WCET and Mixed-criticality: How to Quantify the Confidence in WCET Estimations?

Sebastian Altmeyer, Björn Lisper, Claire Maiza, Jan Reineke,
and Christine Rochange

2015-06-07

Aim

Starting point: Dagstuhl seminar on mixed-criticality

Issue: how confident can be we that a WCET estimate is safe?

Position paper: no hard results, bringing up issues, food for discussion

Mixed Criticality

Mixed criticality: to let tasks with different safety levels coexist in the same system

Dream of industry: **consolidation**, to replace many smaller processors, and their wiring, with a few powerful processors. Cost reduction

Challenge: guarantee that highly safety-critical tasks satisfy safety standards despite sharing resources with low-criticality tasks

Includes timing guarantees (as well as other requirements)

Vestal’s Model

How ensure that timing requirements for high criticality tasks are met, and still have high resource utilisation?

Vestal’s model (RTSS 2007): define WCET bounds for tasks with different *confidence*:

C(LO): WCET bound with low confidence

C(HI): WCET bound with high confidence

(C(LO) $\leq$ C(HI). Easy to generalise to more than 2 levels of criticality)

Idea: Overruns of low-criticality tasks less harmful. Thus, for scheduling: use C(LO) for such tasks, and C(HI) for high-criticality tasks

Will yield good resource utilisation while ensuring, with high confidence, that timing requirements for high-criticality tasks are met

Confidence in WCET bounds

How estimate confidence in bounds C(LO), C(HI)?

Hard problem! Depends on how these bounds are obtained, but in what way?

Something for our research community to consider

We discuss sources of uncertainty, without attempting to quantify, for:

- static analysis
- measurement-based, and hybrid analysis

(Probabilistic approaches: future work)

Confidence in Static WCET Analysis

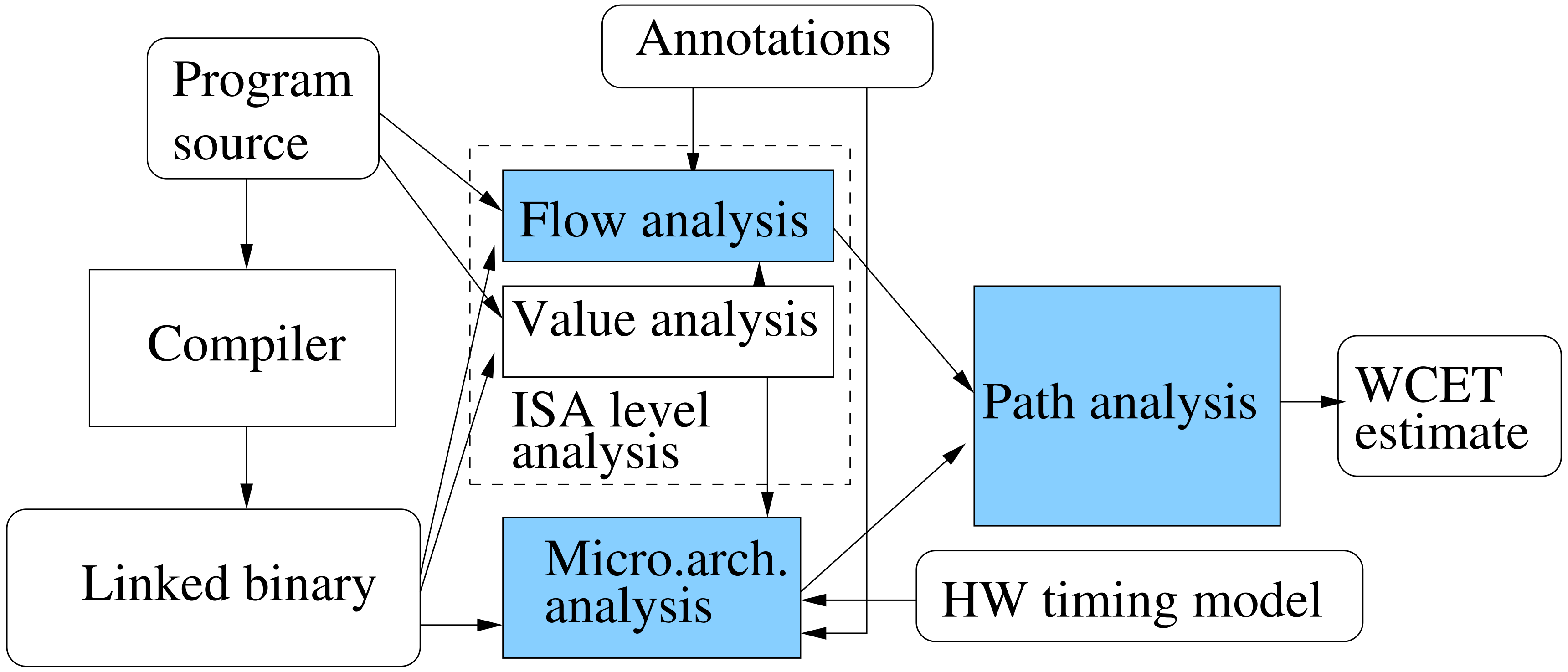
Static WCET analysis is often stated to be **safe** (no underestimation possible, i.e., confidence 1)

But this is true only **within the mathematical model** where the analysis is formulated

True for the **real system** only to **the extent that the system fulfils the assumptions in the model**

(Indeed the case for **all** mathematical modelling!)

