Justifying the Service Provided to Low-Criticality Tasks in an Avionic Mixed-Criticality Multi-Core System

Iain Bate

Dept of Computer Science

University of York

iain.bate@york.ac.uk

Overview of the Seminar

- **My interest in Mixed-Criticality Scheduling (MCS)**
- **Some real world requirements for MCS**
  - Existing work on deploying MCS
  - The challenge of guaranteeing service for lower-criticality services
  - Limitations of observations
  - Based on S. Law, I. Bate, B. Lesage, Justifying the Service Provided to Low Criticality Tasks in a Mixed Criticality System, RTNS, 2020
- **Additional challenges from multi-core**
- **Planned work in the near future**
- **Open research questions**

My Interest in MCS

- **It brings together a number of areas of work**
  - Multi-objective optimisation
  - Search-based testing
  - Safety arguments
  - Evidence based on static and dynamic analysis backed up by statistics
  - Real world applicability
  - Research challenges inspired by the real world

Some Real Requirements for MCS

- **WCET processes are pessimistic, but we would struggle to prove this to a certification authority**
  - Can we better use this ‘spare’ utilisation?

- **Mixed Criticality Scheduling allows low criticality tasks to execute on the same target hardware as high criticality tasks**
  - Allowing low criticality tasks to have deadlines, periods and timing requirements
  - Giving a good balance between safety, flexibility and maximising utilisation

Some Real Requirements for MCS

- **Low DAL tasks are developed / tested to the same standard as a high-DAL task!!**
- **Saving is gathering less evidence of integrity**
  - Remember writing code is relatively cheap
  - The code may even be autocoded
  - Partitioning must be employed as certification is often based on segregation and isolation
- **It is very important we address exactly what we mean by a 'low-DAL' task**
  - What tasks/operations are appropriate/safe as 'low-DAL' tasks?

Some Real Requirements for MCS

- **Additionally we have a number of tasks we would consider to be high criticality**

- **We can afford for them to be disabled for short periods of time**
  - For instance recording error logs in non-volatile memory, a time consuming but still important process

- **Principal benefits – Cost & Flexibility**

Some Real Requirements for MCS

- **In particular we studied the application of a monitoring task responsible for writing to Non-Volatile Memory**
  - Robust and low-DAL
  - Responsible for writing data from a queue to NVM
  - Able to drop some jobs, then need to run normally to catch up
  - <u>Can we be confident its write queue will not overflow?</u>

Some Real Requirements for MCS

- **AFDX (time-triggered aircraft comms) is a similar example**
- **Aspects of AFDX have tight timing requirements as comms schedule is slot based**
  - Transactional-style requirements
  - Reading and writing to the device have very tight deadlines
  - Gathering data and putting it into packets takes more time
  - Short periods of not putting data into packets could be okay
- **A buffer overflow could have consequences**
  - The system should be designed accordingly though
  - A key part of safety is understand components failure modes

Existing Work on Deploying MCS

- **Two models have been considered**
  - AMC+
  - Robust scheduling
  - A. Burns, R. I. Davis, S. Baruah, I. Bate, Robust Mixed-Criticality Systems, IEEE Transactions on Computers, Vol. 67, No. 10, pp. 1478-1491, 2018.
- **Robust scheduling intended to give**
  - Greater control over what happens when a task exceeds C_{Lo}
  - To improve degradation of services

Existing Work on Deploying MCS

- **In robust scheduling**
  - *Normal* mode
    - F hi-criticality tasks can exceed C_{LO}
  - The system then moves into *resilient* mode
    - Up to M tasks can exceed C_{LO}
    - Each robust task can skip up to S jobs
  - Then, the system enters *high-criticality* mode
    - Low-criticality tasks are not released
  - On an idle tick the counters for jobs skipped (JF) are reset
  - If C_{Hi} is exceeded, then there is a power cycle

Existing Work on Deploying MCS

- **Current static schedulability analysis confirms**
  - High-criticality tasks always meet their deadlines
  - Low-criticality tasks meet their deadlines when jobs are released and completed
  - If jobs are allowed to be skipped, then the number is bounded

