from* relation rf, $\mathsf{rf}_{\mathsf{wk}} = \emptyset$. Thus $\mathsf{vis}_{\mathsf{wk}} = \emptyset$, trivially satisfying α_w as well as φ. Thus, the lower bound for testing the correctness of the hybrid history H is the complexity of testing the correctness of the H_{wk} and H_{st} with respect to their respective consistency criteria. It has been shown in [13] that testing the correctness of a read-write history with respect to sequential consistency is $\mathsf{NP\text{-}COMPLETE}$. In [6], the authors use the same reduction to show that testing the correctness with respect to causal consistency is $\mathsf{NP\text{-}COMPLETE}$. However, it can be shown that the reduction works for any consistency criterion stronger than FIFO consistency, and checking correctness with respect to such a consistency criterion is $\mathsf{NP\text{-}COMPLETE}$. Thus, in general, though testing the multi-level correctness of a hybrid history is a hard problem, the hardness is not due to the multilevel constraints but due to the constraints of the individual consistency criteria and the read-write specification.

In [6], the authors identify the class of read-write data-stores called *data-independent* data-stores whose behaviour is not dependent on the exact values written to the keys. Thus, for such stores, if there is a bad history, there is an equivalent bad *differentiated history* where a particular value is written to a particular memory location at most once. Thus, we can restrict our testing to only the correctness of differentiated histories. The authors show that the problem of testing the correctness of differentiated-histories with respect to causal consistency is solvable in polynomial time.

Note that for differentiated histories, there is exactly one *reads-from* relation which associates every Read operation with at most one Write operation which has written that value to the memory location read by the Read operation. Thus, if neither of α_w or α_s contain the subformula $\mathsf{total}(vis)$, the procedure $\mathsf{TestMultiCorrect}$ terminates in polynomial time. Thus, our procedure generalizes the result from [6] to all the consistency criteria defined in terms of the set of formulas involving only visibility, but not totality constraints. Our procedure checks the multi-level correctness of hybrid histories where the individual consistency levels do not require the visibility relation to be a total order, in polynomial time.

On the other hand, if one of α_w or α_s contains $\mathsf{total}(vis)$, then the worst case complexity remains $O(2^{N \log N})$. Once again, this does not come as a surprise, since the problem of testing the correctness of a differentiated history w.r.t. sequential consistency is not known to have a polynomial time solution.

5 Related Work

There is prior work that illustrates the need for multiple levels of consistency provided by the distributed data-stores to provide a trade off between consistency and availability/latency [2, 16, 17, 19]. The work by Kraska et al. [17] provides a transactional paradigm that allows applications to define the consistency level on data instead of transactions, and also allows the application to switch consistency guarantees at runtime. In the work by Guerraoui et al. [16], the authors provide a generic library that allows applications to request multiple responses to the same query, where the response that comes later in time is *more-correct* than the prior responses. Thus, later responses are supposed to have more knowledge of the state of the system compared to earlier responses. In our work, we have defined multilevel constraints, which can model the requirement of incremental consistency guarantees by requiring that subsequent strong responses see the effects observed by prior weak responses.

Burckhardt [8] provides a generic methodology for formalizing the specification of distributed data-stores in terms of histories, visibility and arbitration orders and provides an axiomatic characterization for consistency criteria. In our work, we have derived the specification for read-write stores based on this formalism. We have adapted this characterization to define consistency criteria as a conjunction of individual formulas. Our work extends [8] in terms of the definition of hybrid histories and provides a definition of multi-level correctness for read-write stores.

There is prior work on verifying the correctness of a behaviour with respect to individual consistency criteria. Examples include [7], which deals with verifying the correctness with respect to eventual consistency, [5], which investigates the feasibility of checking a concurrent implementation with respect to a consistency criterion that has a sequential specification, including sequential consistency, linearizability and conflict-serializability and [6], which focusses on correctness with respect to causal consistency. Our work provides a generic procedure for checking the correctness of read-write histories for all these individual consistency criteria. Further, [6] show that verification of correctness of a history with respect to causal consistency is NP-COMPLETE. However, for differentiated histories, the problem is solvable in polynomial time. In our work, we generalize the technique of computing the minimal visibility relation and checking for the absence of bad patterns for all the consistency criteria defined using our syntax. In [11], the authors models quiescent consistency using Mazurkiewicz Trace Theory to model the notion of independence between the events prior to the quiescent point. They work shows that the testing problem (which they call the membership problem) for a history is NP-COMPLETE. We cannot model quiescent consistency in our framework since we cannot model quiescent point. In [13], the authors present a detailed complexity analysis of the problem of testing the correctness of a history with respect to various consistency criteria. Our findings are consistent with the results from [13] with respect to hardness of testing consistency criteria that require the visiblity relation to be a total order. In a recent work [12], the authors provide a technique for testing the correctness of a history of a datastore with respect to a weak consistency criterion. That work also characterizes correctness in terms of minimal visibility relation extending the session order (called program-order there) and the happened-before relation (called returns-before relation in [8]). Our work applies this concept to read-write stores, where we observe that correctness with respect to visibility constraints can be satisfied by constructing a minimal visibility relation while the correctness with respect to read-write specifications and arbitration constraints can be reduced to checking for absence of certain bad patterns. In particular, our characterization of the arbitration relation in terms of the conflict relation saves the step of searching through all possible arbitration relations which is used in [12].

[15] deals with verification of *red-blue* consistency where, in a history, a subset of operations are labelled *red* while the remaining are labelled *blue*. The *blue* operations are expected to satisfy a weaker consistency criterion, while the *red* operations are supposed to satisfy a stronger consistency criterion. The effects of the strong operations and weak operations are visible to each other. We can model this by setting $\varphi = \psi_{thru}^{write} \wedge \psi_{back}^{read}$.

Our work should also be contrasted with [3], which addresses the problem of checking the consistency of CRDTs against their specifications, and covers a wide range of CRDTs including replicated sets, flags, counters, registers, etc. The relevant data structure in our case is registers, where the results are comparable (checking w.r.t. the weaker consistency criterion is tractable). However, we also consider registers with multiple consistency criteria in this paper, which is not considered there.

Another related work is [4], which uses the reads-from relation (called the *write-read* relation there) to show that testing the correctness of an execution (containing transactions) with respect to various consistency criteria like Read Committed (RC), Read Atomic (RA), Causal Consistency (CC), Prefix Consistency, and Snapshot Isolation. The key difference in the current work is that we consider histories having multiple consistency levels simultaneously while [4] considers executions consisting of transactions, under a single consistency criterion.

References

[1] Amazon DynamoDB Developer Guide (API Version 2012-08-10).: DAX and Dy-

namoDB consistency Models (2018 (Accessed Sep 26, 2019)), https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/DAX.consistency.html
[2] Bailis, P., Ghodsi, A., Hellerstein, J.M., Stoica, I.: Bolt-on causal consistency. In: Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data. pp. 761–772. SIGMOD '13, ACM, New York, NY, USA (2013), http://doi.acm.org/10.1145/2463676.2465279
[3] Biswas, R., Emmi, M., Enea, C.: On the complexity of checking consistency for replicated data types. In: Dillig, I., Tasiran, S. (eds.) Computer Aided Verification. pp. 324–343. Springer International Publishing, Cham (2019)
[4] Biswas, R., Enea, C.: On the complexity of checking transactional consistency. Proc. ACM Program. Lang. 3(OOPSLA), 165:1–165:28 (Oct 2019), http://doi.acm.org/10.1145/3360591
[5] Bouajjani, A., Emmi, M.: Analysis of recursively parallel programs. ACM Trans. Program. Lang. Syst. 35(3), 10:1–10:49 (2013), https://doi.org/10.1145/2518188
[6] Bouajjani, A., Enea, C., Guerraoui, R., Hamza, J.: On verifying causal consistency. In: Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages. pp. 626–638. POPL 2017, ACM, New York, NY, USA (2017), http://doi.acm.org/10.1145/3009837.3009888
[7] Bouajjani, A., Enea, C., Hamza, J.: Verifying eventual consistency of optimistic replication systems. In: The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014. pp. 285–296 (2014), https://doi.org/10.1145/2535838.2535877
[8] Burckhardt, S.: Principles of eventual consistency. Foundations and Trends in Programming Languages 1(1-2), 1–150 (2014), https://doi.org/10.1561/2500000011
[9] Burkhardt, S., Gotsman, A., Yang, H., Zawirski, M.: Replicated data types: specification, verification, optimality. In: The 41st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL '14, San Diego, CA, USA, January 20-21, 2014. pp. 271–284 (2014)
[10] Damien.: DynamoDB vs Cassandra (2017 (Accessed Nov 16, 2018)), https://www.beyondthelines.net/databases/dynamodb-vs-cassandra/
[11] Dongol, B., Hierons, R.M.: Decidability and complexity for quiescent consistency. In: Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science. pp. 116–125. LICS '16, ACM, New York, NY, USA (2016), http://doi.acm.org/10.1145/2933575.2933576
[12] Emmi, M., Enea, C.: Monitoring weak consistency. In: Computer Aided Verification - 30th International Conference, CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings, Part I. pp. 487–506 (2018), https://doi.org/10.1007/978-3-319-96145-3_26
[13] Furbach, F., Meyer, R., Schneider, K., Senftleben, M.: Memory model-aware testing - A unified complexity analysis. In: 14th International Conference on Application of Concurrency to System Design, ACSD 2014, Tunis La Marsa, Tunisia, June 23-27, 2014. pp. 92–101 (2014), https://doi.org/10.1109/ACSD.2014.27
[14] Gilbert, S., Lynch, N.A.: Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. SIGACT News 33(2), 51–59 (2002)
[15] Gotsman, A., Yang, H., Ferreira, C., Najafzadeh, M., Shapiro, M.: 'cause i'm strong enough: reasoning about consistency choices in distributed systems. In:

Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20 - 22, 2016. pp. 371–384 (2016), https://doi.org/10.1145/2837614.2837625

[16] Guerraoui, R., Pavlovic, M., Seredinschi, D.A.: Incremental consistency guarantees for replicated objects. In: Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation. pp. 169–184. OSDI'16, USENIX Association, Berkeley, CA, USA (2016), http://dl.acm.org/citation.cfm?id=3026877.3026891

[17] Kraska, T., Hentschel, M., Alonso, G., Kossmann, D.: Consistency rationing in the cloud: Pay only when it matters. PVLDB 2(1), 253–264 (2009), http://www.vldb.org/pvldb/2/vldb09-759.pdf

[18] Lamport, L.: How to make a multiprocessor computer that correctly executes multiprocess programs. IEEE Trans. Comput. 28(9), 690–691 (Sep 1979)

[19] Li, C., Porto, D., Clement, A., Gehrke, J., Preguiça, N., Rodrigues, R.: Making geo-replicated systems fast as possible, consistent when necessary. In: Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation. pp. 265–278. OSDI'12, USENIX Association, Berkeley, CA, USA (2012), http://dl.acm.org/citation.cfm?id=2387880.2387906

[20] Ozkan, B.K., Majumdar, R., Niksic, F., Befrouei, M.T., Weissenbacher, G.: Randomized testing of distributed systems with probabilistic guarantees. PACMPL 2(OOPSLA), 160:1–160:28 (2018), http://doi.acm.org/10.1145/3276530

[21] Perrin, M., Mostefaoui, A., Jard, C.: Causal consistency: Beyond memory. In: Proceedings of the 21st ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. pp. 26:1–26:12. PPoPP '16, ACM, New York, NY, USA (2016)

[22] Terry, D.B., Theimer, M., Petersen, K., Demers, A.J., Spreitzer, M., Hauser, C.: Managing update conflicts in bayou, a weakly connected replicated storage system. In: Proceedings of the Fifteenth ACM Symposium on Operating System Principles, SOSP 1995, Copper Mountain Resort, Colorado, USA, December 3-6, 1995. pp. 172–183 (1995), https://doi.org/10.1145/224056.224070

[23] Wolper, P.: Expressing interesting properties of programs in propositional temporal logic. In: Conference Record of the Thirteenth Annual ACM Symposium on Principles of Programming Languages, St. Petersburg Beach, Florida, USA, January 1986. pp. 184–193 (1986), https://doi.org/10.1145/512644.512661

A Correctness of the Bad Patterns Charecterization

Lemma 15. *If* rf_ℓ *is a* reads-from *relation over the history* H_ℓ *and* $(\mathsf{vis}_\ell, \mathsf{arb})$ *realize* rf_ℓ*. Then,* $\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell) \subseteq \mathsf{arb}$*.*

Proof. Suppose $(o'', o') \in \mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell)$. By definition, there exists a Read operation o such that both o', o'' are in the maximal related writes of o and $o' = \mathsf{rf}_\ell^{-1}(o)$. Since rf_ℓ is realized by $(\mathsf{vis}_\ell, \mathsf{arb})$, by definition, $\mathsf{rf}_\ell^{-1}(o) = \mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o)$. Hence o' is the effective write of o.

Now by the definition, the arbitration relation arb orders an effective write of a read operation after all the other maximal related writes of that read operation. Thus $(o'', o) \in \mathsf{arb}$. □

We now prove the correctness of Theorem 13

Let $H = (\mathcal{O}, \mathsf{so})$ be a hybrid history and let α_w and α_s respectively be the weak and strong consistency criteria. Let the multilevel constraints be defined by φ. We need to show that H is multilevel correct with respect to α_w, α_s and φ iff there exists a *reads-from* relation rf and visibility relations $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$ that extend $\mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{rf}_{\mathsf{st}}$ respectively such that $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}} \models \alpha_w$, $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}} \models \alpha_s$, $H, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}} \models \varphi$ and none of the bad patterns $\{\mathsf{BADVISIBILITY}, \mathsf{THINAIR}, \mathsf{BADINITREAD}, \mathsf{BADREAD}, \mathsf{BADARB}\}$ exists in $(H, \mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}})$.

In the proof below, and the ones that follow we shall use the following notation:

- For a read-operation $o = \mathsf{Read}(x, \mathit{val}, \mathit{level})$ in $\mathcal{O}$, we denote by $\mathit{Var}(o)$ the variable x, $\mathit{Ret}(o)$ the return value val and $\mathit{Level}(o)$ the level level.
- Similarly , for a write-operation $o = \mathsf{Write}(x, \mathit{val})$ in $\mathcal{O}$, we denote by $\mathit{Var}(o)$ the variable x, $\mathit{Args}(o)$ the input value val.

Proof. ($\Longrightarrow$): Suppose hybrid history H is correct. Then, there exists visibilty relations $\mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}}$ and arbitration relations arb such that $(H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{arb})$ and $(H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}}, \mathsf{arb})$ are functionally correct, $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{arb} \models \alpha_w$, $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}}, \mathsf{arb} \models \alpha_s$ and $H, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}} \models \varphi$.

Thus, we have

- $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}} \models \alpha_w$ and $\mathsf{vis}_{\mathsf{wk}} \restriction_{\mathsf{Write}} \subseteq \mathsf{arb}$
- $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}} \models \alpha_s$ and $\mathsf{vis}_{\mathsf{st}} \restriction_{\mathsf{Write}} \subseteq \mathsf{arb}$

For $\ell \in \{\mathsf{wk}, \mathsf{st}\}$,we set $\mathsf{rf}_\ell = \{(\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o), o) \mid o \in \mathcal{O}_{\mathsf{Read}} : \mathit{Level}(o) = \ell\}$. $\mathsf{rf} = \mathsf{rf}_{\mathsf{wk}} \cup \mathsf{rf}_{\mathsf{st}}$. By definition $\mathsf{vis}_{\mathsf{wk}}$ extends $\mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$ extends $\mathsf{rf}_{\mathsf{st}}$.

We will now show that none of the aforementioned bad patterns exists for $(H, \mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}})$.

Since H is multilevel correct, $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$ by definitions are acyclic relations. So $\mathsf{BADVISIBILITY}$ bad pattern doesn't exist.

Further, due to functional correctness of $(H_\ell, \mathsf{vis}_\ell, \mathsf{arb})$, for any read operation o of level ℓ, $\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o) = \perp$ iff $\mathit{Ret}(o) = \perp$. Since for every read operation

o, $\mathsf{rf}_\ell^{-1}(o) = \mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o)$, it follows that $\mathsf{rf}_\ell^{-1} = \bot$ iff $\mathit{Ret}(o) = \bot$. Thus, the $\mathsf{THINAIR}$ bad pattern doesn't exist.

