Data-Race-Freedom (DRF) Theorem
Recap: C/C++ Concurrency

C11 semantics

C11 compiler transformations

C11 mappings to processors

Observation: Weak accesses allows more transformations
Q: Should we weaken the model as far as possible?

A:

(1) Mapping to other model is efficient.

(2) May not always be beneficial for program transformations.
    (why?)

(3) Programmability is affected.
Can developers get rid of relaxed accesses?

Solution:
Do not use weak accesses.

Cons: compilation is expensive.

Another approach:
Use strong accesses for synchronization.
Use weak accesses otherwise.
Programmability Concerns

No synchronization $\implies$ no communication among threads

Synchronization $\implies$ data race

Strong accesses for synchronization $\implies$ data race on strong accesses

Do programmers have any guarantee?

Program behaviors should be dictated by stronger accesses involved in data race.
Example

Program with lock/unlock.

Global order on lock and unlock operations in an execution.

Sequential consistency/interleaving execution.

\[
\begin{align*}
\text{lock}(m) & \quad \text{lock}(m) \\
X = 1; & \quad a = X; \\
\text{unlock}(m) & \quad \text{unlock}(m)
\end{align*}
\]

(i) \( a = 0, X = 1 \) (ii) \( X = 1, a = 1 \)
C11 Programmability Concerns

\[
\begin{align*}
\text{if}(X) & \quad \parallel \quad \text{if}(Y) \\
Y &= 1; & X &= 1;
\end{align*}
\]

The program is race-free.

C11 allows racy execution $X = Y = 1$.

No guarantee for programmer.
Motivation for the DRF theorem

WMC is complicated:
- Most programmers “do not understand” WMC.
- Leads to subtle bugs $\leadsto$ hard to debug and fix.

Define programming disciplines that:
- Avoid weak behaviors.
- Can be understood without referring to the WMM.

The DRF discipline:
- Do not have any data races.
- Just use locks for synchronization.
Definition (DRF property)

A memory model $M$ satisfies the *DRF property* if for every program that is race-free under SC semantics, its allowed outcomes under $M$ are the same as under SC.

- A programming discipline to avoid weak behavior.
- The premise requires us to establish race-freedom under SC.
- So a defensive programmer does not need to understand WMM.

For specific memory models, one can establish more permissive programming disciplines that ensure the absence of weak behaviors.
What constitutes a race under SC? (declaratively)

Given an execution graph $G$ and a relation $R \subseteq G.E \times G.E$, we say that two events $a, b$ $R$-race in $G$ if the following hold:

- $a \neq b$
- $a.loc = b.loc$
- $\{a.op, b.op\} \cap \{W, U\} \neq \emptyset$
- $\langle a, b \rangle \notin R^+$ and $\langle b, a \rangle \notin R^+$

An execution graph $G$ is called $R$-racy if there are two events that $R$-race in $G$.

$P$ is called racy under SC if there exists an execution graph $G$ such that the following hold:

- $G$ is a $(\mathit{po} \cup \mathit{rf})^+$-prefix of an execution of $P$
- $G$ is SC-consistent
- $G$ is $(\mathit{po} \cup \mathit{rf})$-racy
What constitutes a race under SC? (declaratively)

DRF theorem proof taken from
https://people.mpi-sws.org/~viktor/wmc/
Proof outline (1)

- Let $G$ be a RA-consistent $(\text{po} \cup \text{rf})$-racy execution graph of $P$.
- Let $G'$ be a minimal $(\text{po} \cup \text{rf})$-prefix of $G$ that is $(\text{po} \cup \text{rf})$-racy.
- NB: This prefix might not be unique (e.g., SB).
- Let $a, b$ be two events that $(\text{po} \cup \text{rf})$-race in $G'$.
- Let $x = \text{loc}(a) = \text{loc}(b)$.
- $G'$ is RA-consistent. (why?)
- Let $G'' \triangleq G' \setminus \{a, b\}$.
  - $G''$ is RA-consistent.
  - $G''$ is not $(\text{po} \cup \text{rf})$-racy.
- Therefore, $G''$ is SC-consistent.
Proof outline (2)

Possible cases:

- $\text{typ}(a) = W$ and $\text{typ}(b) = W$
- $\text{typ}(a) \in \{R, \text{RMW}\}$ and $\text{typ}(b) = W$
- $\text{typ}(a) = W$ and $\text{typ}(b) \in \{R, \text{RMW}\}$ (symmetric)
- $\text{typ}(a) = R$ and $\text{typ}(b) = \text{RMW}$
- $\text{typ}(a) = \text{RMW}$ and $\text{typ}(b) = R$ (symmetric)

We cannot have $\text{typ}(a) = \text{RMW}$ and $\text{typ}(b) = \text{RMW}$. (why?)
CASE 1: \( \text{typ}(a) = \bar{W} \) and \( \text{typ}(b) = \bar{W} \)

- \( G' \) is SC-consistent.
  
  (Take an \textit{sc}-order for \( G'' \) and add \( a \) and \( b \) at the end)
CASE 2: \( \text{typ}(a) \in \{R, \text{RMW}\} \) and \( \text{typ}(b) = W \)

- There exists \( a' \sim a \) (\( a \) and \( a' \) are identical except for the read value, and \( a' \) may be a read if \( a \) is an \( \text{RMW} \)) such that some \( G_a \in \text{Add}(G'', a') \) is SC-consistent.
  (read from the last write to \( x \) in the sc-order for \( G'' \))

- Let \( G_{ab} \in \text{Add}(G_a, b) \).

- \( G_{ab} \) is SC-consistent and \( (\text{po} \cup \text{rf}) \)-racy.
CASE 3: typ(a) = R and typ(b) = RMW

- Let $G_b \triangleq G' \setminus \{a\}$.
- $G_b$ is SC-consistent. (why?)
- $b$ is the $(\text{po} \cup \text{rf})^+$-maximal write to $x$ in $G_b$.
- There exists $a' \sim a$ ($a$ and $a'$ are identical except for the read value) such that some $G_{ba} \in \text{Add}(G_b, a')$ is SC-consistent and $\langle b, a' \rangle \not\in G_{ba}.\text{rf}$.  
  (read from the $(\text{po} \cup \text{rf})^+$-maximal write to $x$ in $G''$)
- $G_{ba}$ is $(\text{po} \cup \text{rf})$-racy.