On Constructing a Byzantine Linearizable SWMR Atomic Register from SWSR Atomic Registers

Presented by,
Manaswini Piduguralla

Authors:
Ajay D. Kshemkalyani, Manaswini Piduguralla,
Sathya Peri, Anshuman Misra

ICDCN 2025

Table of Contents

Research Objective

Constructing a Byzantine Linearizable single-writer multi-reader register (SWMR) Atomic Register from single-writer single-reader (SWSR) Atomic Registers

Register Class

Register Class[1]:

```
1.    public T read() {
2.        return value;
3.    }
4.    public void write(Tv) {
5.        value = v;
6.    }
```

[1] Maurice Herlihy and Nir Shavit. *The Art of Multiprocessor Programming, Revised Reprint*. 1st. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012. ISBN: 9780123973375, 9780123977953.

Atomic Register

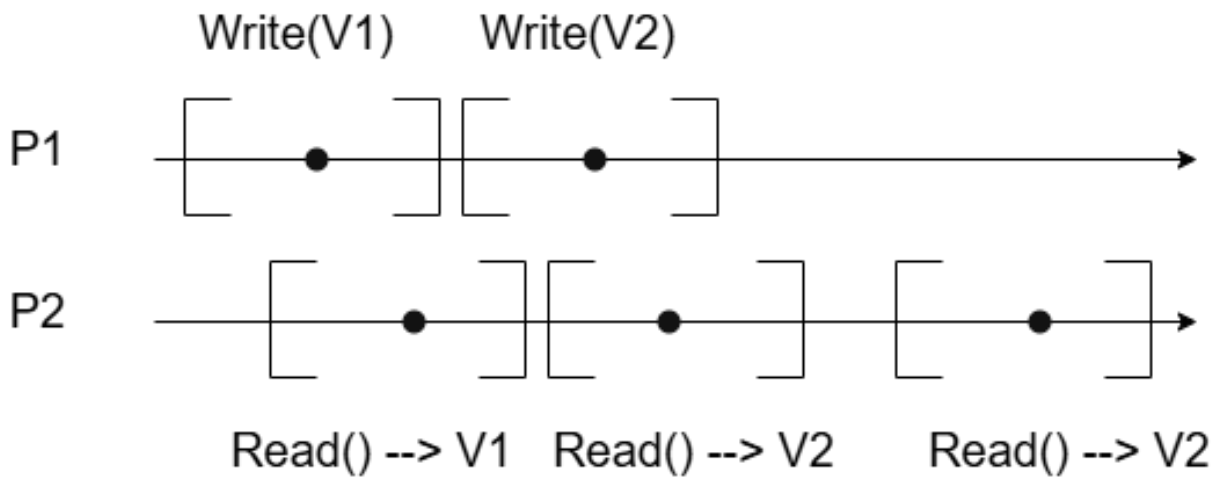

Figure: Atomic Register Example

Histories

History:[2] An execution of a system is modeled by a history, which is a finite sequence of operation (e.g., reads, writes) invocation and response events.

1. A history H defines a partial order $\prec_H$ on operations. $op_1 \prec_H op_2$ if the response event of op_1 precedes the invocation event of op_2 in H.
2. op_1 is concurrent with op_2 if neither precedes the other.

[2] Maurice Herlihy and Nir Shavit. *The Art of Multiprocessor Programming, Revised Reprint*. 1st. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2012. ISBN: 9780123973375, 9780123977953.

Linearizability of a Concurrent History

A *linearization* of a concurrent history H of object o is a sequential history H' such that:

1. After removing some pending operations from H and completing others by adding matching responses, it contains the same invocations and responses as H',
2. H' preserves the partial order $\prec_H$, and
3. H' satisfies o's sequential specification.