Since $(H_\ell, \mathsf{vis}_\ell, \mathsf{arb})$ is functionally correct, for any read operation o with level ℓ such that $\mathit{Ret}(o) = \bot$, $\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o) = \bot$ which implies that $\mathit{RelWrites}_{\mathsf{vis}_\ell}(o) = \emptyset$. Hence, the $\mathsf{BADINITREAD}$ pattern doesn't exist.

For a functionally correct history H_ℓ, for any read operation o with level ℓ, if $\mathit{Ret}(o) \neq \bot$, then $\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o) \neq \bot$. This implies that $\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o) \in \mathit{MaxRelWrites}_{\mathsf{vis}_\ell}(o)$. But we have set $\mathsf{rf}_\ell^{-1}(o) = \mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o)$. Thus $\mathsf{rf}_\ell^{-1}(o) \in \mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o)$. Hence, the $\mathsf{BADREAD}$ pattern doesn't exist.

By construction, rf_ℓ is realized by $(\mathsf{vis}_\ell, \mathsf{arb})$. Hence from lemma 15 for $\ell \in \{\mathsf{wk}, \mathsf{st}\}$, $\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell) \subseteq \mathsf{arb}$. Due to functional correctness of $(H_\ell, \mathsf{vis}_\ell, \mathsf{arb})$, we have $\mathsf{vis}_\ell \upharpoonright_{\mathsf{Write}} \subseteq \mathsf{arb}$. Hence $\bigcup_{\ell \in \{\mathsf{wk},\mathsf{st}\}} (\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell) \cup \mathsf{vis}_\ell \upharpoonright_{\mathsf{Write}}) \subseteq \mathsf{arb}$. By definition arb is a total order. Thus BADARB would imply a cycle in arb which is not true. Hence, the BADARB pattern doesn't exist.

This completes one side direction of the proof.

$(\Longleftarrow)$: Suppose there exists a $(\mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}})$ such that $\mathsf{vis}_{\mathsf{wk}} \supseteq \mathsf{rf}_{\mathsf{wk}}$, $\mathsf{vis}_{\mathsf{st}} \supseteq \mathsf{rf}_{\mathsf{st}}$, $H, \mathsf{vis}_{\mathsf{wk}} \models \alpha_w$ and $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}} \models \alpha_s$, $H, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}} \models \varphi$ and $(H, \mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}})$ does not have any bad-patterns. To show that H is multi-level correct, we need to show that there exists an arbitration relation arb such that $\mathsf{vis}_\ell \upharpoonright_{\mathsf{Write}} \subseteq \mathsf{arb}$ and $(H_\ell, \mathsf{vis}_\ell, \mathsf{arb})$ is functionally correct for $\ell \in \{\mathsf{wk}, \mathsf{st}\}$.

We first construct the arbitration relation arb. Since the BADARB bad pattern doesn't exist, $\bigcup_{\ell \in \{\mathsf{wk},\mathsf{st}\}} (\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell) \cup \mathsf{vis}_\ell \upharpoonright_{\mathsf{Write}})$ is an acyclic relation. We set arb to be a topological sort of this acyclic relation along with the Write operations from o, not appearing in this acyclic relations. Thus arb is a total order. By construction, $\mathsf{vis}_\ell \upharpoonright_{\mathsf{Write}} \subseteq \mathsf{arb}$ for $\ell \in \{\mathsf{wk}, \mathsf{st}\}$. From this, and what is given we can conclude that $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{arb} \models \alpha_w$ and $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}}, \mathsf{arb} \models \alpha_s$.

We now only need to show that for each $\ell \in \{\mathsf{wk}, \mathsf{st}\}$, $(H_\ell, \mathsf{vis}_\ell, \mathsf{arb})$ is functionally correct.

Let o be a read operation with level ℓ. Suppose $\mathit{MaxRelWrites}_{\mathsf{vis}_\ell}(o) = \emptyset$. Then $\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o) = \bot$. Since $\mathsf{rf}_\ell \subseteq \mathsf{vis}_\ell$, $\mathsf{rf}_\ell^{-1}(o) = \bot$. Since $\mathsf{THINAIR}$ bad pattern doesn't exist, it has to be the case that $\mathit{Ret}(o) = \bot$. Thus, if $\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o) = \bot$ then $\mathit{Ret}(o) = \bot$. Conversely, suppose $\mathit{Ret}(o) = \bot$. Then, since $\mathsf{BADINITREAD}$ bad pattern doesn't exist, $\mathit{RelWrites}_{\mathsf{vis}_\ell}(o) = \emptyset$. Thus, by definition, $\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}\ell}(o) = \bot$. Thus, we can conclude that $\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o) = \bot \iff \mathit{Ret}(o) = \bot$.

Suppose $\mathit{MaxRelWrites}_{\mathsf{vis}_\ell}(o) \neq \emptyset$. Since $\mathsf{BADINITREAD}$ bad pattern doesn't exist, $\mathit{Ret}(o) \neq \bot$. Further, since $\mathsf{THINAIR}$ badpattern doesn't exist, $\mathsf{rf}_\ell^{-1}(o) \neq \bot$. Let $\mathsf{rf}_\ell^{-1}(o) = o'$. Since $\mathsf{BADREAD}$ bad pattern doesn't exist, $\mathsf{rf}_\ell^{-1}(o) = o' \in \mathit{MaxRelWrites}_{\mathsf{vis}_\ell}(o)$. For any $o'' \in \mathit{MaxRelWrites}_{\mathsf{vis}_\ell}(o)$ we have $(o'', o') \in \mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell)$. Now, by construction of arb, we have $\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell) \subseteq \mathsf{arb}$.

Thus, for any $o'' \in \mathit{MaxRelWrites}_{\mathsf{vis}_\ell}(o)$, $o'' \xrightarrow{\mathsf{arb}} o'$. Thus by definition, $\mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o) = o'$. However, since $o' = \mathsf{rf}_\ell^{-1}(o)$, by definition of a *reads-from* relation, $\mathit{Ret}(o) = \mathit{Args}(o')$. Thus $o' = \mathit{EffWrite}^{\mathsf{arb}}_{\mathsf{vis}_\ell}(o)$ wrote the value read by o.

Since o is an arbitrary Read operation with level ℓ in H, what we have shown holds for all Read operation with level ℓ. Hence $(H_\ell, \mathsf{vis}_\ell, \mathsf{arb})$ is functionally correct. Hence H is multi-level correct. $\square$

B Correctness of the testing procedure

We will first prove a set of lemmas with respect to the termination and the correctness of the helper procedures.

Lemma 16 (Termination of Helper functions). *For a given hybrid history, and a given visibility relations over the history, the methods* MinVisOne*,* MinVisMulti *and* ComputeStableExt *terminate.*

Proof. We first observe that MinVisMulti terminates since it doesn't have any loops. The visibility relations it outputs is a superset of the input visibility relations.

We will now show the termination of MinVisOne. Let $\mathsf{vis}_\mathsf{n}^{i,j}$ denote the value of vis_n at the end of the j^{th} iteration of the inner **for**-loop within the i^{th} iteration of the outer **while**-loop. Let $\mathsf{vis}_\mathsf{n}^{i}$ denote the value of vis_n at the end of the i^{th} iteration of the outer **while**-loop.

We note that for $j > 0$, $\mathsf{vis}_\mathsf{n}^{i,j} \supseteq \mathsf{vis}_\mathsf{n}^{i,j-1}$ since we only keep extending vis_n inside the inner **for**-loop by adding to it the result of evaluation of the *RelTerms* in α_ℓ. $\mathsf{vis}_\mathsf{n}^{i,0} = \mathsf{vis}_\mathsf{n}^{i-1}$. If $|\mathit{RelTerms}(\alpha_\ell)| = k$, then, $\mathsf{vis}_\mathsf{n}^{i} = \mathsf{vis}_\mathsf{n}^{i,k}$. Since $\mathsf{vis}_\mathsf{n}^{i,k} \supseteq \mathsf{vis}_\mathsf{n}^{i,0}$, it follows that $\mathsf{vis}_\mathsf{n}^{i} \supseteq \mathsf{vis}_\mathsf{n}^{i-1}$. At the end of the outer-while loop we check if $\mathsf{vis}_\mathsf{n} = \mathsf{vis}_\mathsf{o}$ which is equivalent to checking $\mathsf{vis}_\mathsf{n}^{i} = \mathsf{vis}_\mathsf{n}^{i-1}$. If true, the function returns. Since $\mathsf{vis}_\mathsf{n} \subseteq \mathcal{O} \times \mathcal{O}$, and since $\mathcal{O}$ is a finite set, it will be the case that $\mathsf{vis}_\mathsf{n} = \mathsf{vis}_\mathsf{o}$ after a finite number of iterations. Hence the procedure terminates.

