

Prosa: A Case for Readable Mechanized Schedulability Analysis

Felipe Cerqueira, Felix Stutz, Björn B. Brandenburg



Max Planck Institute for Software Systems





Open-source foundation for formally **proven schedulability analysis**



Mechanized proofs provide an opportunity to avoid the correctness pitfalls in real-time scheduling.

Mechanized proofs provide an opportunity to avoid the correctness pitfalls in real-time scheduling. By focusing on <u>readability</u> and by <u>maintaining the</u> <u>established proof culture</u>, mechanized proofs <u>can reach the community at large</u>.

Mechanized proofs provide an opportunity to avoid the correctness pitfalls in real-time scheduling. By focusing on **readability** and by **maintaining the** established proof culture, mechanized proofs can reach the community at large. Thanks to mature proof assistants and libraries,

Prosa allows mechanizing recent and complex schedulability analyses in reasonable time.

Outline of the Talk

Why mechanized proofs?

Challenges & Principles

A Taste of Prosa

Outline of the Talk

Why mechanized proofs?

Challenges & Principles

A Taste of Prosa

What do we mean by mechanized?

RTS theory has been built with pen-and-paper proofs



What is **mechanized** schedulability analysis?



What is the difference?





What is **mechanized** schedulability analysis?



Why mechanized proofs?

Why mechanized proofs?



RTS have become more complex



Source: G. Buttazzo (Keynote @ RTSS'14)

RTS have become more complex



Source: G. Buttazzo (Keynote @ RTSS'14)

This complexity comes with a price

Analysing Real-Time Communications: Controller Area Network (CAN)

K. W. Tindelt^{*}, H. Hansson Department of Computer Systems, University of Uppsala, Sweden

Abstract

The increasing use of communication networks in time critical applications presents engineers with fundamental problems with the determination of response times of communicating distributed processes. Although there has been some work on the analysis of communication protocols, most of this is for idealized networks. Experience with single processor scheduling analysis has shown that models which abstract among from implementation details are at best very prasiminiti and at word lind at watercheduling desender chedulation. In this paper, we derive idealized schedulation is the target of the two the analysis can be applied.

1. Introduction

One of the fundamental difficulties in engineering hard eal-time systems is the development of analysis to bound behaviour of the system. Much work in recent een developing this analysis for a ran-tim patching algorithm known as fland priority pre-emptiing. This work has recently addressed the ing of messages on shared broadcast buses [6], and far token-passing and 'priority pre-emptive ions about the idea our of the interface between the host processor and cations adapter for these buses tion may not meet these and cent research has examined a particular bu mentations from a num her of diffe acturers. This paper reports on this analysis, an we how small diffe sces in the implementation of an can have dramatic effe ance of messages

The real-time has we existing in this paper is called Controller Area Network (CAN) [13]. In particular we examine in detail two-interface chips the 12217 controller from hard, and the 12C200 controller from Philips. We show how the last controller has a very much better worst-case timing performance than the Philips controller. A. J. Wellings Department of Computer Science, University of York, England

CAN is a hereadcast bus designed to operate at speech of up to 1 Mbiblee. Data is transmitted in messager containing between 0 and 8 bytes of data. As 11 bit number is associated with each messages. The identifier is required to be enique, is the sense that two simultaneously active messages originating from different sources must have distinct identifiers (typically, as identifier is corresponds to a particular type of message from a specific source). The identifiers sources two purposes: (1) assigning messages. A station filters messages from so the messages, and mark registers. Thus CAN messages have no explicit destination, since any mation with an approprise filter may resolve a message.

Like Ethernet, CAN is a collision-deter to contention. The identifier field of a CAN moused to control access to the bus after colli advantage of certain electrical characteristics of a CA3 buy if multiple stations are transmitting concurrently an sits a 'V', then all st bus will see a '0'. Conversely, only if all stations to 17 will all moursaurs more situring the bus see a 'T'. I effect, the CAN bus acts like a large AND-gate, with each station able to use the output of the gate. This behaviour is used to resolve collisions: each station waits until bus ids (as with Ethernet). When silence is detected, each station begins to transmit the highest priority mes output queue whilst monitoring the bus the first part of the message to be transmitted; the identifie and from mo If a station transmits a rece hus and sees a dominant bit ('0'), then it stops trans since it knows that the message it is to highest priority message in the system. B are deemed unique within the system, a station tra the last bit of the identifier must be transmitting the highest priority queued message and hence can start tra d hence can start transmitting the body of the message. The CAN message format contains 47 hits of protocol control information (the identifier, CRC

⁶The work was supported in particy de U.K. (BFREC, part worker ORENOCS). ¹Department of Computer Systems, P.O. Bes, 125, 5-751 (SI Typeda, Sweller (E-mail: Annihilocts.cov.net) 1003-6725556 (Soc.00 © 1096) (EEEE 2009). The original analysis for CAN had a bug that remained undetected from 1994 to 2006 [1].

