Thread-Modular Abstract Interpretation for Multi-Threaded Code

Michael Schwarz
m.schwarz@tum.de

Supervisors: Helmut Seidl & Dirk Beyer

Collaborators: Julian Erhard (ConVeY) & Sarah Tilscher (ConVeY) & Simmo Saan (UTartu) & Vesal Vojdani (UTartu) & Kalmer Apinis (UTartu)

ConVeY

Setting

Multi-threaded software is ubiquitous, a lot of communication happens via global variables. Thread-modular analyses needed to avoid state explosion

Approaches from literature

Miné's style (e.g. [2])

- Propagate values from `unlocks` of a mutex to its `locks`, provided appropriate side-conditions are met

Vojdani's style (e.g. [5, 6])

- Compute set of *protecting mutexes* always held when global is accessed
- Publish value on `unlock` of last protecting mutex

```
main:
   lock(b); g = 0; unlock(b);
   y = create(t1);
   lock(a);
   lock(b);
   x = g;
   ...
```

```
t1:
   lock(a);
   lock(b);
   g = 42;
   unlock(a);
   g = 17;
   unlock(b);
```

- Miné: $x \mapsto \{0, 17, 42\}$
- Vojdani: $x \mapsto \{0, 17\}$

Generally, **incomparable!**

Contributions in Non-Relational Setting

- **Formulation of both styles in a common framework**
- **Comparison**
- **Principled soundness proofs for both styles of analyses**
- **Identify weaknesses and propose improved versions**

Side-Effecting Equation Systems[1]

Accumulate flow-insensitive information for globals during flow-sensitive analysis of locals

$$u \xrightarrow{\texttt{unlock}(b)} v$$

$$[\![u, \texttt{unlock}(b)]\!]^\sharp\ \eta = \textbf{Let}\ \sigma' = \ldots\ \textbf{In}\ \underbrace{(\{[g] \mapsto (\eta\,[u])\,g \mid g \in \ldots\}}_{\text{Side-Effects}}, \underbrace{\sigma')}_{\text{Contribution to } [v]}$$

Ingredients for More Precise Analyses

Consider further finite abstraction to exclude more reads

For each global g

- $W\,g$: Set of locksets held *when* last writing to g
- $P\,g$: Set of locksets held *since* last writing to g

For each mutex a

- $L\,a$: Set of locksets held when last acquiring a
- $V\,a$: Set of globals that must have been written locally since last acquiring a

Combined
(L, V, W, P)

Lock-Centered (L, V) — Write-Centered (W, P)

Miné — Protection-Based (Vojdani-style)

Experimental Evaluation

Analyses implemented within the static analysis framework for multi-threaded C programs Goblint[6].