Existing Work on Deploying MCS

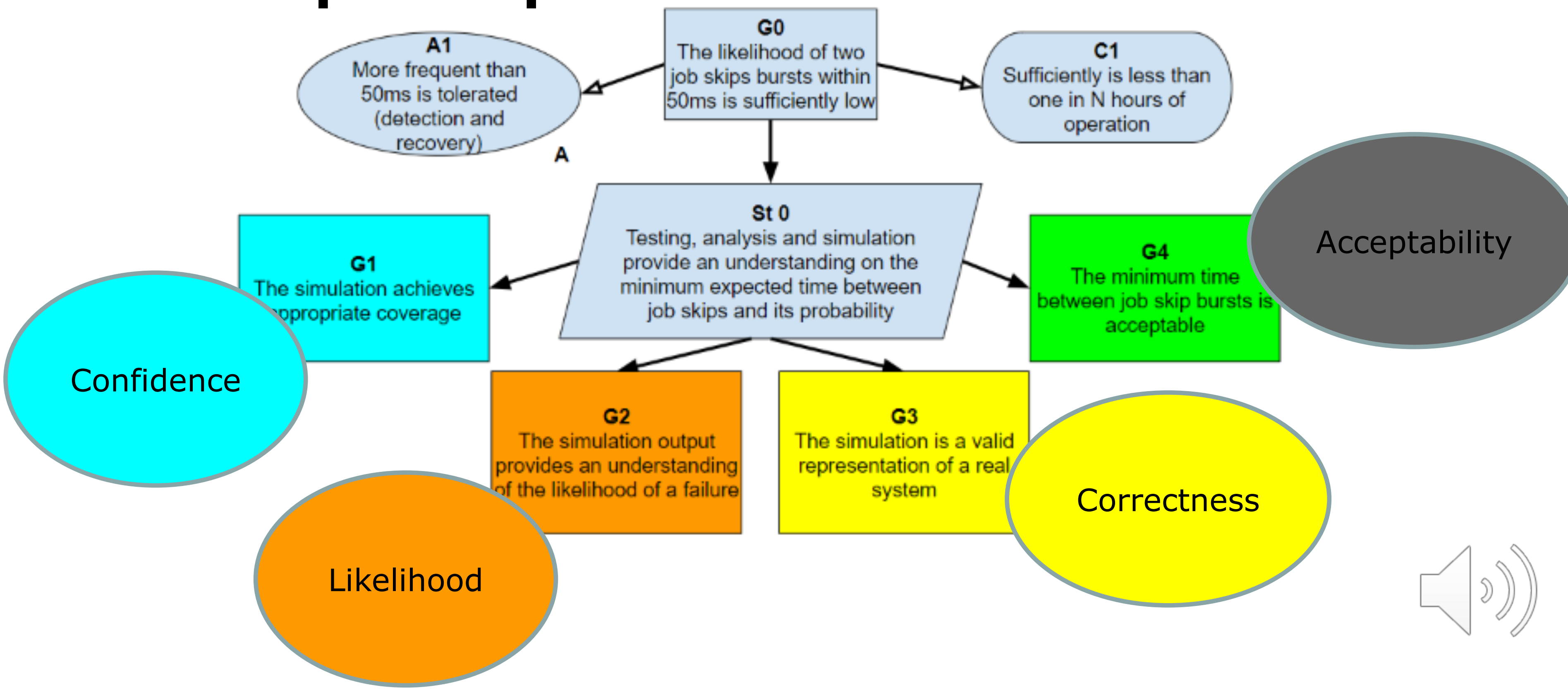
- **We looked at the deployment of both AMC+ and robust scheduling to give**
  - Equivalent partitioning and segregation to current operational systems
  - Quantified the overheads and included in schedulability analysis
  - Clustering tasks to reduce the overheads
  - AMC+ assessment is covered in S. Law, I. Bate, B. Lesage, Industrial Application of a Partitioning Scheduler to Support Mixed Criticality Systems, EUROMICRO Conference on Real-Time Systems, 2019.
  - Robust scheduling in Steve Law's thesis

