list ceatech

Frama-C Day 2016

# Proof of Concurrent C Code
Via Automatic Code Transformation

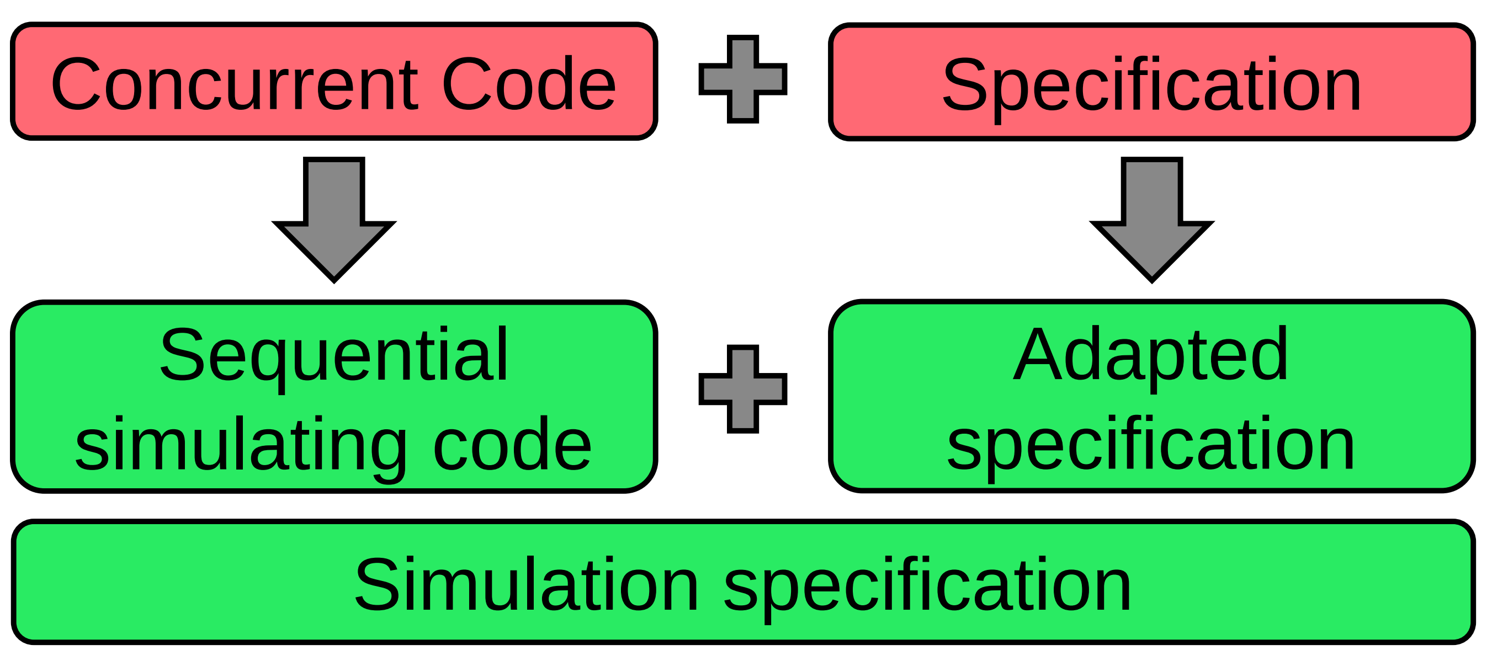
## Motivation and Goals

Frama-C and WP allow deductive proof of sequential C code. To deal with concurrent code, we propose to transform it into sequential code.