[1] Davis, R. I., Burns, A., Bril, R. J., & Lukkien, J. J. "Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised." Real-Time Systems, 35(3), 239-272, 2007.

Bugs are no longer an exception

Proofs have become so complicated that they often contain bugs.



Analysis for safety-critical systems?



How to ensure that schedulability analysis is actually correct?

examine in deal two interface shape the RENT controller. He has he of the Devolve weboar devolves a collocate	Average & Marting Star Section of		and a to purply softed to to be see	A Disease of the second s
from head, and the EXC200 controller from Philips. We must be transmitting the highest priority qurant message, show how the last controller has a very much better and heart message de high of the message.		And a set of the second s	sense Telescondy for advance from the last	And I ARR AND A REAL PROPERTY AND A REAL PROPE
worst-case timing performance than the Philips controller. The CAN memage format contains 47 bits of protocol	the second	an attack " " " " even before a far "n fore " " " a star ten for in the star of the set	administration of the second s	the second s
ported information (the identifier, CBC data,	With the second se	Another and a series and a series state of the series and a series of the series of th	and because of some second sec	the second se
The soft was approved in party do U.K. (PERC, part number 0800005 "Department of Compare Fasters, PC dos 125, 571 (07 Equation, Senter C and Ameridians, us., and	and a state of the second	and the second the William and a second second second second second	174.078	proved as her some to survey addressed - a state of the to the to the solution of the solution
	and an and spectra to the function density for the second se	ten te dar. 1 and 1. 1	a perception administra per dia far administra da dar	the second s
	The first state of the state of	Andre and the second second and the second s	at he mayb, mapp a sense on stands	of the loss descention of the loss of the
1/62-4735/4-804.00-0-1894-0828	And a state of the	And the first second se	these as an anomality in. When conducting new sector	the spatial is induced in the local of the local in the second seco
	A PART AND ADDRESS OF A DECEMBER AND ADDRESS A	at a peak start of the	provide and the second of the process of the second of the	and the state of t
	The second	and the arrive segment place is seen block when	serving any other for comparation true for some radi	contraction in contract in the state of the
	Real a size is large section in the section in the section is a section of the section of the section in the section is a section of the sect	and a set of the sequences in the set of the composition for a large	or the sector rate dealine if a real or is provide.	a strend process for a configuration of the strend process of the statement of the strend with a strend process
	and the second sec			and append to a data mount of the second to have also be added as and all a second to be added as a second to be added as
	The state of the s	the second		a claim suprovid, hought has an annu and annu the claim of the line of the state of the
the second se	THE ADDRESS OF THE AD	to be the send whether and the sengrating wheth	the life is the property is not describe product.	process of the local lines and proof is an experimental of the lines where the lines we have been been been been been been been be
make further up that a share of a	states and a first state while the state of	and and a set of the s	a second to be a second finance, when per-	a constant and and a second bigself a range - weather a reference at 60 basising control of the second second
	Training of stress for a stress for a stress of the stress			and the second states and the second se
and a second	The second secon			and a bolice bank that an one was an an and an instant of the part of the second
and the second s	in anyone sees in it from Line, here the set of the set	the second descent and	CPS	a of how, is all sound particular in their . And he will shall be the stating on the same of how and the how on
	All which has not an end of the set of the s			a provide a se formant, treast, the party of the second se
	Real and a real real real and real real real real real real real real			the strange for burger, the provide the plant, we are affining single for singless that burgers and the second secon
man data gails. Notice process tolering	a design from the state of the		Machanitan Inc. Inc. 199	a find, and there are in the first the plane has been as a start of the start of th
start the desired by a second s	the local sector is a sector by the sector is the sector i		1. T	in the second
the local line where the second line was a second line was a second line where the second li	Annual Sciences and an Annual Science of a second science of the S			and a second sec
second district and process artifactory	and any and another and the second the second to the second of the second to the secon	to a second se		second and the second s
	become become prove strategy or an in the law becomes the strategy means that	- C.C.		and a second and a second period of the second
