Practical Detection of Concurrency Issues at Coding Time

Luc Bläser

Hochschule für Technik Rapperswil
Switzerland

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

HSR Concurrency Lab
Prof. Dr. Luc Bläser

ISSTA 2018
17 July 2018, Amsterdam

Motivation

Detect concurrency errors in the IDE

- Interactively mark issues during coding
- Primary focus on data races

Requirements

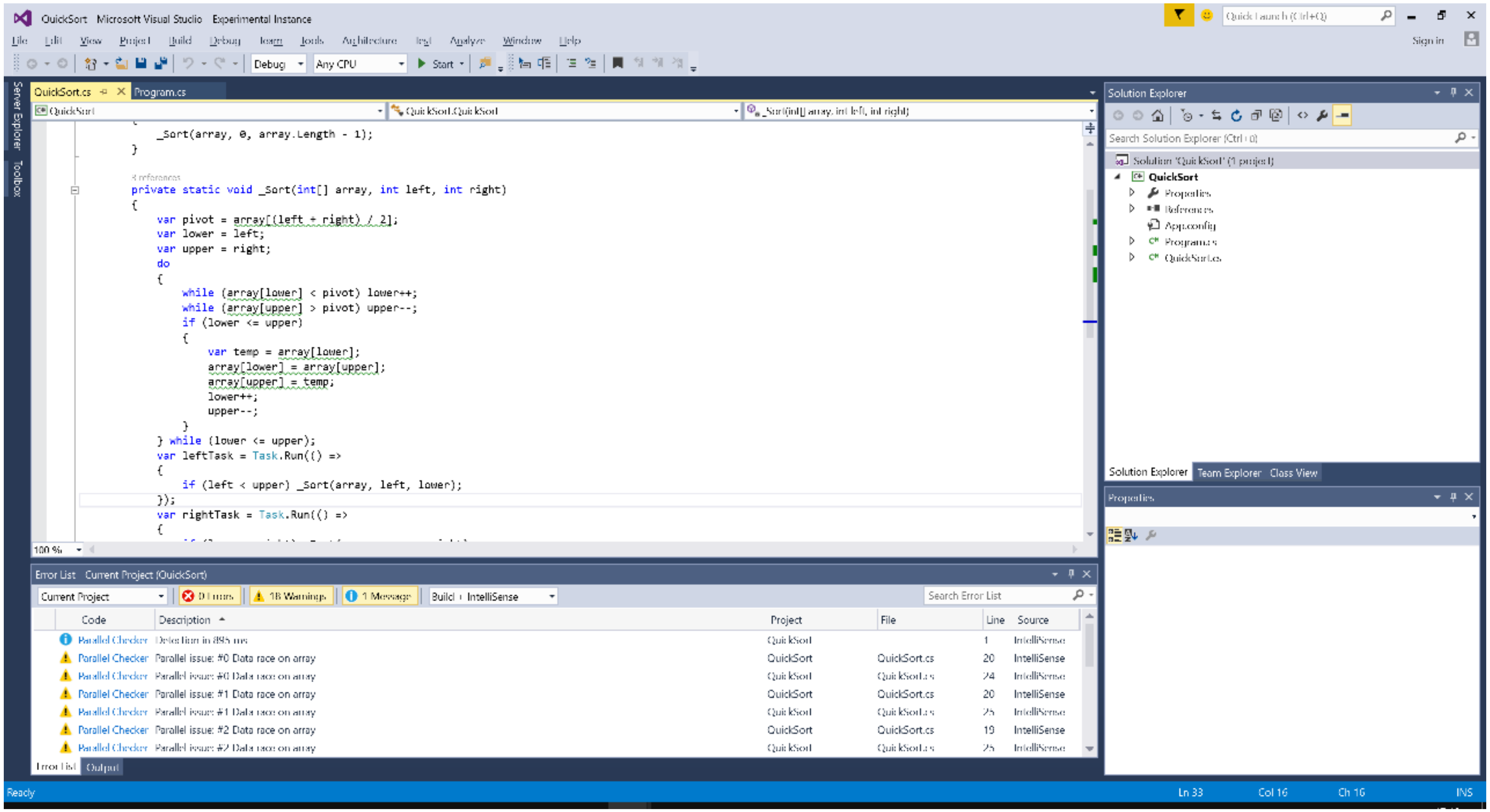
- **Static**: Analyze source code, even if not compileable
- **Fast**: Quick feedbacks within a few seconds
- **Precise**: As few false warnings as possible

