Axiomatic Semantics
An execution consists of shared memory access events and relations among the events.

Axioms provides certain consistency constraints on the events and relations.

A consistent execution satisfies all the constraints.

Program semantics is represented by a set of consistent executions.

Program behavior is represented by the set of outcomes of the consistent executions.

An execution is represented as a graph where nodes denote events and edges denote relations among the events.
- An event $e = \langle uid, i, \ell \rangle$ is a tuple where
  - $uid \in \mathbb{N}$ is an unique identifier,
  - $i \in Tid \cup \{0\}$ is thread identifier, and
  - $\ell$ is a label
- Label $\ell = \langle op, loc, rVal, wVal \rangle$ is a tuple where $op$, $loc$, $rVal$, $wVal$ denote operation, location, read value, and written value respectively.
- $e.uid$, $e.tid$, $e.op$, $e.loc$, $e.rVal$, and $e.wVal$ denote the identifier, thread id, operation, location, read value, and written value of event $e$.
- Representation of an event:
  - Read event $e = R(X, v)$ as $e.wVal = \perp$
  - Write event $e = W(X, v)$ as $e.rVal = \perp$
  - Update event $e = U(X, v_r, v_w)$
  - Fence event $e = F$ as $e.loc = \perp$, $e.rVal = \perp$, and $e.wVal = \perp$
Each relation relates a pair of events.

Two types of relations: primitive and derived

Primitive relations are components of execution. For example

- program-order (po)
- read-from rf

Derived relations are derived from other primitive or derived relations. For example

- happens-before (hb)
An execution \( \langle E, \text{po}, \text{rf} \rangle \) is a tuple where:

- \( E \) is a finite set of events
- \( \text{po} \) is a partial order on \( E \). Denotes the syntactic order among events.
- \( \text{rf} \) is a binary relation on \( E \) such that for all \((w, r) \in \text{rf}\):
  - \( w.\text{op} = W \) or \( w.\text{op} = U \)
  - \( r.\text{op} = R \) or \( r.\text{op} = U \)
  - \( w.\text{loc} = r.\text{loc} \)
  - \( w.\text{wVal} = r.\text{rVal} \)
Let $G = \langle E, po, rf \rangle$ be an execution graph.

- $G.E = E$, set of events
- $G.po = po$, set of po edges
- $G.rf = rf$, set of rf edges

Moreover,

- $G.R = \{a \mid a.op = R\}$
- $G.W = \{a \mid a.op = W\}$
- $G.U = \{a \mid a.op = U\}$
- $G.F = \{a \mid a.op = F\}$
- $G.WU = G.W \cup G.U$
- $G.RU = G.R \cup G.U$
- $G.E^i = \{a \mid a \in G.E \land a.Tid = i\}$
- $G.E_x = \{a \mid a \in G.E \land a.loc = x\}$
- \ldots
Execution of a thread $i$ results in respective sequential graph $G^i$.

(Silent)

$$c, s \stackrel{e}{\rightarrow} c', s'$$

$$c, s, G \Rightarrow c', s', G$$

(Non-silent)

$$c, s \stackrel{\ell}{\rightarrow} c', s'$$

$$a = \langle n, i, \ell \rangle$$

$$\forall e \in G.E \ e.uid \neq n$$

$$c, s, G \Rightarrow c', s', G'$$

where

$$G'.E = G.E \cup \{a\}$$

$$G'.po = G.po \cup (G.E^i \times \{a\})$$

$$G'.rf = G.rf$$
G is an execution graph of a program P if \( G^i \) is an execution of \( P(i) \) for every thread \( i \in \text{Tid} \).

**Complete Execution Graph**

An execution graph is complete if every read or event reads from a write or update event.

\[ \forall r \in G.\text{RU}. \exists w \in G.\text{WU}. \ (w, r) \in G.\text{rf} \]

Alternatively, \( \text{codom}(G.\text{rf}) = G.\text{RU} \)
Consistent Execution

Let MM be a set of consistency constraints. 
Cons\((G, MM) = true\) if a complete execution graph \(G\) satisfies all MM constraints.

Semantics of a Program

Semantics of a program \(P\) in consistency model MM is is denoted by the set of of its MM consistent executions.

\[
\llbracket P, MM \rrbracket = \{ G \mid Cons(G, MM) \}\]