	a set of the set of th	Landa and a landa		and the second se
	the second secon	Included of the same tacking and a 10 K.	design the improvement of printed or	CALLER CALL LAND AND AND AND AND AND AND AND AND AND
	artista of the second s	Journal of Systems Automotion		and and have been been and the second second second the second se
When you have been a state of the second sec	and a familiar a familiar and a state of the set of the	and because any descent and because	and the same state of the same state of the	the second
	International Antiparty and Control of Contr			
special data for a set that we wanted	while same I have not a margine root of the party with the same of the same of the same is said to save		second diverse of spacing the loss with the	
	Service and the service of the servi	⁴ University for design space of multiprocessor nucleositypics property whEPCP: A Sup-	achevaluation Francesco Supervise a restant	The second
The Rest Control of Security Security and a security in a strength and again in the security of the security o	has been all and the second se	for collecting systems 14.50 C	The Revenue of the second seco	and the second s
and a started start of which is an of the started started started and	and the property of the second s	Multi-L	AND VETURE N IN THE WARD AND ADDRESS OF ADDR	the second based of the second
	the distance of the second sec	a construction of the second s	(Non-York' Reporter by his sector of the se	AND AND AND ADDRESS AND ADDRESS ADDRES
Improved Blocking Time Analysis and Evaluation for the	the schementing, where the properties delays and spins. We have been been been been been been been be			has a to use of some control distance data control of writes of a prior 10,200,200 page. The economic and we contribution and
Multipromage Printing Colling Protocol	- partners taken for to provid privat spectra to any fur the further to the second state and	here a second		
	the second design of the secon		sector and the sector of the s	the second second prove here a second s
The last 3 R. South of a characteristic	the first present of the principal and the second	The part of the part of the part of the part of the second s		The second
	the second	the second se	Contract of the second se	strate to the state of the stat
	instantial to all the second provide real and the second of the second sec	and the second se		and several se
	when both of a court and a board of a scatter a reaction over a courtering in the day of the distance in the			a is the interface as a shift where
	a respect of the second second second second second second second to be an expected to be an expected to be an		And the local division of the local division	Name is an inclusion of the company of the
Restart Reported A (RD) actual highly RDA	the private is the second to be a second to be the second the second sec		and should be a set of the second sec	the set of
	and work on a set of the set of t		and an end of the second secon	the second
	of the same of the	The second	and have a second	
	the plant work	The second	The second secon	The second
	the second state of the se	a contract of the second		
	the Property of the second sec	a to see all to be designed of say while to be assessed on party of a setting the	the state of the second st	
	start when the 1 ran and an and an and an and a start of the start of			
Reports which dealing addresses along hims and have a state or an exercise	second billing and the second dependence of the second			and a second secon
	and a set of the second s	a contract of any lock work of any state of the state of the state and the second	the last of the	
1 Manufacture service and	have a second \$1.1.1.2. Here a second a second second method have been been been been been been been be	the second second as have been been being the shared. Here have being being have to be an address being bein		
hand the proof shalling and thing and a speed has no on a solar hing and	had not all the property of th	I The second sec	And and the set of the local division of the	
	and the second s	A		
	the interpretation of the restored of the second state of the seco	the state is an in the same a second state of the second state of		
	strend long all rate of the strength of the st			
	that they would be in work in the first strategy of a sharehold by any sector at the provided			
	and the second			
	structure and first sharing a first second structure of some pair work the descences of MAT and in 1987	the splitters of Assessments as proved, and prices are splittered on the local of the second present splittere and		
	provided vieways, hences, a speer is in the second and women provide your party and that are been related and the			
	strange, and defines address has been prepared a price of the strange of the stra	the second		
	there are the first the second test and tes	The second s		
	The second		a infla-	

Analysis for safety-critical systems?



Opportunity: correctness is **inherently guaranteed**.



Analyses sometimes need refining

Basic Analysis

In most analyses, practical details are assumed to be negligible.

Analyses sometimes need refining



But when deploying actual systems, we might need to **refine the analysis**.