In case of ComputeStableExt , we note that it obtains the new values for $\mathsf{vis}_\mathsf{wk}^\mathsf{n}$ and $\mathsf{vis}_\mathsf{st}^\mathsf{n}$ individually by invoking the procedure MinVisOne, which returns a relation that is a superset of the input visibility relation. Similarly, in the inner **for**-loop, we obtain the new values for the pair $(\mathsf{vis}_\mathsf{wk}^\mathsf{n}, \mathsf{vis}_\mathsf{st}^\mathsf{n})$ by calling MinVisMulti, which returns visibility relations that are supersets of the corresponding input relations. Thus, at the end of each iteration of **while**-loop, either the values of $\mathsf{vis}_\mathsf{wk}^\mathsf{n}$ and $\mathsf{vis}_\mathsf{st}^\mathsf{n}$ are the same as their values at the end of the previous iteration of the **while**-loop, or they are a superset of their values at the end of the the previous generation. Since both $\mathsf{vis}_\mathsf{wk}^\mathsf{n}$ and $\mathsf{vis}_\mathsf{wk}^\mathsf{n}$ are binary relations over $\mathcal{O}_\mathsf{wk}$ and $\mathcal{O}_\mathsf{st}$, their maximal size is bound by $|\mathcal{O}|^2$. Thus, the iterations of the outer while loop are bounded by $O(|\mathcal{O}|^2)$ iterations. Hence ComputeStableExt terminates. $\square$

Theorem 17 (Termination of Testing Procedure). *For any given hybrid-history H, and consistency criteria α_w, α_s and multilevel constraints φ, the procedure* TestMultiCorrect *terminates.*

Proof. Since H is a finite history, the number of *reads-from* relations that can be defined over it are finite. Further, for each rf from the set of *reads-from* relations, the extensions $\mathsf{vis}_\mathsf{wk}^{\min}$ and $\mathsf{vis}_\mathsf{st}^{\min}$ are finite. In the worst case when both α_w as well as α_s contain the subsformula $\mathsf{total}(vis)$, the sizes of $\mathsf{visSet_{wk}}$ and $\mathsf{visSet_{st}}$ is finite. Since the procedures called within the inner **for**-loop, i.e. ComputeStableExt and BadPatterns, terminate, the inner **for**-loop (Lines 71-75) will iterate only for a finite number of times.

Thus, the procedure will terminate when either it has found a witness rf, $\mathsf{vis_{wk}^{stb}}$ and $\mathsf{vis_{st}^{stb}}$ for the correctness of the hybrid history, or when it has iterated over all the finitely many *reads-from* relation. □

Lemma 18 (Correctness of MinVisOne). *Let* vis_ℓ *be a visiblity relation over the history* H_ℓ*. Let* α_ℓ *be a consistency criteria. Let* $\mathsf{vis} := \mathsf{MinVisOne}(H_\ell, \mathsf{vis}_\ell, \alpha_\ell)$*. Then* $H_\ell, \mathsf{vis} \models \mathit{VisBasic}(\alpha_\ell)$*.*

Proof. We will denote the value of $\mathsf{vis_n}$ at the end of the i^{th} iteration of the outer while-loop as $\mathsf{vis_n^{\mathit{i}}}$. We shall denote the value of $\mathsf{vis_n}$ at the end of the the j^{th} iteration in the i^{th} iteration of the inner for loop as $\mathsf{vis_n^{\mathit{i,j}}}$.

Let vis be the value returned by MinVisOne at the end of the k^{th} iteration of the outer **while**-loop. Then, $\mathsf{vis} = \mathsf{vis_n^{\mathit{k}}}$.

Note that $\mathsf{vis_o}$ is the value of $\mathsf{vis_n}$ at the end of the previous iteration of while loop. Thus $\mathsf{vis_o^{\mathit{k}}} = \mathsf{vis_n^{\mathit{k-1}}}$. Further since $\mathsf{vis_o} = \mathsf{vis_n}$ for the function to return, we have $\mathsf{vis_n^{\mathit{k}}} = \mathsf{vis_o^{\mathit{k}}} = \mathsf{vis_n^{\mathit{k-1}}}$. Let $\mathsf{vis_n^{\mathit{k,0}}}$ denote the value of $\mathsf{vis_n}$ at the beginning of the inner for-loop. Then $\mathsf{vis_n^{\mathit{k,0}}} = \mathsf{vis_n^{\mathit{k-1}}}$. Suppose $\mathit{RelTerms}(\alpha)$ has N terms where the n^{th} term is denoted by τ_n, then, we can see that for $j \in [1, \ldots, N]$, $\mathsf{vis_n^{\mathit{k,j}}} = \mathsf{vis_n^{\mathit{k,j-1}}} \cup \tau_j[\mathsf{so}_\ell, \mathsf{vis_n^{\mathit{k,j-1}}}]$. Thus, we can conclude that $\tau_j[\mathsf{so}_\ell, \mathsf{vis_n^{\mathit{k,j-1}}}] \subseteq \mathsf{vis_n^{\mathit{k,j}}}$.

Also, we can note that $\mathsf{vis_n^{\mathit{k-1}}} = \mathsf{vis_n^{\mathit{k,0}}} \subseteq \mathsf{vis_n^{\mathit{k,1}}} \subseteq \cdots \subseteq \mathsf{vis_n^{\mathit{k,N}}} = \mathsf{vis_n^{\mathit{k}}}$. Since, $\mathsf{vis_n^{\mathit{k-1}}} = \mathsf{vis_n^{\mathit{i}}}$, this implies that for each $j \in [0, \ldots, N]$, $\mathsf{vis_n^{\mathit{k,j}}} = \mathsf{vis_n^{\mathit{k}}} = \mathsf{vis}$.

Thus, for each $j \in [1, \ldots, N]$, we have $\tau_j[\mathsf{so}_\ell, \mathsf{vis}] \subseteq \mathsf{vis}$. Hence, $\mathsf{so}_\ell, \mathsf{vis} \models \bigwedge_{\tau_j \in \mathit{RelTerms}(\alpha_\ell)} (\tau_j \subseteq vis)$. But by definition, $\bigwedge_{\tau_j \in \mathit{RelTerms}(\alpha_\ell)} (\tau_j \subseteq vis) = \mathit{VisBasic}(\alpha)$. Hence $\mathsf{so}_\ell, \mathsf{vis} \models \mathit{VisBasic}(\alpha)$ which implies that $H_\ell, \mathsf{vis} \models \mathit{VisBasic}(\alpha)$.

□

Lemma 19 (Monotonicity of ***RelTerms*****).**

Let $H = (\mathcal{O}, \mathsf{so})$ *be a history and let* vis *and* vis' *be two visibility relation over* H *such that* $\mathsf{vis} \subseteq \mathsf{vis}'$*. Then for any term* $\tau \in \mathit{RelTerms}$*,*
$\tau[so := \mathsf{so}, vis := \mathsf{vis}] \subseteq \tau[so := \mathsf{so}, vis := \mathsf{vis}']$*.*

Proof. We shall write $\tau[\mathsf{so}, \mathsf{vis}]$ to mean $\tau[so := \mathsf{so}, vis := \mathsf{vis}]$ and $\tau[so, \mathsf{vis}']$ to mean $\tau[so := \mathsf{so}, vis := \mathsf{vis}']$.

We will prove this by induction over the number of compositions in the term τ. The base case is when there are no compositions. We have two cases $\tau = so$ and $\tau = vis$.

In the former case, the result trivially follows. In the latter case, the result follows since it is given that $\mathsf{vis} \subseteq \mathsf{vis}'$.