**Needs :**
- Automatic proof of generated code
- Automatic code transformation
- Correctness of the transformation

**Hypotheses :**
- Interleaving semantics
- No dynamic thread spawning
- User-specified atomic sections

## Overview of the Code Transformation

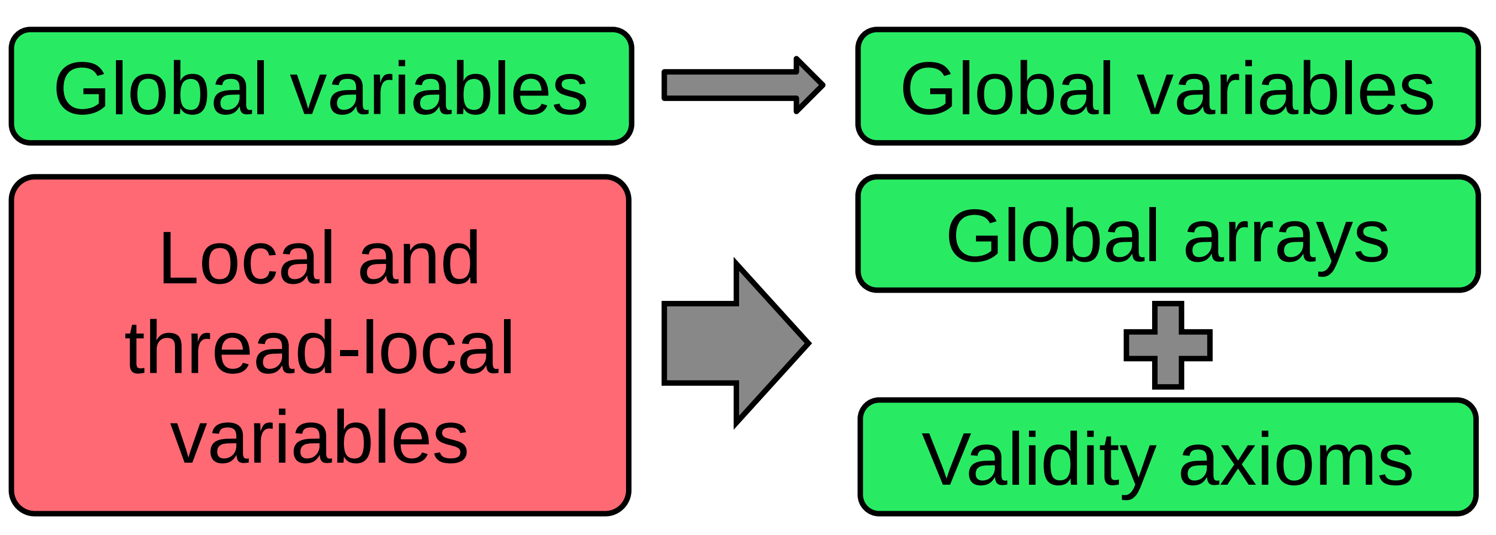


**Global Approach**

We transform the original code into a sequential simulating one. The code we get can be then proved using WP :

