Safety Analysis for Loop Parallelism in Java

Mihai Tarce, “Politehnica” University of Timisoara, Romania

ADVISORS: Danny Dig, Ralph Johnson

**Motivation**

- The **multi-core** revolution is here!
- **Existing software** is designed for single core, and needs to be refactored
- **Java** provides the ParallelArray class (JSR166), which executes array operations (e.g., map, apply, reduce) in parallel

**Goal**

Determine statically whether loop iterations can be parallelized.

**Fundamental Questions/Challenges**

**Correctness**

- Determine **independence of iterations**
- Determine **uniqueness of array elements**

- Determine that loop operations do not have **conflicting accesses** (read, write, I/O)

**Performance**

- Because the analysis will be used by an interactive tool, it has to take **less than 30 seconds** to run.

**Approach**

Use a static analysis library (WALA) to check array element uniqueness.

- Use a **demand-driven, context-sensitive points-to analysis** to determine whether array elements point to different objects

```
for (int i = 0; i < array.length; i++) {
    Product p = new Product(i);
    ... array[i] = p;
}
```

Unroll loop iterations twice

```
Product p = new Product(j);
... array_j = p;
Product p1 = new Product(k);
... array_k = p;
```

**Research Results**

- The resulting analysis was used to check several **benchmarks** (Java Grande, JOlden) and a medium-sized **project** (NLP - a natural language processing library)

- Analysis runtimes were **under 30 seconds**, making it feasible as an interactive analysis for an Eclipse plug-in

**Related Work/Interaction with Other Projects**

- The T.J. Watson Libraries for Analysis (WALA), a state-of-the-art static analysis library for Java (wala.sourceforge.net)
- M. Marron et al., **Sharing analysis of arrays, collections, and recursive structures**, PASTE’08, pp. 43-49
- M. Sridharan, R. Bodik, **Refinement-based context-sensitive points-to analysis for Java**, PLDI’06, pp. 387-400