Benchmarks: 13 not-too-small multi-threaded Posix programs

Runtime: Increases with sophistication

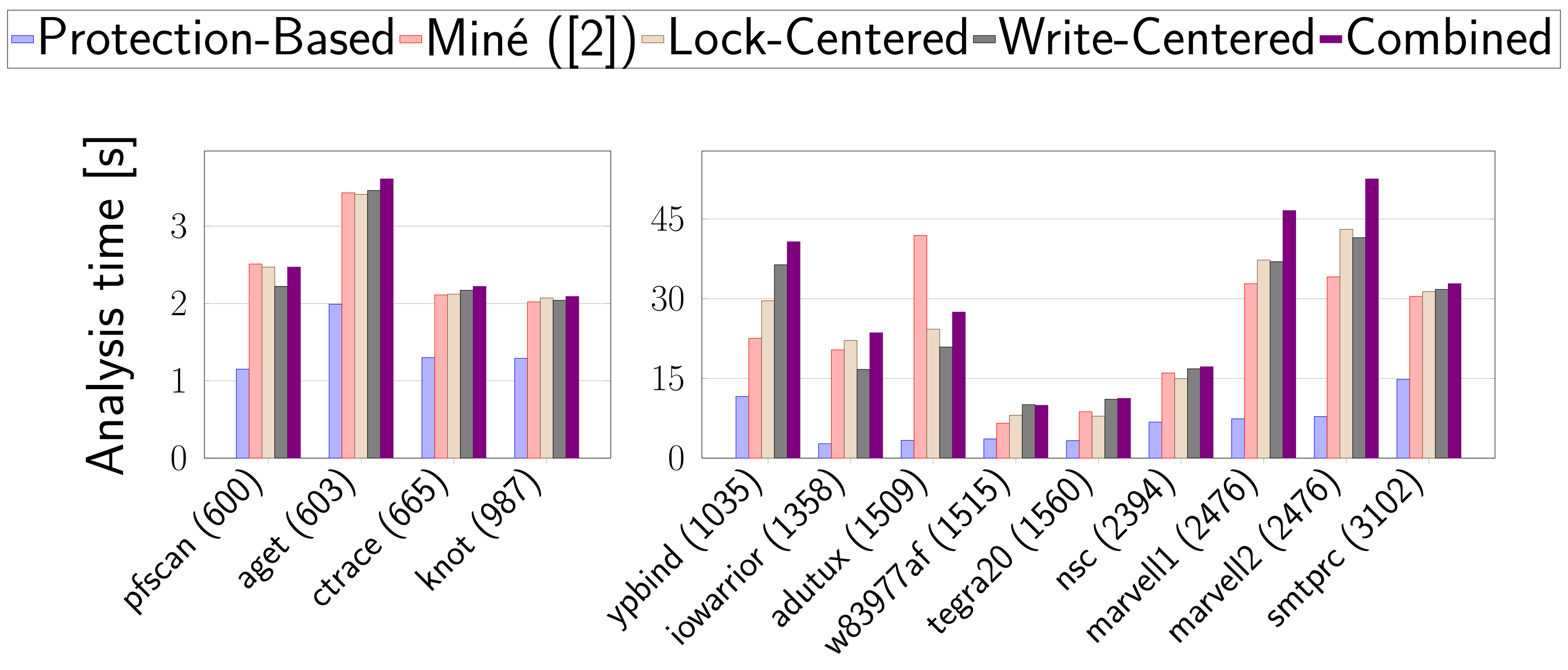

Figure 1: Analysis times per benchmark program (logical LoC in parentheses).

Precision: (as measured by abstract values of globals read)

- Equally precise for 11/13 benchmarks
- For `pfscan` and `ypbind`: [2] less precise for 6% resp. 16% of globals

Experimental Conclusions

Protection-Based Analysis sufficiently precise at low cost.

Clustered Relational Analysis with Local Traces [4]

- Relations between globals are likely to be mediated by mutexes
- Relational analyses inspired by Protection-Based Analysis
- Framework for precision improvement by seamlessly incorporating further finite abstractions (e.g. thread *ids*)
- Tracking bigger clusters may be both more and less precise ?!
- For certain domains (e.g. Octagons), tracking subclusters of size ≤ 2 already yields maximum precision

References

[1] Kalmer Apinis, Helmut Seidl, and Vesal Vojdani. Side-effecting constraint systems: a swiss army knife for program analysis. In *APLAS '12*, pages 157–172. Springer, 2012.

[2] Antoine Miné. Static analysis of run-time errors in embedded real-time parallel C programs. *Logical Methods in Computer Science*, 8(1):1–63, mar 2012.

[3] Michael Schwarz, Simmo Saan, Helmut Seidl, Kalmer Apinis, Julian Erhard, and Vesal Vojdani. Improving thread-modular abstract interpretation. In *SAS '21*. Springer, 2021.

[4] Michael Schwarz, Simmo Saan, Helmut Seidl, Julian Erhard, and Vesal Vojdani. Clustered relational thread-modular abstract interpretation with local traces. In *ESOP '23*. Springer, 2023.

[5] Vesal Vojdani. *Static Data Race Analysis of Heap-Manipulating C Programs*. PhD thesis, University of Tartu., December 2010.

[6] Vesal Vojdani, Kalmer Apinis, Vootele Rõtov, Helmut Seidl, Varmo Vene, and Ralf Vogler. Static Race Detection for Device Drivers: The Goblint Approach. In *ASE '16*, pages 391–402. ACM, 2016.