Compromise: We may miss issues (incompleteness)

HSR Parallel Checker

- New static checker tool for Visual Studio IDE
- For latest C#, covering wide concurrency spectrum
  - Tasks, async/await, parallel loops, various sync. constructs, atomics, volatile, finalizers, timers, parallel queries …
  - UI-apps/libraries/unit tests/console-apps
- Downloadable on Visual Studio Marketplace (>2.5k installs)

Brief Demo



```
        _Sort(array, 0, array.Length - 1);
    }

    3 references
    private static void _Sort(int[] array, int left, int right)
    {
        var pivot = array[(left + right) / 2];
        var lower = left;
        var upper = right;
        do
        {
            while (array[lower] < pivot) lower++;
            while (array[upper] > pivot) upper--;
            if (lower <= upper)
            {
                var temp = array[lower];
                array[lower] = array[upper];
                array[upper] = temp;
                lower++;
                upper--;
            }
        } while (lower <= upper);
        var leftTask = Task.Run(() =>
        {
            if (left < upper) _Sort(array, left, lower);
        });
        var rightTask = Task.Run(() =>
        {
```

Error List Current Project (QuickSort)

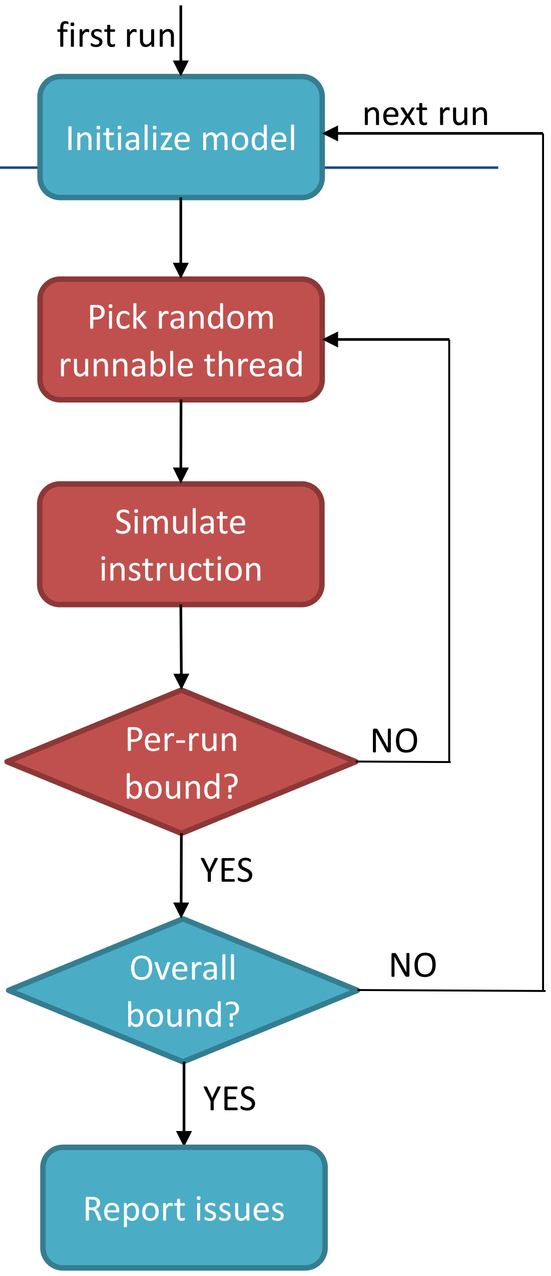
Current Project | 0 Errors | 18 Warnings | 1 Message | Build + IntelliSense

Code	Description	Project	File	Line	Source
Parallel Checker	Detection in 895 ms	QuickSort		1	IntelliSense
Parallel Checker	Parallel issue: #0 Data race on array	QuickSort	QuickSort.cs	20	IntelliSense
Parallel Checker	Parallel issue: #0 Data race on array	QuickSort	QuickSort.cs	24	IntelliSense
Parallel Checker	Parallel issue: #1 Data race on array	QuickSort	QuickSort.cs	20	IntelliSense
Parallel Checker	Parallel issue: #1 Data race on array	QuickSort	QuickSort.cs	25	IntelliSense
Parallel Checker	Parallel issue: #2 Data race on array	QuickSort	QuickSort.cs	19	IntelliSense
Parallel Checker	Parallel issue: #2 Data race on array	QuickSort	QuickSort.cs	25	IntelliSense