Existing Work on Deploying MCS

- **Many academic papers have looked at improving low DAL service**
- **None (to our knowledge) have identified ways to quantify it**
- **We want to know**
  - What is the minimum gap between entering high-criticality mode?
  - The max jobs skip allow us to guarantee from a normal level the buffers don't overflow
  - The minimum gap then allows us to guarantee the buffers return to their *normal* level

Assessing Low Criticality Service

- **A GSN supported statistical approach built around a scheduler simulator, seeded with real data, and updated throughout the software development process**



What About Low-Criticality Services?

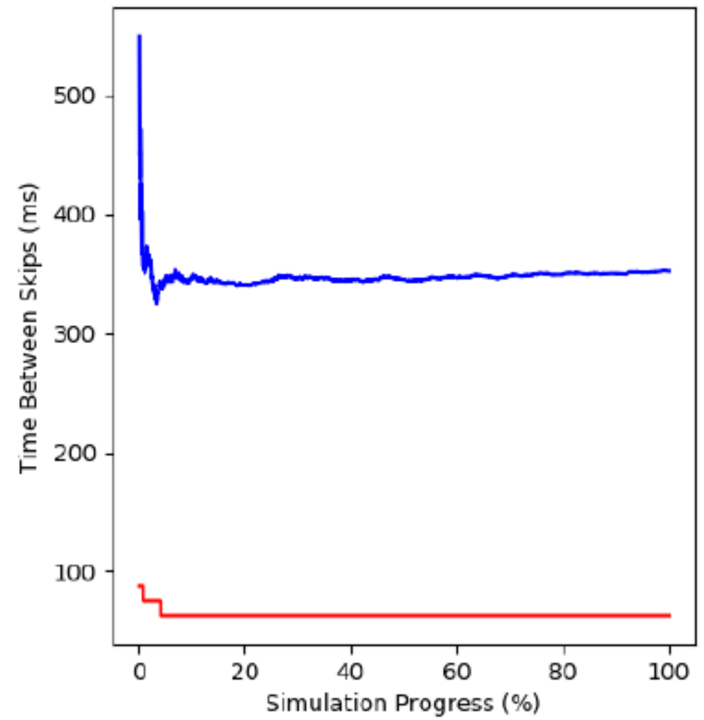
Confidence

- **Hard to get enough data off a real fully-integrated system**
- **Established a three-part simulator**
  - Used the actual high-level set of tasks and associated attributes (e.g. period, deadlines and priorities)
  - Used low-level timings based on extensive search-based execution times
  - S. Law, I. Bate, Achieving Appropriate Test Coverage for Reliable Measurement-Based Timing Analysis, EUROMICRO Conference on Real-Time Systems, 2016
  - Realistic overhead model based on actual RTOS and timings
- **40% low DAL utilisation added into the system**
- **AMC+ assessed first**

What About Low-Criticality Services?

Confidence

- **How can we have confidence that the simulator has observed a large enough sample of the search space?**
- **How can we have confidence that continued testing will not reveal new results?**
- **Clearly average and minimum give limited confidence**



Blue line is the average

Red line is the minimum

What About Low-Criticality Services?