Linearizable Histories Example

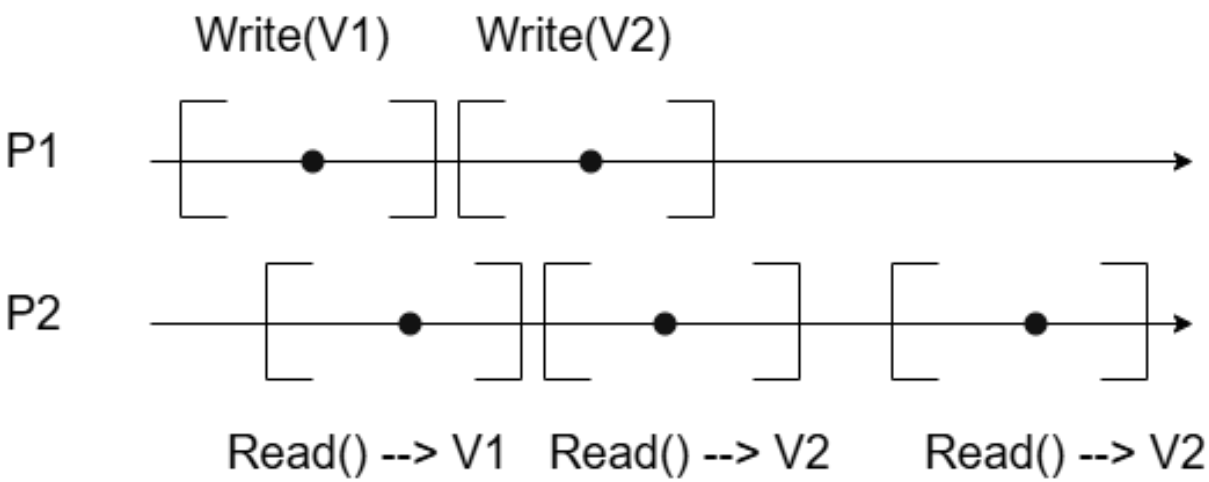

Figure: Regular Register Example

1. R write(v1) P1
2. R Read() P2
3. R OK() P1
4. R write(v2) P1
5. R OK(V1) P2
6. R Read() P2
7. R OK() P1
8. R OK(V2) P2
9. R Read() P2
10. R OK(V2) P2

Linearizable Histories Example

History H:

1. R write(v1) P1
2. R Read() P2
3. R OK() P1
4. R write(v2) P1
5. R OK(V1) P2
6. R Read() P2
7. R OK() P1
8. R OK(V2) P2
9. R Read() P2
10. R OK(V2) P2

History H‘:

1. R write(v1) P1
2. R OK() P1
3. R Read() P2
4. R OK(V1) P2
5. R write(v2) P1
6. R OK() P1
7. R Read() P2
8. R OK(V2) P2
9. R Read() P2
10. R OK(V2) P2

Faults

Fault Types:

- **Crash Fault:** A fault in which a process abruptly stops working and ceases all operations. The process does not recover or perform any additional actions.
- **Byzantine Fault:** A fault where a process may exhibit arbitrary behavior, including sending conflicting or incorrect information to other processes, often due to malicious or unpredictable issues.

Byzantine linearizable

Byzantine linearization of a concurrent history[3]: A history H is Byzantine linearizable if there exists a history H' such that $H'|_{correct} = H|_{correct}$ and H' is linearizable.

- $H|_{correct}$ denote the projection of history H to all correct processes.
- An object supports Byzantine linearizable executions if all of its executions are Byzantine linearizable.

[3] Shir Cohen and Idit Keidar. "Tame the Wild with Byzantine Linearizability: Reliable Broadcast, Snapshots, and Asset Transfer". In: *35th International Symposium on Distributed Computing, DISC 2021*. Ed. by Seth Gilbert. Vol. 209. LIPIcs. 2021, 18:1–18:18. DOI: 10.4230/LIPICS.DISC.2021.18. URL: https://doi.org/10.4230/LIPIcs.DISC.2021.18.

Register Linearizability

Register Linearizability[4]**:** In a system with Byzantine process failures, an implementation of a SWMR register is linearizable if and only if the following holds. If the writer is not malicious, then:

- (Reading a "current" value) If a read operation R by a process that is not malicious returns the value v then:
  - there is a write v operation that immediately precedes R or is concurrent with R, or
  - $v = v_0$ (the initial value) and no write operation precedes R.
- (No "new-old" inversion) If two read operations R and R' by processes that are not malicious return values v_k and $v_{k'}$, respectively, and R precedes R', then $k \leq k'$.