Suppose the result holds for all τ with fewer than n compositions. We now consider a $\tau = \tau'; \tau''$ where both τ' and τ'' have at most $n-1$ compositions. Now

$\tau[\mathsf{so},\mathsf{vis}] = \tau'[\mathsf{so},\mathsf{vis}];\tau''[\mathsf{so},\mathsf{vis}]$. By induction hypothesis, $\tau'[\mathsf{so}, vis] \subseteq \tau'[\mathsf{so},\mathsf{vis}']$ and $\tau''[\mathsf{so}, vis] \subseteq \tau''[\mathsf{so},\mathsf{vis}']$. Since $A \subseteq A'$ and $B \subseteq B'$ implies $A;B \subseteq A';B'$ we can conclude that $\tau'[\mathsf{so},\mathsf{vis}];\tau''[\mathsf{so},\mathsf{vis}] \subseteq \tau'[\mathsf{so},\mathsf{vis}'];\tau''[\mathsf{so},\mathsf{vis}'] = \tau[\mathsf{so},\mathsf{vis}']$. Thus the result is true for a τ with n compositions.

Hence, the result is true for all $\tau \in \mathit{RelTerms}$

Lemma 20 (Minimality of $\mathsf{MinVisOne}$). *Let vis_ℓ be a visibility relation over the history H_ℓ. Let α_ℓ be axioms defining the consistency criteria. Let vis' be a visibility relation over H_ℓ such that $\mathsf{vis}_\ell \subseteq \mathsf{vis}'$ and $H_\ell, \mathsf{vis}' \models \mathit{VisBasic}(\alpha_\ell)$.*

Then if, $\mathsf{vis} := \mathsf{MinVisOne}(H_\ell, \mathsf{vis}_\ell, \alpha_\ell)$, we have $\mathsf{vis} \subseteq \mathsf{vis}'$.

Proof. As before, we will denote the value at the end of the i^{th} iteration of the outer while-loop as $\mathsf{vis}_\mathsf{n}^i$. We shall denote the value of vis_n at the end of the the j^{th} iteration in the i^{th} iteration of the inner for loop as $\mathsf{vis}_\mathsf{n}^{i,j}$. We set $\mathsf{vis}_\mathsf{n}^0 = \mathsf{vis}_\mathsf{n}^{0,0} = \mathsf{vis}_\ell$.

Let $|\mathit{RelTerms}(\alpha)| = n$ and let τ_j denote the j^{th} member of $\mathit{RelTerms}(\alpha)$.

We will first show that for $j \in [1, \dots, n]$, If $\mathsf{vis}_\mathsf{n}^{i,j-1} \subseteq \mathsf{vis}'$ then $\mathsf{vis}_\mathsf{n}^{i,j} \subseteq \mathsf{vis}'$. Note that $\mathsf{vis}_\mathsf{n}^{i,j} = \mathsf{vis}_\mathsf{n}^{i,j-1} \cup \tau_j[\mathsf{so}_\ell, \mathsf{vis}_\mathsf{n}^{i,j-1}]$. By assumption, $\mathsf{vis}_\mathsf{n}^{i,j-1} \subseteq \mathsf{vis}'$. By lemma 19, $\tau_j[\mathsf{so}_\ell, \mathsf{vis}_\mathsf{n}^{i,j-1}] \subseteq \tau_j[\mathsf{so}_\ell, \mathsf{vis}']$. Thus, we can conclude that $\mathsf{vis}_\mathsf{n}^{i,j} \subseteq \mathsf{vis}'$.

Since for any i, $\mathsf{vis}_\mathsf{n}^{i,0} \subseteq \mathsf{vis}_\mathsf{n}^{i,1} \subseteq \cdots \subseteq \mathsf{vis}_\mathsf{n}^{i,n} = \mathsf{vis}_\mathsf{n}^i$, we can conclude that if $\mathsf{vis}_\mathsf{n}^{i,0} \subseteq \mathsf{vis}'$ then, $\mathsf{vis}_\mathsf{n}^i \subseteq \mathsf{vis}'$. Finally note that $\mathsf{vis}_\mathsf{n}^i = \mathsf{vis}_\mathsf{n}^{i+1,0}$. Thus, if $\mathsf{vis}_\mathsf{n}^i \subseteq \mathsf{vis}'$ then $\mathsf{vis}_\mathsf{n}^{i+1} \subseteq \mathsf{vis}'$. Finally we note that $\mathsf{vis}_\mathsf{n}^{0,0} = \mathsf{vis}_\ell \subseteq \mathsf{vis}'$. Thus for all $i > 0$, $\mathsf{vis}_\mathsf{n}^i \subseteq \mathsf{vis}'$. Since the value vis returned by $\mathsf{MinVisOne}$ is the value of vis_n at the end of some iteration i, it follows that $\mathsf{vis} \subseteq \mathsf{vis}'$.

Lemma 21 (Correctness of $\mathsf{MinVisMulti}$). *Let $H = (\mathcal{O}, \mathsf{so})$ be a hybrid history and let vis_wk and vis_st respectively be visibility relations over H_wk and H_st. Let ψ be a subformula in φ.*

Let $(\mathsf{vis}_\mathsf{wk}^{ret}, \mathsf{vis}_\mathsf{st}^{ret}) = \mathsf{MinVisMulti}(\mathcal{O}, \mathsf{so}, \mathsf{vis}_\mathsf{wk}, \mathsf{vis}_\mathsf{st}, \psi)$.

Then, $\mathsf{vis}_\mathsf{wk} \subseteq \mathsf{vis}_\mathsf{wk}^{ret}$, $\mathsf{vis}_\mathsf{st} \subseteq \mathsf{vis}_\mathsf{st}^{ret}$ and $H, \mathsf{vis}_\mathsf{wk}^{ret}, \mathsf{vis}_\mathsf{st}^{ret} \models \psi$.

Proof. Note that if $\psi \in \{\psi_{back}^{write}, \psi_{thru}^{read}\}$, we set $\mathsf{vis}_\mathsf{wk}^{ret} = \mathsf{vis}_\mathsf{wk}$ and $\mathsf{vis}_\mathsf{st}^{ret} = \mathsf{vis}_\mathsf{st}$. Since in this case $\psi = \top$, since trivially $H, \mathsf{vis}_\mathsf{wk}^{ret}, \mathsf{vis}_\mathsf{st}^{ret} \models \top$, the lemma is proved for these cases.

We now prove the result for the cases when $\psi = \psi_{thru}^{write}$.

For $\psi = \psi_{thru}^{write}$, we note that $\ell = \mathsf{st}$ and $\ell' = \mathsf{wk}$ and $\mathsf{vis}_\mathsf{st}^\mathsf{o} = \mathsf{vis}_\mathsf{st}$ and $\mathsf{vis}_\mathsf{wk}^\mathsf{o} = \mathsf{vis}_\mathsf{wk}$.

Now $\mathsf{vis}_\mathsf{st}^\mathsf{n} = \mathsf{vis}_\mathsf{st}^\mathsf{o} \cup (\mathsf{vis}_\mathsf{wk}^\mathsf{o}; \mathsf{so}) \restriction_\mathsf{st}$ and $\mathsf{vis}_\mathsf{wk}^\mathsf{n} = \mathsf{vis}_\mathsf{wk}^\mathsf{o}$. Thus, we can rewrite this as $\mathsf{vis}_\mathsf{st}^\mathsf{n} = \mathsf{vis}_\mathsf{st}^\mathsf{o} \cup (\mathsf{vis}_\mathsf{wk}^\mathsf{n}; \mathsf{so}) \restriction_\mathsf{st}$. Thus, $(\mathsf{vis}_\mathsf{wk}^\mathsf{n}; \mathsf{so}) \restriction_\mathsf{st} \subseteq \mathsf{vis}_\mathsf{st}^\mathsf{n}$. Hence, we can write that $\mathsf{so}, \mathsf{vis}_\mathsf{wk}^\mathsf{n}, \mathsf{vis}_\mathsf{st}^\mathsf{n} \models \psi_{thru}^{write}$.

The case where $\psi = \psi_{back}^{read}$ is proved with similar reasoning by interchanging wk and st.

Lemma 22 (Minimality of $\mathsf{MinVisMulti}$). *Let $H = (\mathcal{O}, \mathsf{so})$ be a hybrid history and let vis_wk and vis_st be visibility relations over histories H_wk and H_st. Let ψ be a subformula in the hybrid constraint φ.*

Suppose there exists $\mathsf{vis}'_\mathsf{wk}$ and $\mathsf{vis}'_\mathsf{st}$ over H_wk and H_st respectively such that

– $\mathsf{vis}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$
– $\mathsf{vis}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$
– $H, \mathsf{vis}'_{\mathsf{wk}}, \mathsf{vis}'_{\mathsf{st}} \models \psi$.