- Automatic proof of generated code

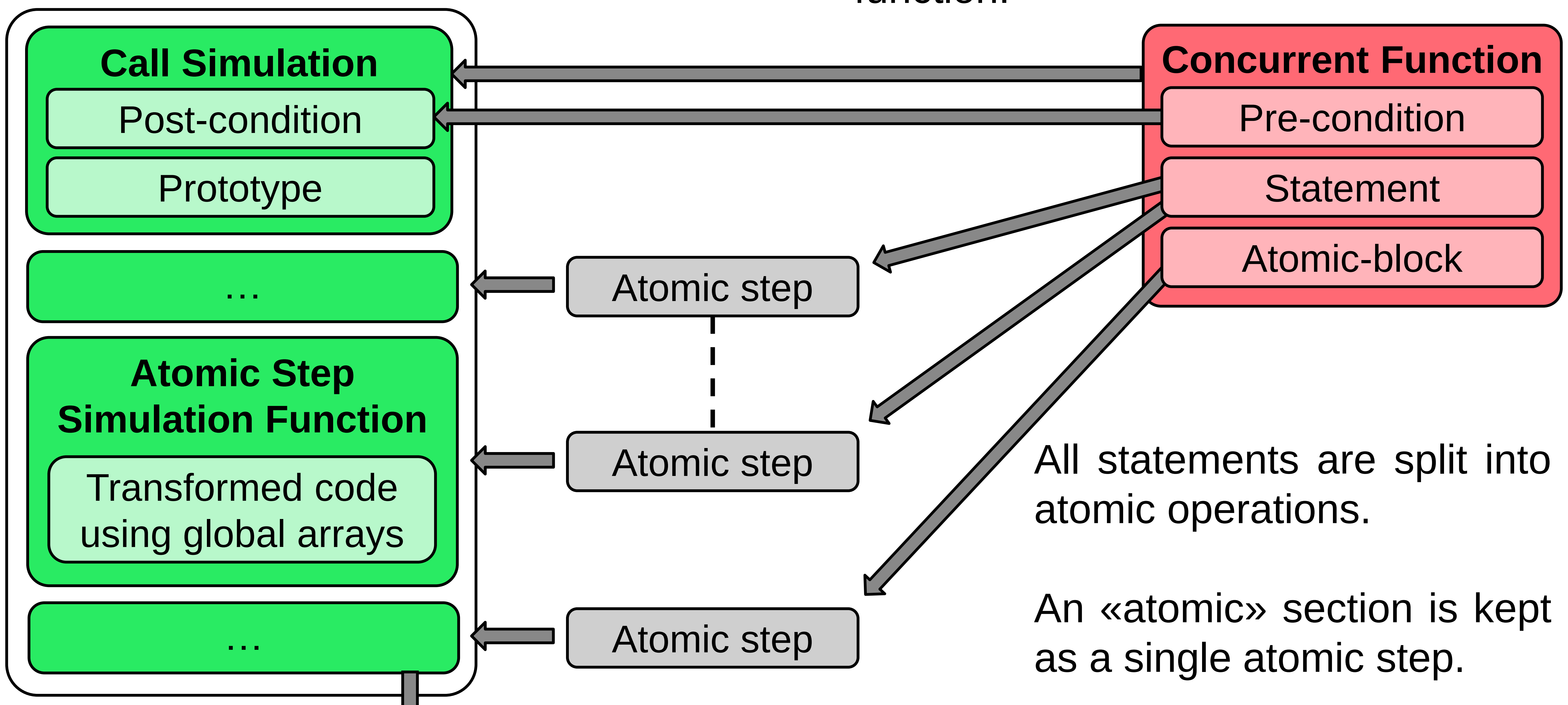
**Variables**



**Functions and statements**

A concurrent function call is modeled by a specified C function prototype.

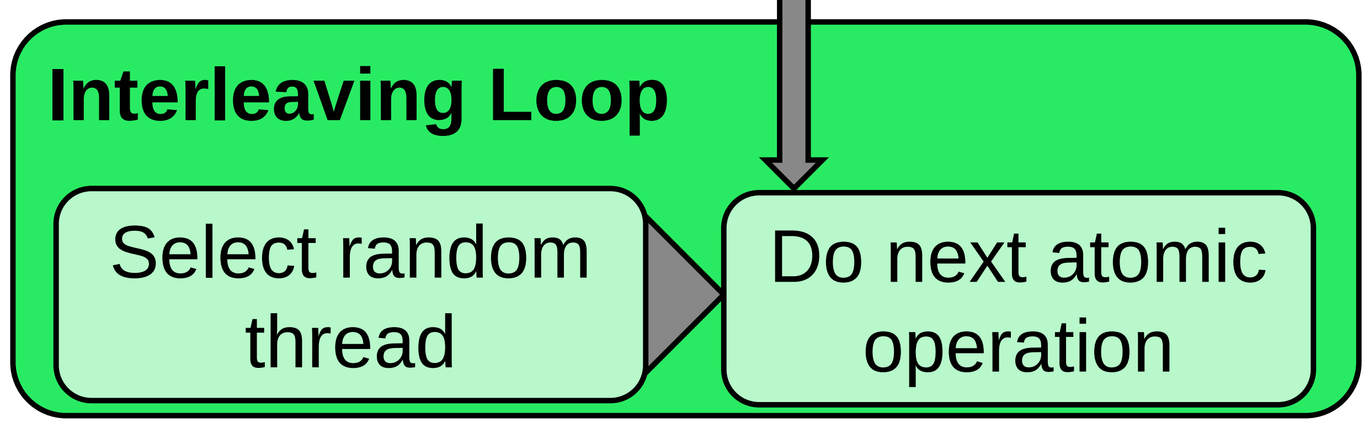
Every atomic step is simulated by a function.



All statements are split into atomic operations.

An «atomic» section is kept as a single atomic step.

**Interleavings**



Atomic operations are interleaved using an infinite loop.

**Specifications**

We state simulation invariants and global invariants as pre/post conditions of simulating functions, and as a loop invariant of the interleaving loop :

- Automatic transformation
  - Code
  - Specification

## Ongoing and Future Work

**Ongoing Work :**
- Correctness of the transformation
  - Code (with Coq Proof Assistant)
  - Specification

**Future Work :**
- Add concurrency primitives to ACSL
- Support weaker semantics

**Reference** : A. Blanchard, N. Kosmatov, M. Lemerre, and F. Loulergue. A Case Study on Formal Verification of the Anaxagoros Hypervisor Paging System with Frama-C. FMICS 2015.

**Supervisors** : Frédéric Loulergue (Université d'Orléans, LIFO)
Nikolai Kosmatov (CEA LIST)

**A.BLANCHARD | Allan.Blanchard@cea.fr**

DGA LIFO Open Source Innovation Spring INSTITUT CARNOT CEA LIST université PARIS-SACLAY