[4] Xing Hu and Sam Toueg. "On Implementing SWMR Registers from SWSR Registers in Systems with Byzantine Failures". In: *36th International Symposium on Distributed Computing, DISC 2022, , Augusta, Georgia, USA*. vol. 246. LIPIcs. 2022, 36:1–36:19. DOI: 10.4230/LIPICS.DISC.2022.36.

Example of Byzantine Linearizable Register

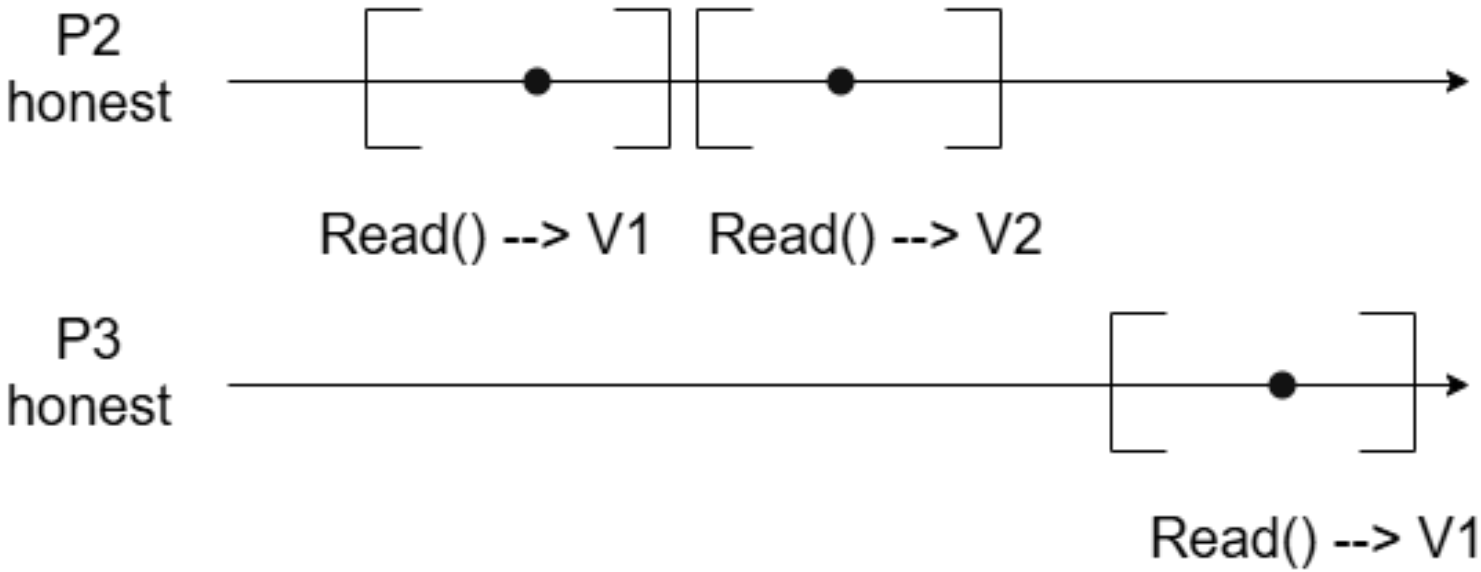

Figure: Byzantine Linearizable Register History

Example of Byzantine linearizable

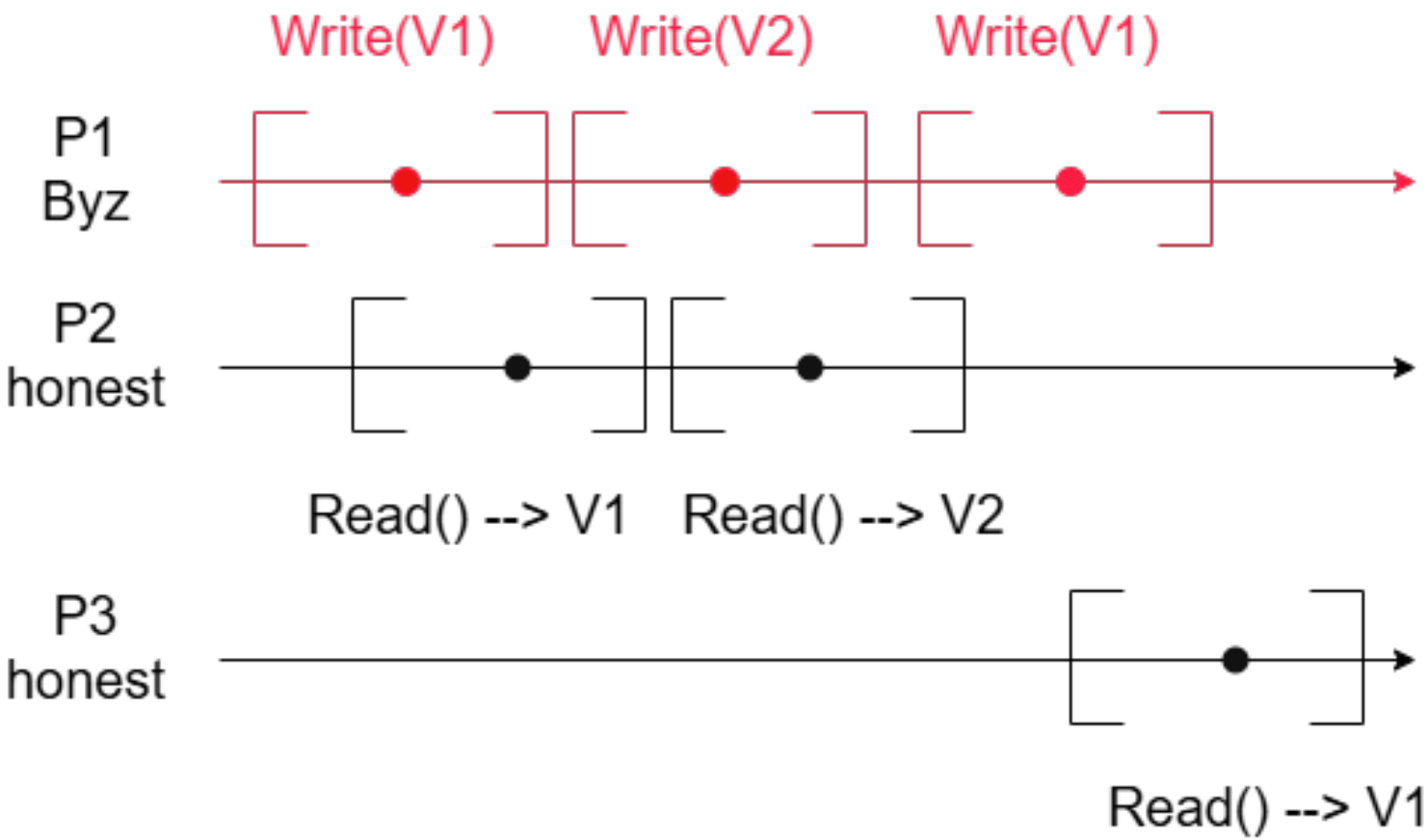