Then, if $(\mathsf{vis}^{ret}_{\mathsf{wk}}, \mathsf{vis}^{ret}_{\mathsf{st}}) = \mathsf{MinVisMulti}(\mathcal{O}, \mathsf{so}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}}, \psi)$*, it is the case that* $\mathsf{vis}^{ret}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$ *and* $\mathsf{vis}^{ret}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$

Proof. When $\psi \in \{\psi^{write}_{back}, \psi^{read}_{thru}\}$, since $\mathsf{vis}^{ret}_{\mathsf{wk}} = \mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}^{ret}_{\mathsf{st}} = \mathsf{vis}_{\mathsf{st}}$, it follows that $\mathsf{vis}^{ret}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$ and $\mathsf{vis}^{ret}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$.

We will now prove the result for the case when $\psi = \psi^{write}_{thru}$.

Suppose ψ is ψ^{write}_{thru}. We have $\mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}} = \mathsf{vis}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$ and $\mathsf{vis}^{\mathsf{o}}_{\mathsf{st}} = \mathsf{vis}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$

Since $\mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$, we have $\mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}}; \mathsf{so} \subseteq \mathsf{vis}'_{\mathsf{wk}}; \mathsf{so}$. From this, we have $(\mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}}; \mathsf{so}) \restriction_{\mathsf{st}} \subseteq (\mathsf{vis}'_{\mathsf{wk}}; \mathsf{so}) \restriction_{\mathsf{st}}$. This implies $\mathsf{vis}^{\mathsf{o}}_{\mathsf{st}} \cup (\mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}}; \mathsf{so}) \restriction_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}} \cup (\mathsf{vis}'_{\mathsf{wk}}; \mathsf{so}) \restriction_{\mathsf{st}}$ since $\mathsf{vis}^{\mathsf{o}}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$. Since $H, \mathsf{vis}'_{\mathsf{wk}}, \mathsf{vis}'_{\mathsf{st}} \models \psi^{write}_{thru}$ implies $(\mathsf{vis}'_{\mathsf{wk}}; \mathsf{so}) \restriction_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$ we can conclude that
$\mathsf{vis}^{\mathsf{o}}_{\mathsf{st}} \cup (\mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}}; \mathsf{so}) \restriction_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$. However $\mathsf{vis}^{\mathsf{o}}_{\mathsf{st}} \cup (\mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}}; \mathsf{so}) \restriction_{\mathsf{st}} = \mathsf{vis}^{\mathsf{n}}_{\mathsf{st}}$. Thus $\mathsf{vis}^{\mathsf{n}}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$. Since $\mathsf{vis}^{ret}_{\mathsf{st}} = \mathsf{vis}^{\mathsf{n}}_{\mathsf{st}}$ and $\mathsf{vis}^{ret}_{\mathsf{wk}} = \mathsf{vis}_{\mathsf{wk}}$, it follows that $\mathsf{vis}^{ret}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$ and $\mathsf{vis}^{ret}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$. Hence this case is proved.

The proof for the case $\psi = \psi^{read}_{back}$ follows via similar reasoning by interchanging wk and st.

Lemma 23 (Correctness of $\mathsf{ComputeStableExt}$). *Let H be a hybrid history and let $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$ respectively be a visibility relations over H_{wk} and H_{st}. Let α_w and α_s respectively be the weak and strong consistency criteria and let φ be the multilevel constraints. Let*
$(\mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}})$ *be the return value obtained from*
$\mathsf{ComputeStableExt}(\mathcal{O}, \mathsf{so}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}}, \alpha_w, \alpha_s, \varphi)$. *Then*

– $H_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}} \models \mathit{VisBasic}(\alpha_w)$
– $H_{\mathsf{st}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \mathit{VisBasic}(\alpha_s)$
– $H, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \varphi$

Proof. We note that the value returned by $\mathsf{ComputeStableExt}$ $(\mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}})$ are respectively the values of variables $\mathsf{vis}^{\mathsf{n}}_{\mathsf{wk}}$ and $\mathsf{vis}^{\mathsf{n}}_{\mathsf{st}}$ at the end of the outer while loop, when they respectively match the values $\mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}}$ and $\mathsf{vis}^{\mathsf{o}}_{\mathsf{st}}$. Furthermore, $\mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{o}}_{\mathsf{st}}$ were the values of $\mathsf{vis}^{\mathsf{n}}_{\mathsf{wk}}$ and $\mathsf{vis}^{\mathsf{n}}_{\mathsf{st}}$ at the end of the previous iteration of the outer **while**-loop.

We will replay the iteration of the outer-while loop which returned the value. Here, we note that $\mathsf{vis}^{\mathsf{p}}_{\mathsf{wk}} = \mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}}$ and $\mathsf{vis}^{\mathsf{p}}_{\mathsf{st}} = \mathsf{vis}^{\mathsf{o}}_{\mathsf{st}}$.

Let the value computed in line 49 by invoking the method $\mathsf{MinVisOne}$ be denoted as $\mathsf{vis}^{1}_{\mathsf{wk}}$. Now $\mathsf{vis}^{1}_{\mathsf{wk}} \subseteq \mathsf{vis}^{\mathsf{p}}_{\mathsf{wk}} = \mathsf{vis}^{\mathsf{o}}_{\mathsf{wk}}$. Further, by Lemma 18 $H_{\mathsf{wk}}, \mathsf{vis}^{1}_{\mathsf{wk}} \models \mathit{VisBasic}(\alpha_w)$.

Let the value computed in line 51 by invoking the method $\mathsf{MinVisOne}$ be denoted as $\mathsf{vis}^{1}_{\mathsf{st}}$. Now $\mathsf{vis}^{1}_{\mathsf{st}} \subseteq \mathsf{vis}^{\mathsf{p}}_{\mathsf{st}} = \mathsf{vis}^{\mathsf{o}}_{\mathsf{st}}$. Further, $H_{\mathsf{st}}, \mathsf{vis}^{1}_{\mathsf{st}} \models \mathit{VisBasic}(\alpha_s)$.

Let $\varphi = \psi_2 \wedge \psi_3$

We let $(\mathsf{vis}^{i}_{\mathsf{wk}}, \mathsf{vis}^{i}_{\mathsf{st}}) = \mathsf{MinVisMulti}(\mathcal{O}, \mathsf{so}, \mathsf{vis}^{i-1}_{\mathsf{wk}}, \mathsf{vis}^{i-1}_{\mathsf{st}}, \psi_i)$ for $i \in [2, 3]$

By lemma 21, for $i \in [2, 3]$, we have

- $\mathsf{vis}^{i-1}_{\mathsf{wk}} \subseteq \mathsf{vis}^{i}_{\mathsf{wk}}$
- $\mathsf{vis}^{i-1}_{\mathsf{st}} \subseteq \mathsf{vis}^{i}_{\mathsf{st}}$
- $H, \mathsf{vis}^{i}_{\mathsf{wk}}, \mathsf{vis}^{i}_{\mathsf{st}} \models \psi_i$

And $\mathsf{vis}^{3}_{\mathsf{wk}} = \mathsf{vis}^{\mathsf{n}}_{\mathsf{wk}}$, $\mathsf{vis}^{3}_{\mathsf{st}} = \mathsf{vis}^{\mathsf{n}}_{\mathsf{st}}$
Thus, for $\ell \in \{\mathsf{wk}, \mathsf{st}\}$ we have $\mathsf{vis}^{\mathsf{o}}_{\ell} \subseteq \mathsf{vis}^{1}_{\ell} \subseteq \mathsf{vis}^{2}_{\ell} \subseteq \ldots \mathsf{vis}^{\mathsf{n}}_{\ell} = \mathsf{vis}^{\mathsf{o}}_{\ell}$.
From this we can conclude that $\mathsf{vis}^{i}_{\ell} = \mathsf{vis}^{\mathsf{n}}_{\ell} = \mathsf{vis}^{\mathsf{stb}}_{\ell}$ for $i \in [1, 2, 3]$.
Since

