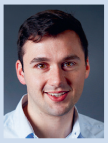
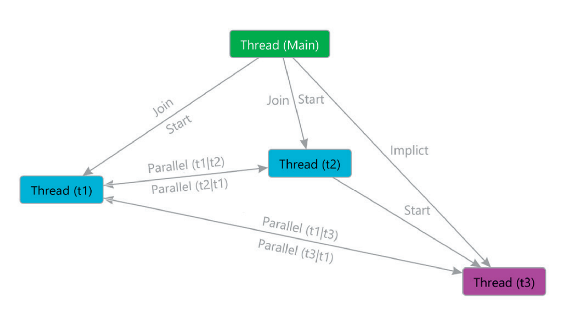


Thomas Sebastian Charrière

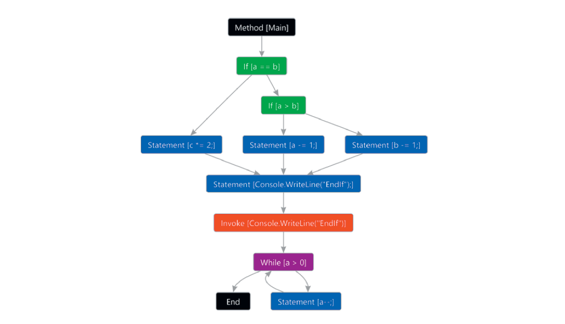
Graduate Candidate	Thomas Sebastian Charrière
Examiner	Prof. Dr. Luc Bläser
Co-Examiner	Dr. Felix Friedrich, ETHZ
Subject Area	Software

Extended Static Race Detection for Visual Studio

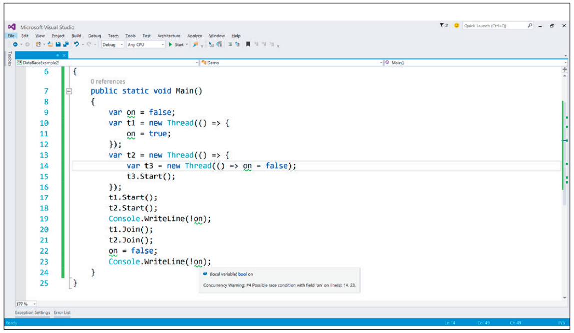
An on-the-fly static race condition checker for Visual Studio



The static data race analysis produces a thread graph representing all threads in a program with their start/join relations.



A control flow graph generated from Roslyn's Syntax API enables dataflow analysis.



```
6   {
        0 references
7       public static void Main()
8       {
9           var on = false;
10          var t1 = new Thread(() => {
11              on = true;
12          });
13          var t2 = new Thread(() => {
14              var t3 = new Thread(() => on = false);
15              t3.Start();
16          });
17          t1.Start();
18          t2.Start();
19          Console.WriteLine(!on);
20          t1.Join();
21          t2.Join();
22          on = false;
23          Console.WriteLine(!on);
24      }
25  }
```

```
(local variable) bool on
Concurrency Warning: #4 Possible race condition with field 'on' on line(s): 14, 23.
```

Potential data races are directly highlighted and marked in the Visual Studio IDE.

Introduction: Multi-threaded programs are known to be prone to concurrency errors, in particular to data races. These issues are hard to find because of their non-deterministic occurrence. Static analysis offers an approach for the systematic detection of such errors, conservatively finding all issues with possible false positives. Highest effectiveness can be achieved through an on-the-fly detection, working directly within the programmer's integrated development environment. Unfortunately, no such tool currently exists for prominent programming languages, such as C#.

Approach/Technologies: This thesis implements a static detection of low-level data races for the C# programming language. This is realised as a Visual Studio plug-in based on Roslyn, the new .NET compiler framework. The tool is designed in abstraction layers: a generic interprocedural dataflow analysis and a data race checker on top. The data race detection employs a novel algorithm, which considers start/join relations among threads.

Result: The result proves to be an efficient static checker that detects potential data races in C# program code within Visual Studio, highlighting and marking such issues during code writing. In contrast to other existing solutions, this checker performs a real static analysis (not only simple local bug pattern location) as a practical Visual Studio plug-in. Because of the generic architecture, the tool could be extended to support deadlock detection in the future as well.