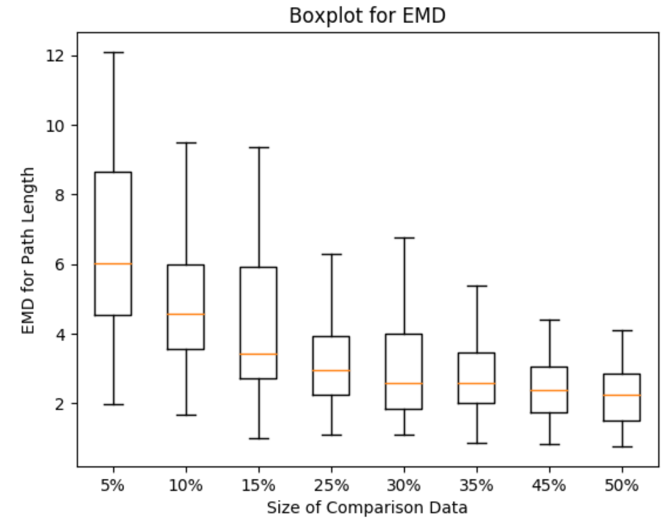
Confidence

- **Convergence was assessed**
  - Take X% of the simulation results and compare to the rest
- **Use confidence intervals**
  - How confident are we that the minimum gap is greater than Y%?
- **and chi-squared test**
  - Does the first X% come from the same distribution as the rest?
- **and Earth Movers Distance**
  - How different are the distributions?
- **Substantial evidence from whole system testing**
  - Null hypothesis is not refuted
  - *The system will not exceed the required minimum inter-skip rate*

What About Low-Criticality Services?

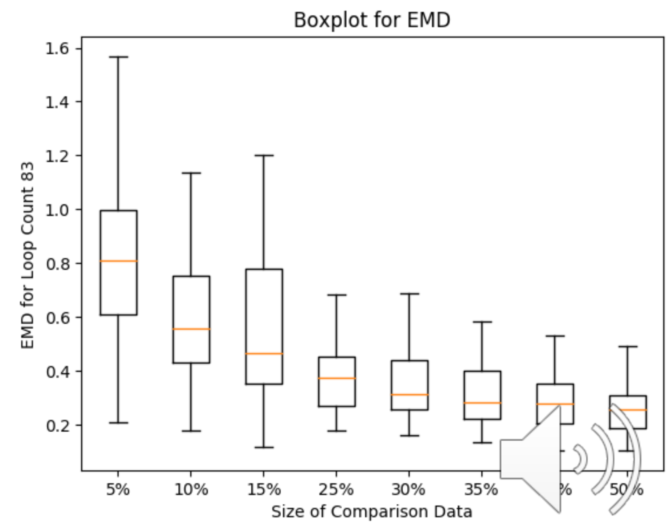
Confidence

- **Our work looked at whether the distribution was changing as #simulations increased**
  - I Bate, D Griffin, B Lesage, Establishing Confidence and Understanding Uncertainty in Real-Time Systems, RTNS, 2020.



- **Distributions of execution times gave some confidence**

- **Distribution of significant factors added to this, e.g.**
  - Path length, loop counts, #iPoints



What About Low-Criticality Services?

Likelihood

- **How often do we come close to seeing an error?**
- **If an error has been observed, what is the frequency of occurrence?**
- **If an error has *not* been observed, use a fitted distribution to assess exceedance probability**
  - Noting the usual health warnings here

Time Between Skips	% Results More Frequent Than Time Between Skips
50ms	99.9948%
60ms	99.9947%
70ms	99.76%
80ms	98.73%
90ms	96.76%
100ms	75.43%

What About Low-Criticality Services?

Correctness

- **How can we be sure the simulation is correct?**
  - Simulation offers a route to fast, iterative, repeatable testing… provided the simulation is correct
- **Mostly by construction**
  - The data underpinning it is right
  - My PhD student claims he is a good software developer ☺

What About Low-Criticality Services?

Acceptability

- **Are the results acceptable?**
- **40% additional utilisation could be added to the process**
- **Task could be expected to complete its operation, without error, in 99.995% of cases**
  - But… that's potentially 360 low DAL timing errors per hour…
  - If less than 40% utilisation was added, then it is likely there would be a substantial reduction in timing errors

What About Low-Criticality Services?

Acceptability

- **Is this good enough…?**
  - Depends on the task’s system requirements
  - If not, the system can be refined, with the simulation easily repeated
  - Hopefully issues understood early point in the design lifecycle

What About Low-Criticality Services?