- $H_{\mathsf{wk}}, \mathsf{vis}^{1}_{\mathsf{wk}} \models \mathit{VisBasic}(\alpha_w)$.
- $H_{\mathsf{st}}, \mathsf{vis}^{1}_{\mathsf{st}} \models \mathit{VisBasic}(\alpha_s)$.
- $H, \mathsf{vis}^{2}_{\mathsf{wk}}, \mathsf{vis}^{2}_{\mathsf{st}} \models \psi_2$
- $H, \mathsf{vis}^{3}_{\mathsf{wk}}, \mathsf{vis}^{3}_{\mathsf{st}} \models \psi_3$

we can conclude that

- $H_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}} \models \mathit{VisBasic}(\alpha_w)$.
- $H_{\mathsf{st}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \mathit{VisBasic}(\alpha_s)$.
- $H, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \psi_2 \wedge \psi_3 = \varphi$

This proves the result. □

Lemma 24 (Minimality of ComputeStableExt). *Let H be a hybrid history and let $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$ respectively be a visibility relations over H_{wk} and H_{st}. Let α_w, α_s be weak and strong consistency criteria and let φ be multilevel constraints.Let $(\mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}}) := \mathsf{ComputeStableExt}(\mathcal{O}, \mathsf{so}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}}, \alpha_w, \alpha_s, \varphi)$. If there exists visibility relations $\mathsf{vis}'_{\mathsf{wk}}$ and $\mathsf{vis}'_{\mathsf{st}}$ over H_{wk} and H_{st} respectively such that*

- $\mathsf{vis}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$
- $\mathsf{vis}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$
- $H_{\mathsf{wk}}, \mathsf{vis}'_{\mathsf{wk}} \models \mathit{VisBasic}(\alpha_w)$
- $H_{\mathsf{st}}, \mathsf{vis}'_{\mathsf{st}} \models \mathit{VisBasic}(\alpha_s)$
- $H, \mathsf{vis}'_{\mathsf{wk}}, \mathsf{vis}'_{\mathsf{st}} \models \varphi$

Then $\mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$ *and* $\mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$

Proof. The proof for this follows the line of argument showing the minimality of MinVisOne. We note that at each step we compute extensions of the weak and strong visibility relations via invoking MinVisOne and MinVisMulti.

From Lemmas 20 and 22, the output produced by these procedures $\mathsf{vis}^{ret}_{\mathsf{wk}}$ and $\mathsf{vis}^{ret}_{\mathsf{st}}$ from inputs $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$ respectively will satisfy $\mathsf{vis}^{ret}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$ and $\mathsf{vis}^{ret}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$ whenever it is the case that $\mathsf{vis}_{\mathsf{wk}} \subseteq \mathsf{vis}'_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}} \subseteq \mathsf{vis}'_{\mathsf{st}}$.

Thus, even the final output $(\mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}})$ will satisfy the containment. □

We shall prove another interesting result pertaining to the conflict relations of two visibility relations extending the same *reads-from* relations, with one visibility relation contained inside another.

Lemma 25. *Let* rf_ℓ *be a* reads-from *relation over the history H_ℓ and let* vis_ℓ *and* vis'_ℓ *be two visibility relations over H_ℓ, both extending* rf_ℓ *such that* $\mathsf{vis}_\ell \subseteq \mathsf{vis}'_\ell$. *Then,* $\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell) \subseteq (\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}'_\ell) \cup (\mathsf{vis}'_\ell\restriction_{\mathsf{Write}})^+$

Proof. Suppose $(o'', o') \in \mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}_\ell)$. That implies that there exists a read operation o such that $o'', o' \in \mathit{MaxRelWrites}_{\mathsf{vis}_\ell}(o)$ and $o' = \mathsf{rf}_\ell^{-1}(o)$.

Since $\mathsf{vis}_\ell \subseteq \mathsf{vis}'_\ell$, it implies that $o'', o' \in \mathit{RelWrites}_{\mathsf{vis}'_\ell}(o)$.

We consider two cases.

Suppose $o'' \in \mathit{MaxRelWrites}_{\mathsf{vis}'_\ell}(o)$, then by definition, $(o'', o') \in \mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}'_\ell)$. Therefore, in this case $(o'', o') \in (\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}'_\ell) \cup (\mathsf{vis}'_\ell \upharpoonright_{\mathsf{Write}})^+$.

Suppose $o'' \notin \mathit{MaxRelWrites}_{\mathsf{vis}'_\ell}(o)$. Then, this implies that o'' is not a maximal write in the vis'_ℓ view of o restricted to its related writes. Thus, either $o'' \xrightarrow{\mathsf{vis}'_\ell \upharpoonright \mathsf{Write}} o'$ or there exists a path from $o'' \xrightarrow{\mathsf{vis}'_\ell \upharpoonright \mathsf{Write}} o_1 \xrightarrow{\mathsf{vis}'_\ell \upharpoonright \mathsf{Write}} \ldots \xrightarrow{\mathsf{vis}'_\ell \upharpoonright \mathsf{Write}} o_k \xrightarrow{\mathsf{vis}'_\ell \upharpoonright \mathsf{Write}} o'''$ where $o''' \in \mathit{MaxRelWrites}_{\mathsf{vis}'_\ell}(o)$ and each of $o_1, \ldots, o_k \in \mathit{RelWrites}_{\mathsf{vis}'_\ell}(o)$. In this case too, either $o''' = o'$ or $(o''', o') \in \mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}'_\ell)$. Thus even in this case $(o'', o') \in (\mathsf{CF}(\mathsf{rf}_\ell, \mathsf{vis}'_\ell) \cup \mathsf{vis}'_\ell \upharpoonright_{\mathsf{Write}})^+$. $\square$

With this we can now prove the correctness of Theorem 14. We need to prove the following:

For a hybrid read-write history $H = (\mathcal{O}, \mathsf{so})$, weak and strong consistency criteria α_w, α_s and multilevel constraints φ, the procedure TestMultiCorrect returns a witness $(\mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{st}})$ over H iff H is multi-level correct with respect to α_w, α_s and φ.

Proof. Suppose the hybrid history H is multi-level correct with respect to the consistency criteria α_w, α_s, and multilevel constraints φ. Then, by theorem 13, there exists a *reads-from* relation rf and visibility relations $\mathsf{vis}'_{\mathsf{wk}}$ and $\mathsf{vis}'_{\mathsf{st}}$ over H_{wk} and H_{st} extending $\mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{rf}_{\mathsf{st}}$ respectively such that

- $H_{\mathsf{wk}}, \mathsf{vis}'_{\mathsf{wk}} \models \alpha_w$
- $H_{\mathsf{st}}, \mathsf{vis}'_{\mathsf{st}} \models \alpha_s$
- $H, \mathsf{vis}'_{\mathsf{wk}}, \mathsf{vis}'_{\mathsf{st}} \models \varphi$

Since the procedure, iterates through all possible *reads-from* relation, if it returns before encountering the rf mentioned earlier, then we have nothing to prove. Suppose it does not return. Then, we will consider the iteration with the *reads-from* relation being rf.

Note that since $\mathsf{vis}_{\mathsf{wk}}^{\min}$ and $\mathsf{vis}_{\mathsf{st}}^{\min}$ are extensions of $\mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{rf}_{\mathsf{st}}$ via the procedure MinVisOne, by Lemma 20, we have $\mathsf{vis}_{\mathsf{wk}}^{\min} \subseteq \mathsf{vis}'_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}^{\min} \subseteq \mathsf{vis}'_{\mathsf{st}}$.

Now, suppose for $\mathsf{total}(\mathit{vis})$ is a subformula in α_w. Then $\mathsf{vis}'_{\mathsf{wk}}$ is a total order. Similarly if $\mathsf{total}(\mathit{vis})$ is a subformula in α_s, then $\mathsf{vis}'_{\mathsf{st}}$ is a total order.

For $\ell \in \{\mathsf{wk}, \mathsf{st}\}$,since we iterate through all the total orders extending $\mathsf{vis}_\ell^{\min}$, if the procedure returns before the iteration reaches vis'_ℓ, then, there is nothing to prove. Suppose, the procedure returns with none of the prior total orders extending $\mathsf{vis}_\ell^{\min}$. Then we consider the case where the iterating variable vis_ℓ is the total order vis'_ℓ.

On the other hand, if $\mathsf{total}(\mathit{vis})$ is not a subformula in α_w or α_s, then we would set the corresponding vis_ℓ to $\mathsf{vis}_\ell^{\min}$. In both these cases, we can notice that $\mathsf{vis}_\ell \subseteq \mathsf{vis}'_\ell$.