Figure: Byzantine Linearizable Register History

Stronger Byzantine Linearizable SWMR Object

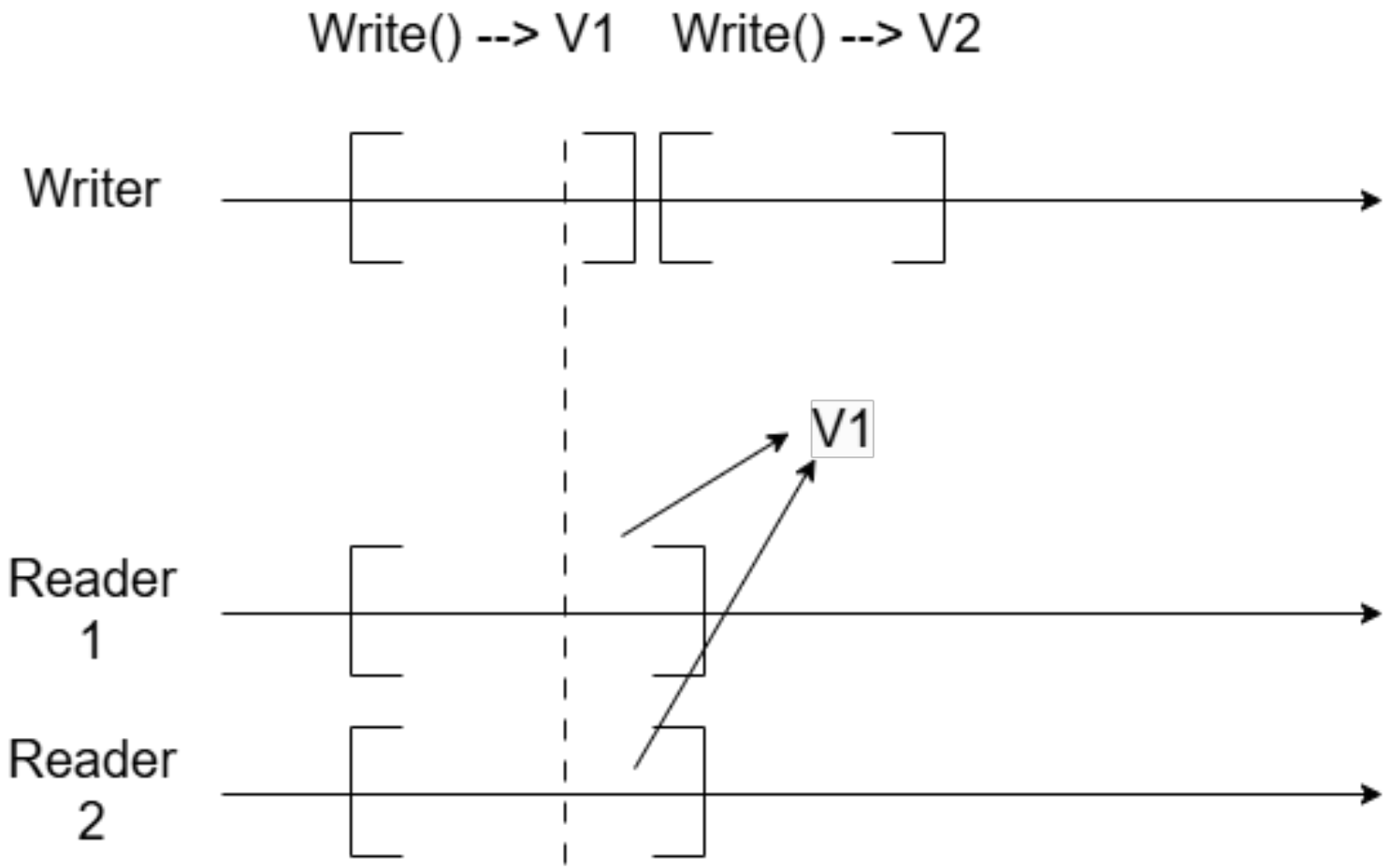

Figure: Byzantine SWMR Object

Stronger Byzantine Linearizable SWMR Object

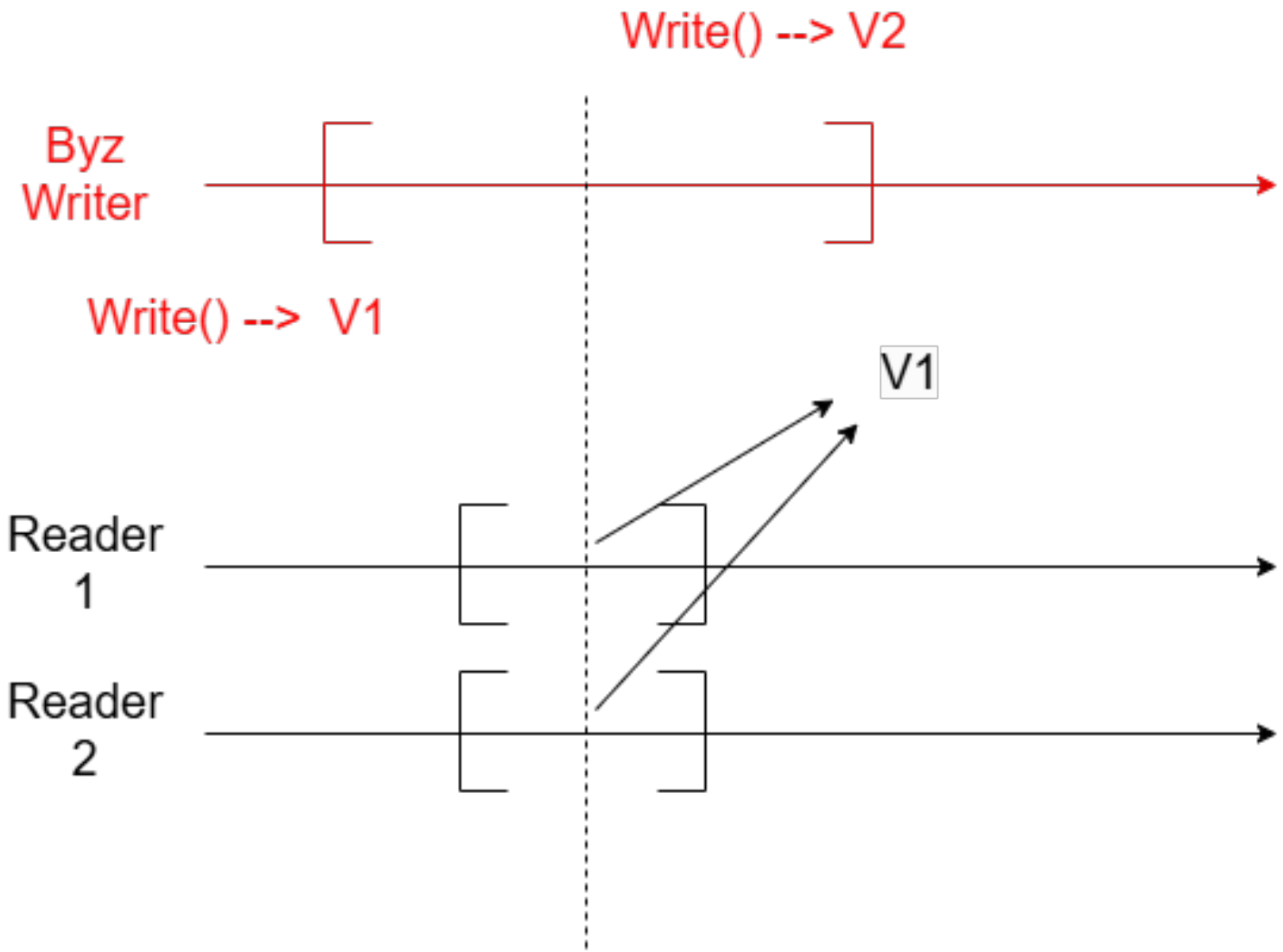

Figure: Byzantine SWMR Object

pseudo-correct operation

pseudo-correct operation definition

A pseudo-correct operation is an operation executed by a process where it results in a value being returned to honest processes, is indistinguishable from the steps of a correct operation.