Structure of Static WCET Analysis



Uncertainties in ISA-level Analysis

Due to user annotations

Due to analysis method

Due to (lack of) traceability of information from source to binary level

Uncertainties Due to User Annotations

Obvious source of uncertainty

Comes in different formats:

- Flow fact annotations
- Annotations specifying properties of the environment (volatiles, ranges for input variables, . . .)

Can yield unsafe WCET estimates if set the wrong way

Automatic flow fact analyses can help, but hard to eliminate the need completely (Halting Problem)

Uncertainties Due to Analysis Method

The method may be unsound

Examples:

- Not modeling arithmethic wraparounds properly
- Not using target processor’s floating-point arithmetics
- Wrong assumption about endianness
- Assumptions on pointers

Uncertainties Due to Missing Source-Binary Traceability

Sound & precise WCET analysis must be done on the linked binary

Hard! Much information is lost

Thus tempting to do some analyses on source level, and map to binary level

But compiler optimisations can change things like program flow, rendering mapped flow facts unsound

There is research how to trace flow facts through compiler optimisations, but not implemented in production compilers

Microarchitectural Analysis

Uncertainties in the timing models

Uncertainties in the analysis

Uncertainties in the Timing Models

Hard to get architectural details from the HW manufacturer (missing or incorrect documentation)

Human errors translating architectural descriptions into formal models

Possible alleviations:

- Reverse-engineering HW parameters (like cache replacement policy) from observations by running micro-benchmarks
- (Semi)automatic extraction of timing models from formal HW descriptions (like VHDL)

These methods can reduce uncertainties, but not eliminate them completely

Uncertainties in the Analysis

Microarchitectural analysis: estimate possible HW states to obtain local WCET bounds for small program fragments (basic blocks)

Can be very many such states . . .

Abstraction often necessary. Done for caches, but not for pipelines so far

Tempting to use only local worst cases (# of states can be reduced), but can be unsound if timing anomalies are present

Also tempting to treat subsystems separately (like analysing cache and pipeline separately), but the architecture may not be timing-compositional

Some research projects aim at timing-predictable architectures: for them sound and efficient analyses are likely to exist, but COTS HW is not designed according to these principles

Path Analysis

Combining flow facts from ISA-level analysis with local WCET bounds to produce a global WCET bound

Most often IPET is used, applying some ILP solver

Main source of doubt: the WCET bound is usually expressed in # of machine cycles, and can thus be numerically large: will the solver (using finite number representation) then produce a correct solution?

Confidence in Tool Implementations

Analysis tools are complex pieces of software

Even if the analysis is sound, the tool implementation can be wrong

This is an additional source of uncertainty

Ongoing research to alleviate this problem: formal verification of the implementation using proof assistants

Goal: reduce the *trusted code base* that is not verified. Smaller trusted code base $\implies$ increased confidence in the correctness of the tool

Obviously a huge job. . .

Done for a simplified WCET analysis tool implemented in the CompCert environment, demonstrates that it can be done in principle

Confidence in Measurement-based WCET Analysis

End-to-end measurement-based: run a lot of test cases, high-water-mark, add safety margin

Fundamental problem: measured times always underestimate the WCET unless one happens to hit the right combination of input and initial HW state

Quality of test data becomes crucial for the condfidence in the result

Uncertain how to estimate this quality. Some coverage measures seem to be relevant, but they don’t give guarantees that the worst case is found

Hybrid Methods

Use measurements to estimate local WCETs for program fragments

Then combine with flow facts in a traditional path analysis

Two ways to estimate local WCETs:

- Direct instrumentation
- From end-to-end measurements using model identification

Confidence for both rely on quality of test data

Instrumentation can give probe effects

Model identification can suffer from incomplete information due to poor test data

Interference through Shared Resources (Single Core)

WCET analysis assumes that tasks run in isolation

Rarely true in practice

On single-core: interference through preemption (modified hardware state)

Modeled by an additional delay, like CRPD for cache effects

Often estimated by static analysis methods: then suffers from similar uncertainties as static WCET analysis

Interference through Shared Resources (Multi-core)

Interference through shared resources (buses, shared memories, caches, . . .)

A precise estimate of the interference requires very good knowledge of the accesses to shared resources + a complex analysis on top of that – a source of uncertainty

Requires that delays due to conflicts can be upper bounded. Depends on HW design, like for the bus arbiter

Delays sometimes hard to estimate due to poorly documented HW features of COTS multi-cores

Ongoing research on time-predictable multi-cores: such architectures will allow higher confidence in WCET bounds

But commercial multi-cores are typically not time-predictable

Discussion, Conclusions

We discussed possible sources of errors for some classes of WCET analysis methods, and their influence on the confidence in WCET bounds

Motivation: need to quantify confidence in WCET estimates for mixed-criticality systems. like in Vestal's model

Sources of (reduced) confidence:

- Methods relying on mathematical models: how well the models comply with reality
- Methods relying on measurements: quality of test data

Main issue: **HOW QUANTIFY CONFIDENCE??**