Analyses sometimes need refining



We call these extensions (i.e., same results + tweaks) neighboring proofs.

But when deploying actual systems, we might need to **refine the analysis**.

Example: incorporating release jitter

	Basic RTA	RTA with Jitter	
Uniprocessor	$R_i \leftarrow e_i + \sum_{\tau_j \in hp_i} \left\lceil \frac{R_i}{T_j} \right\rceil e_j$	$r_i \leftarrow e_i + \sum_{\tau_j \in hp_i} \left\lceil \frac{r_i + J_j}{T_j} \right\rceil e_j$ $R_i = J_i + e_i + r_i$	

It has been known for more than 20 years how to incorporate release jitter into uniprocessor RTA [3].

[3] Audsley, N., Burns A., Richardson, M., Tindell, K., and Wellings, A. "Applying new scheduling theory to static priority pre-emptive scheduling," Software Engineering Journal, vol. 8, no. 5, pp. 284-292, 1993.

Example: incorporating release jitter

	Basic RTA	RTA with Jitter
Uniprocessor	$R_i \leftarrow e_i + \sum_{\tau_j \in hp_i} \left\lceil \frac{R_i}{T_j} \right\rceil e_j$	$r_i \leftarrow e_i + \sum_{\tau_j \in hp_i} \left[\frac{r_i + J_j}{T_j} \right] e_j$ $R_i = J_i + e_i + r_i$
Multiprocessor	$R_i \leftarrow e_i + \frac{1}{m} \cdot \sum_{\tau_j \in hp_i} \left\lfloor \frac{I_j(R_i)}{T_j} \right\rfloor r_j$???

But this result has not been proven for multiprocessor RTA.

Can we do the same for multiprocessors?

	Basic RTA	RTA with Jitter
Uniprocessor	$R_i \leftarrow e_i + \sum_{\tau_j \in hp_i} \left\lceil \frac{R_i}{T_j} \right\rceil e_j$	$r_i \leftarrow e_i + \sum_{\tau_j \in hp_i} \left\lceil \frac{r_i + J_j}{T_j} \right\rceil e_j$ $R_i = J_i + e_i + r_i$
Multiprocessor	$R_i \leftarrow e_i + \frac{1}{m} \cdot \sum_{\tau_j \in hp_i} \left\lfloor \frac{I_j(R_i)}{T_j} \right\rfloor r_j$???

Just sum up the max jitter?

The answer is that we don't know

Different system models have **different** assumptions. What if changing the model **breaks some existing proof?**

Recent case: self-suspending tasks

Misuse of release jitter in the analysis caused bugs in 12 papers related to self-suspensions!

Excerpt from [1]:

Incorrect quantification of jitter for dynamic self-suspending task systems, which was used in [3,4,37,58]. This misconception was unfortunately adopted in [12, 14, 28, 36, 40, 73, 74, 76] to analyze the worst-case response time for partitioned multiprocessor real-time locking protocols.

[1] J.-J. Chen, G. Nelissen, W.-H. Huang, M. Yang, B. Brandenburg, K. Bletsas, C. Liu, P. Richard, F. Ridouard, N. Audsley, R. Rajkumar, and D. de Niz, "Many suspensions, many problems: A review of self-suspending tasks in real-time systems," Department of Computer Science, TU Dortmund, Tech. Rep. 854, 2016 32

How to derive safe extensions?

How to derive safe extensions?



We just need to refine the analysis and let the proof assistant recheck the proofs.

If there is a bug, **it will always be detected. we know exactly what to fix.**



Sometimes we have to combine different analyses

Even if each analysis is **individually correct**, they should not be combined if assumptions mismatch.


How to avoid mismatching assumptions?

How to avoid mismatching assumptions?



We just need to avoid stating contradictory assumptions.

But this can also be mechanically verified!

No more correctness pitfalls





Outline of the Talk

Why mechanized proofs?

Challenges & Principles

A Taste of Prosa

Outline of the Talk

Why mechanized proofs?

Challenges & Principles

A Taste of Prosa

"Formal specifications are complex and full of symbols."

"Formal specifications are complex and full of symbols."

"It might take <u>many decades</u> to verify all we know about real-time scheduling."

"Formal specifications are <u>complex</u> and <u>full of symbols.</u>"

"It might take <u>many decades</u> to verify all we know about real-time scheduling."

