Aakash Saini(2018CSM1001)

March 25, 2019

- **Observations:** Paper proposes a tool "Eraser" for detecting the data-races dynamically within the lock-based multi threaded programs. The idea used by Eraser is - it looks for "unprotected" accesses to shared variables. Eraser protects all the shared variables by lock and monitors all read/writes of variables during execution of program. It uses a lock set algorithm in which each variable v builds a set of locks c(v) that holds candidate locks and c(v) are adjusted each time v is accessed. File and the line number are indicated when a race is reported for all active stack frames. The report includes the thread ID, memory address, type of memory access, and important register values such as the program counter and stack pointer. Eraser ensures that all shared variables are only accessed when a corresponding lock is held. For determining which lock corresponds with a variable, a track of locks is kept by the Eraser during the accessing of variable which in turn reduces the set associativity. Eraser detects the race condition if variable is accessed without any of its corresponding locks. Some annotations are also allowed by the Erases which are helpful when using a non-standard locking mechanism i.e., the mechanisms which are not in PThread library.

- **Limitations:** Eraser may introduce the false alarms and the methods used to detect initialization are inadequate. If the code testing lock set algorithm is not fully exhaustive, potential race conditions may not be discovered. A large number of trials are required to make sure that most of the race conditions are detected. Monitoring overhead may increase the execution time of program by a factor of 10 to 30.

- **Conclusions:** Eraser is a practical and effective way for detection and avoidance of the data races. Eraser is able to detect the data races statically and dynamically and also it performs better than happens-before methods. Erases is able to eliminate the false alarms using Annotations from three broad categories of false alarms i.e., memory reuse, private locks and benign races. Its lock set algorithm outperforms all the previous existing algorithms like happens-before.

- **Future work:** In future work, deadlock detection mechanism can be added for flag locking acquisitions which are incorrectly. More effective mechanisms can be implemented to detect and avoiding the false alarms.