Approach

Randomized mostly-concrete interpretation

- Map code to internal runtime model
- Simulate execution on this model
- Maintain exact state where possible
- Repeated random scheduling
- Per-run and overall bound
- Report encountered issues
- Vector clock for race detection



Particular Aspects

- Reproducibility of results
  - Seeded pseudo-random numbers
  - Bounds on logical number of steps and size
- Dynamic technique in a static context
  - Does not run the code
  - Code may be incomplete or incorrect
- Deliberately simple design
  - Random scheduling, no constraint solver
  - Examine more code with less sophistication

Abstract States

- Cope with unknown input
  - Command line args, user/file input etc.
- Uninterpreted value
  - Stands for any possible value
  - Propagates through expressions
- Imprecise assumptions
  - Take random branch on uninterpreted condition
  - Ignore locks, thread starts/joins on uninterpreted object
  - Do not report data races on uninterpreted addresses

May result in false positives (and false negatives)

Experimental Evaluation

- 10 C# GitHub project by user ranking
- 3 C# GitHub projects, «concurrency» tag
- 402 assemblies
- 3.4 MLOC source code

Project	Lines of Code	Assemblies
Roslyn 15.2	1,851,645	114
SignalR 2.2.2	86,574	31
Nancy 2.0.0	72,345	56
ILSpy 2.4	279,432	14
CefSharp 57.0.0	14,116	9
ReactiveUI 7.4.0	33,381	10
MsBuild 15.1.1012	397,281	20
Hangfire 1.6.14	73,986	12
Polly 5.2.0	91,363	6
NLog 4.4.11	63,381	6
Orleans 1.4.2	137,695	29
Akka.NET 1.2.2	225,744	82
Rx.NET 3.1.1	155,358	13
	3,482,302	**402**

Experimental Results

Analyzed assemblies	402
Analysis time	13 min in total
Time per assembly	1.7 sec on average
Detected issues	121 races
False Positives	14 (12%)
Real issues	107
Productive issues	89
Found in	Roslyn, SignalR, NLog, Rx.NET

Found Races

Roslyn

```
private static int s_delayMilliseconds = 0;

static GCManager() {
   System.Threading.Tasks.Task.Run(() => {
      …
      s_delayMilliseconds = (int)key.GetValue(…);
      …
    });
    …
}

internal static void TurnOffLowLatencyMode() {
  // same for UseLowLatencyModeForProcessingUserInput()
  if (s_delayMilliseconds <= 0) …
  …
}
```

data race

SignalR

```
class Client {
  public static void Main() { …
    if (Arguments.IsController) {
       ControllerHub.Start(Arguments);
    }
    Run().Wait();
  }

  static async Task Run() {
    …
    while (TestPhase != ControllerEvents.Connect) {
      …
    }
    …
  }

  static void OnPhaseChanged(ControllerEvents phase) {
    …
    TestPhase = phase;
    …
  }
}

class ControllerHub {
  internal static void Start(…) {
    … ThreadPool.QueueUserWorkItem(_ => Run());
  }

  static void Run() {
    … RunConnect(); …
  }

  static void RunConnect(){
    … SignalPhaseChange(ControllerEvents.Connect); …
  }

  static void SignalPhaseChange(ControllerEvents phase) {
    … Client.OnPhaseChanged(phase);
  }
}
```

data race

and other issues…

Conclusion

- Concurrency checking at development time
  - Directly warn in IDE when races are programmed
  - Requires to be static, fast, and precise
- Full-fledged implementation for C#
  - Broad concurrency feature spectrum
  - It is the sole static race checker for modern C#
- Simple but experimentally effective approach
  - Applicable to other programming languages

Thank You for Your Attention!

- **Contact**
  - Luc Bläser, HSR Hochschule für Technik Rapperswil
  - lblaeser@hsr.ch, http://concurrency.ch
- **Project Website**
  - http://parallel-checker.com
- **VS Marketplace**
  - https://marketplace.visualstudio.com/items?itemName=LBHSR.HSRParallelCheckerforC7VS2017

HSR HOCHSCHULE FÜR TECHNIK RAPPERSWIL
FHO Fachhochschule Ostschweiz

Parallel # Checker

HSR Concurrency Lab
Prof. Dr. Luc Bläser