Acceptability

- **With robust scheduling, there were no timing errors even with the 40% utilisation load**
- **Robust requirements for NVM were**
  - The task is capable of writing data to flash memory at a faster rate than the reporting tasks can write data to the shared memory buffer
  - The buffer means up to four jobs can be skipped – S=4
  - After a job skip burst, the task must execute the following four jobs for at least C_{LO} to ensure no data is lost

Limitations of the Work

- **Skewedness or incompleteness of the timing data**
- **Work was based on a simple but real platform**
- **Argument and evidence falls short of a proof**
- **Argument and evidence may be sufficient**
  - As Low As Reasonably Practicable (ALARP) is accepted
  - The amount of integration testing that should be performed
  - Critical systems should have fault tolerance based around expected failure modes

Additional Challenges from Multicore

- **Confidence in execution time will be diminished with greater variability**
- **Simulation will be more complex as tasks are not independent**
- **Simulation time needed for equivalent confidence vastly increased**
  - General increase in the number of operational scenarios
  - i.e. permutations of task sets executions
- **Small changes may have a bigger wider effect**

Planned Work

- **Hi-Class is a large project with most of UK civil avionics**
- **Key driver is multi-core for avionics**
  - Low numbers of predictable cores
  - Bare metal
  - 653 and non-653 based RTOSs
- **UoY providing advice on**
  - What information is needed from multi-core timing analysis
  - Testing strategy to gain this information
  - Architectural options for multi-core

Planned Work

- **UoY is mainly investigating task allocation and scheduling of multi-core systems**
- **Based on algorithm to generate realistic task sets**
- **Plans to create a simulator for multi-core tasks with the following interference characteristics**
  - No Dependency
  - Additive
  - Super Additive
  - Hidden
- **Nuanced extended robust scheduling policy to deliver more controlled graceful degradation**

Planned Work

- **MOCHA is a Huawei funded project**
- **Much more complex software and platforms**
- **Scheduling policies for DAGs**
- **Digital Twin to support Design Space Exploration (DSE)**
- **DSE includes**
  - Designing memory architectures
  - Allocating tasks to cores
  - Controlling back pressure
  - Reducing RTOS overheads

Acknowledgements

- Members of the RTS group
- Rolls-Royce for years of support, inspiration and access to real systems
- Innovate UK for funding the Hi-Class project
- Innovate UK for funding the ATICS project
- Huawei for funding the MOCHA project
- Members of WashU for stimulating discussions and space to think

Justifying the Service Provided to Low-Criticality Tasks in an Avionic Mixed-Criticality Multi-Core System?

Iain Bate

Dept of Computer Science

University of York

iain.bate@york.ac.uk

Overview of the Talk

- **Lower criticality tasks could be added to the system**
  - Lower criticality doesn’t mean soft real-time
  - Tasks would be implemented to the same level
- **Timing data is available for tasks based on search-based WCET analysis**
- **Timing properties of tasks and RTOS well understood**
- **Bounded loss of service is okay for some tasks**

Overview of the Talk

- **Simulator developed to allow loss of service to be understood**
- **Results demonstrate how the impact of more functionality being added to the system**
- **Results partly empirical which challenges conventional industrial and RTS thinking**
- **Route to certification identified**
- **Multi-core is going to make the challenges harder**
- **Multi-core is going to increase the need to understand the loss of service**

Open Research Questions

As we move away from jobs always being completed periodically and completing within their deadlines

- **What are the real timing requirements?**
- **How to write functions differently for robust tasks?**
- **How do we form representative timing profiles of tasks?**
- **Where do C_{Lo} and C_{Hi} come from?**

Open Research Questions

- **How can the management of time be integrated within Model-Based Engineering?**
  - E.g. embed loss of service into Simulink models
- **How do we know when we have enough data about a system?**
- **How to understand the potential impact of uncertainties?**
- **How to create a CAST-32A argument for multi-core mixed-criticality scheduling?**

Open Research Questions

Digital Twins (DT) is a well-established practice but what are the challenges around timing

- ***Acceptability* – What information can we realistically be expected to extract from a real system?**
- ***Accuracy* - What does it mean for a simulator to be accurate?**
  - Very much depends on the questions to be answered with DT
- **Efficiency – What is the right level of abstraction for the model and the right type of feedback?**

Many of the research questions are socio-technical