> "Knowledge about formal methods tends to be restricted to <u>few research groups.</u>"

"Formal specifications are complex and full of symbols."

"It might take <u>many decades</u> to verify all we know about real-time scheduling."

> "Knowledge about formal methods tends to be restricted to few research groups."

But there's an opportunity to change...

Principles & Goals of Prosa

1. Readability is crucial

2. Some proofs are more important than others

3. We should maintain the proof culture

4. Community involvement

Principle 1: Readability is crucial

The specification should be accessible to researchers with no previous experience with formal methods.

Principle 1: Readability is crucial

The specification should be accessible to researchers with no previous experience with formal methods.

We favor:

Many lemmas, short proofs (few dozen lines) Long, verbose names and few cryptic symbols Heavy use of documentation

Complex notation harms readability

Duration Calculus [Yuhua and Chaochen, 1994]

Furthermore, if there exists a ready task which is not running, then no processor should be idle. So let,

$$SCH_{m} \widehat{=} \begin{pmatrix} \neg \diamondsuit \begin{pmatrix} true; Run(S) \\ \land (\bigvee_{i \in S} \bigvee_{j \in \alpha - S} (Urgt(j, i) ; \llbracket p_{j}.rdy \rrbracket \land Run(S))) \\ \land \Box (Run(S) \land \llbracket \bigvee_{i \in \alpha - S} p_{i}.rdy \rrbracket \Rightarrow \sharp S = m) \end{pmatrix}$$

Complex notation harms readability

Duration Calculus [Yuhua and Chaochen, 1994]

Furthermore, if there exists a ready task which is not running, then no processor should be idle. So let,

$$SCH_{m} \widehat{=} \begin{pmatrix} \neg \diamondsuit \begin{pmatrix} true; Run(S) \\ \land (\bigvee_{i \in S} \bigvee_{j \in \alpha - S} (Urgt(j, i) ; [p_{j}.rdy] \land Run(S))) \\ \land \Box (Run(S) \land [\bigvee_{i \in \alpha - S} p_{i}.rdy] \Rightarrow \sharp S = m) \end{pmatrix}$$

Prosa

```
(* A scheduler is work-conserving iff all processors
  are busy (non-idle) whenever a job is backlogged. *)
Definition work_conserving :=
  ∀ j ∀ t,
  backlogged job_cost sched j t →
  ∀ cpu, ∃ j_other,
    scheduled_on sched j_other cpu t.
```

```
Definition work_conserving :=
∀ j ∀ t,
backlogged job_cost sched j t →
∀ cpu, ∃ j_other,
scheduled_on sched j_other cpu t.
```

```
Definition work_conserving :=
∀ j ∀ t,
backlogged job_cost sched j t →
∀ cpu, ∃ j_other,
scheduled_on sched j_other cpu t.
```

A scheduler is work-conserving iff...

Definition work_conserving :=
 ∀ j ∀ t,
 backlogged job cost sched j t →
 ∀ cpu, ∃ j_other,
 scheduled_on sched j_other cpu t.
 A scheduler is work-conserving iff...
 ...for every job j and time t...
 ...if job j is backlogged at time t, ...

Definition work conserving := ∀ j ∀ t, backlogged job cost sched j t \rightarrow ∀ cpu, ∃ j other, scheduled on sched j other cpu t. A scheduler is work-conserving iff... ...for every job j and time t... ... if job j is backlogged at time t,then every processor cpu has a job *j_other*...

```
Definition work conserving :=
  ∀j∀t,
    backlogged job cost sched j t \rightarrow
    \forall cpu, \exists j other,
      scheduled on sched j other cpu t.
A scheduler is work-conserving iff...
   ... for every job j and time t...
       ... if job j is backlogged at time t, ...
           ...then every processor cpu has a job j_{-}other...
                   ...that is scheduled on cpu at time t.
```

Principle 2: Some proofs are more important than others

To make progress, we should focus on practical results.

Principle 2: Some proofs are more important than others

To make progress, we should focus on practical results.

We should formalize **recent analyses** and move towards **multiprocessor scheduling**.

Critical results should be proven first. E.g., proving **analysis safety** is more important than termination, time complexity or optimality.

Principle 3: Maintain the proof culture

To ensure accessibility, we should reuse the established proof style of the real-time systems community.

Principle 3: Maintain the proof culture

To ensure accessibility, we should reuse the established proof style of the real-time systems community.