Example Byzantine linearizable

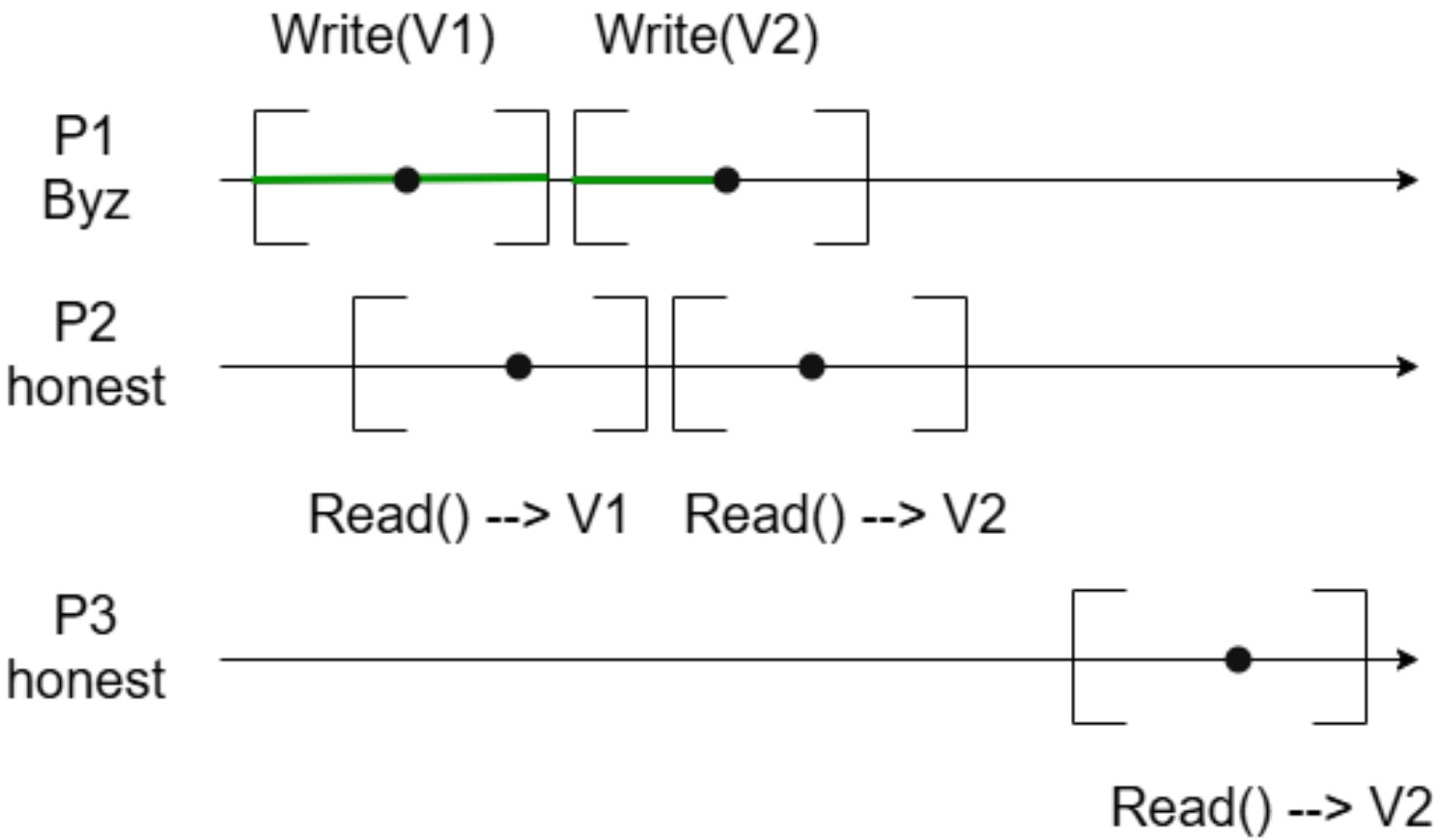

Figure: Byzantine Linearizable Register

Register Linearizability

Register Linearizability In a system with Byzantine process failures, an implementation of a SWMR register is linearizable if and only if the following two properties are satisfied in a Byzantine linearization of a concurrent history.

1. **Reading a current value:** When a read operation R by a non-Byzantine process returns the value v:
   1. if $v = v_0$ then no correct or pseudo-correct write operation precedes R
   2. else if $v \neq v_0$ then v was written by the correct or pseudo-correct write operation that immediately precedes R.
2. **No "new-old" inversions:** If read operations R and R' by non-Byzantine processes return values v^i and v^j, respectively, and R precedes R', then $i \leq j$.

Conclusion

- We have studied Byzantine tolerant construction of a SWMR atomic register from SWSR atomic registers
- We have introduced the notion of a pseudo-correct write operation by a Byzantine writer, which has the effect of a correct write operation
- A definition of stronger Byzantine linearizable register object by non-trivially taking into account Byzantine behavior of the writer and readers

Thank you