Now, we obtain $(\mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}})$ by invoking $\mathsf{ComputeStableExt}$ with $\mathsf{vis}_{\mathsf{wk}}$ and $\mathsf{vis}_{\mathsf{st}}$. By Lemma 23, $H_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}} \models \mathit{VisBasic}(\alpha_w)$ $H_{\mathsf{st}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \mathit{VisBasic}(\alpha_s)$ and $H, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \varphi$.

Further, by Lemma 24, for $\ell \in \{\mathsf{wk}, \mathsf{st}\}$, $\mathsf{vis}^{\mathsf{stb}}_{\ell} \subseteq \mathsf{vis}'_{\ell}$. Which implies that if $\mathsf{total}(\mathit{vis})$ is a subformula in the ℓ-consistency criteria then, $\mathsf{vis}^{\mathsf{stb}}_{\ell}$ is a total order as vis'_{ℓ} is.

From this, we can conclude that $H_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}} \models \alpha_w$, $H_{\mathsf{st}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \alpha_s$ in addition to $H, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \varphi$.

Now we check $H, \mathsf{rf}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}}$ for bad patterns.

Note that, $(H, \mathsf{rf}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}})$ cannot have BADVISIBILITY, THINAIR, BADINITREAD or BADREAD bad patterns, since that would imply the existence of those bad patterns in $(H, \mathsf{rf}, \mathsf{vis}'_{\mathsf{wk}}, \mathsf{vis}')$ since $\mathsf{vis}^{\mathsf{stb}}_{\ell}$ is contained within vis'_{ℓ} for $\ell \in \{\mathsf{wk}, \mathsf{st}\}$.

We will show by contradiction that BADARB bad pattern doesn't exist for $(H, \mathsf{rf}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}})$ doesn't exist. Suppose this bad pattern did exist. Then, there is a cycle $C = o_1 \xrightarrow{\sigma_1} o_2 \xrightarrow{\sigma_2} \ldots \xrightarrow{\sigma_n} o_1$ where each σ_i is one of $\mathsf{CF}(\mathsf{rf}_{\ell}, \mathsf{vis}^{\mathsf{stb}}_{\ell})$ or $\mathsf{vis}^{\mathsf{stb}}_{\ell} \restriction_{\mathsf{Write}}$ for $\ell \in \{\mathsf{wk}, \mathsf{st}\}$

Note that since $\mathsf{vis}^{\mathsf{stb}}_{\ell} \subseteq \mathsf{vis}'_{\ell}$ we have $\mathsf{vis}^{\mathsf{stb}}_{\ell} \restriction_{\mathsf{Write}} \subseteq \mathsf{vis}'_{\ell} \restriction_{\mathsf{Write}}$. Hence in the Cycle C above, we can rewrite the edge $o_i \xrightarrow{\mathsf{vis}^{\mathsf{stb}}_{\ell}\restriction_{\mathsf{Write}}} o_{i+1}$ by $o_i \xrightarrow{\mathsf{vis}'_{\ell}\restriction_{\mathsf{Write}}} o_{i+1}$.

Further from Lemma 25, we have $\mathsf{CF}(\mathsf{rf}_{\ell}, \mathsf{vis}^{\mathsf{stb}}_{\ell}) \subseteq (\mathsf{CF}(\mathsf{rf}_{\ell}, \mathsf{vis}'_{\ell}) \cup (\mathsf{vis}'_{\ell} \restriction_{\mathsf{Write}})^{+}$. Which means that the any edge $o_i \xrightarrow{\mathsf{CF}(\mathsf{rf}_{\ell}, \mathsf{vis}^{\mathsf{stb}}_{\ell})} o_{i+1}$ in the cycle C can be replaced by a path $o_i \xrightarrow{\sigma'_1} \ldots \xrightarrow{\sigma'_{n'}} o_{i+1}$ where each σ'_k is either $\mathsf{CF}(\mathsf{rf}_{\ell}, \mathsf{vis}'_{\ell})$ or $\mathsf{vis}'_{\ell} \restriction_{\mathsf{Write}}$. Thus, we get a cycle C' from C whose edges comprise of $\mathsf{CF}(\mathsf{rf}_{\ell}, \mathsf{vis}'_{\ell})$ and $\mathsf{vis}'_{\ell} \restriction_{\mathsf{Write}}$ for $\ell \in \{\mathsf{wk}, \mathsf{st}\}$. Thus, BADARB bad pattern exists for $(H, \mathsf{rf}, \mathsf{vis}'_{\mathsf{wk}}, \mathsf{vis}'_{\mathsf{st}})$, which is a contradiction. Thus, if H is correct, then we have proved that the procedure TestMultiCorrect produces a satisfying witness.

Conversely we will show that if TestMultiCorrect produces a satisfying witness $(\mathsf{rf}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}})$ then the hybrid history H is multi-level correct.

Suppose $(\mathsf{rf}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}})$ is the witness. Then, $\mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}$ and $\mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}}$ are the visibility relations returned by the procedure ComputeStableExt. Further, none of the bad patterns exist for $(H, \mathsf{rf}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}})$.

By lemma23, we know that

- $H_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}} \models \mathit{VisBasic}(\alpha_w)$
- $H_{\mathsf{st}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \mathit{VisBasic}(\alpha_s)$
- $H, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{wk}}, \mathsf{vis}^{\mathsf{stb}}_{\mathsf{st}} \models \varphi$.

For $\ell \in \{\mathsf{wk}, \mathsf{st}\}$, in order to show that $H_{\ell}, \mathsf{vis}_{\ell} \models \alpha_{\ell}$, we need to show that if $\mathsf{total}(\mathit{vis})$ is a subformula of α_{ℓ}, then, $\mathsf{vis}^{\mathsf{stb}}_{\ell}$ is a total order

Note that if $\mathsf{total}(\mathit{vis})$ is a subformula of α_{ℓ}, then the iterating variable vis_{ℓ} would have been a total order (line 71). By lemma 23, we know that $\mathsf{vis}_{\ell} \subseteq \mathsf{vis}^{\mathsf{stb}}_{\ell}$. Suppose $\mathsf{vis}_{\ell} \subsetneq \mathsf{vis}^{\mathsf{stb}}_{\ell}$, it implies that $\mathsf{vis}^{\mathsf{stb}}_{\ell}$ has at least one additional edges between the operations of $\mathcal{O}_{\ell}$ over what is present in vis_{ℓ}. However, since vis_{ℓ} is a total order, it implies that any additional edges introduce a cycle in $\mathsf{vis}^{\mathsf{stb}}_{\ell}$. But this is not the case since that would imply BADVISIBILITY for $\mathsf{vis}^{\mathsf{stb}}_{\ell}$.

Hence it has to be the case that $\mathsf{vis}_\ell^{\mathsf{stb}} = \mathsf{vis}_\ell$. Thus, $\mathsf{vis}_\ell^{\mathsf{stb}}$ is a total order. This proves that if total is a subformula in the consistency criteria for level ℓ, then, $H_\ell, \mathsf{vis}_\ell^{\mathsf{stb}} \models \mathsf{total}(vis)$. Hence $H_\ell, \mathsf{vis}_\ell^{\mathsf{stb}} \models \alpha_\ell$.

Thus, we can conclude that there exists a *reads-from* relation rf and weak and strong visibility relations $\mathsf{vis}_{\mathsf{wk}}^{\mathsf{stb}}$ and $\mathsf{vis}_{\mathsf{st}}^{\mathsf{stb}}$ extending $\mathsf{rf}_{\mathsf{wk}}$ and $\mathsf{rf}_{\mathsf{st}}$ respectively such that $H_{\mathsf{wk}}, \mathsf{vis}_{\mathsf{wk}}^{\mathsf{stb}} \models \alpha_w$, $H_{\mathsf{st}}, \mathsf{vis}_{\mathsf{st}}^{\mathsf{stb}} \models \alpha_s$, $H, \mathsf{vis}_{\mathsf{wk}}^{\mathsf{stb}}, \mathsf{vis}_{\mathsf{st}}^{\mathsf{stb}} \models \varphi$, and none of the bad patterns exist for $(H, \mathsf{rf}, \mathsf{vis}_{\mathsf{wk}}^{\mathsf{stb}}, \mathsf{vis}_{\mathsf{st}}^{\mathsf{stb}})$. By theorem 13, this implies that the hybrid history H is multi-level correct with respect to $\alpha_w, \alpha_s, \varphi$ $\square$