We avoid **complex logics** (e.g., temporal operators) and **advanced constructs** from the proof assistant (e.g., records, canonical structures, etc.).

We favor instead first-order logic, lists, functions, basic arithmetic.

Unusual notation discourages adoption

EDF Optimality in PPTL [Zhang, 2014]

Lemma 6. If P_i overflows at $t = kT_i$, there is no idle time unit in $[(k - 1)T_i, kT_i]$.

That is,

$$Sch \supset \left(\bigcirc^{kT_i} (ac_i < C_i) \to \bigcirc^t \left(\bigvee_{j=1}^m r_j = 1 \right) \right) \quad (k-1)T_i < t < kT_i.$$

Prosa — Definition of Instantaneous Service
Definition service_at (t: time) :=
 \sum_(cpu < num_cpus | scheduled_on j cpu t) 1.</pre>

Instantaneous Service

Definition service_at (t: time) :=
 \sum_(cpu < num_cpus | scheduled_on j cpu t) 1.</pre>

Instantaneous Service

Definition service_at (t: time) :=
 \sum_(cpu < num_cpus | scheduled_on j cpu t) 1.
 Sum over each processor...</pre>

Instantaneous Service

Definition service_at (t: time) :=
 \sum_(cpu < num_cpus | scheduled_on j cpu t) 1.
 Sum over each processor...
 ...where job j is scheduled...</pre>

Instantaneous Service

Definition service_at (t: time) :=
 \sum_(cpu < num_cpus | scheduled_on j cpu t) 1.</pre>

Sum over each processor...

...where job j is scheduled...

...of 1 (i.e., a count).

Principle 4: Community involvement

Vision: shared repository of real-time scheduling concepts and proofs.

Principle 4: Community involvement

Vision: shared repository of real-time scheduling concepts and proofs.

We encourage participation by the community:



Mechanized proofs can reach the community at large

1. Readability is crucial

2. Some proofs are more important than others

3. We should maintain the proof culture

4. Community involvement

By focusing on <u>readability</u> and by <u>maintaining the</u> <u>established proof culture</u>, mechanized proofs <u>can reach the community at large</u>.

Outline of the Talk

Why mechanized proofs?

Challenges & Principles

A Taste of Prosa

Outline of the Talk

Why mechanized proofs?

Challenges & Principles

A Taste of Prosa

Prosa is a collection of definitions, assumptions and theorems

Definitions Assumptions Theorems
Prosa covers many concepts from real-time scheduling

Definitions

Library schedule: instantaneous service, cumulative service, job is pending, job is complete...

Library interference: total interference, per-task interference...

Library platform: work conservation, priority enforcement...

<u>Assumptions</u> Theorems

Definitions

Assumptions

Hypothesis H_completed_jobs_dont_execute: completed_jobs_dont_execute job_cost sched.

Hypothesis H_enforces_FP_policy: enforces_FP_policy job_cost job_task sched higher_priority.

Hypothesis H_work_conserving: work_conserving job_cost sched.

Hypothesis H_sequential_jobs: sequential_jobs sched.

[...]

<u>Theorems</u>

Definitions

Assumptions

Hypothesis H_completed_jobs_dont_execute: completed_jobs_dont_execute job_cost sched.



Definitions

Assumptions

Hypothesis H_completed_jobs_dont_execute: completed_jobs_dont_execute job_cost sched.

In any given schedule and for any given actual job execution costs, ...



Definitions

Assumptions

Hypothesis H_completed_jobs_dont_execute: completed_jobs_dont_execute job_cost sched.

In any given schedule and for any given actual job execution costs, ...

... jobs do not execute after completion.

Theorems

Definitions

Assumptions

Hypothesis H_completed_jobs_dont_execute: completed_jobs_dont_execute job_cost sched.

```
Definition completed_jobs_dont_execute :=
   ∀ j ∀ t,
   service sched j t ≤ job_cost j.
```



Definitions

Assumptions

Hypothesis H_completed_jobs_dont_execute: completed_jobs_dont_execute job_cost sched.

```
Definition completed_jobs_dont_execute :=

\forall j \forall t,

service sched j t \leq job_cost j.

For every job j at any time t,
```

Theorems

Definitions

Assumptions

Hypothesis H_completed_jobs_dont_execute: completed_jobs_dont_execute job_cost sched.

Definition completed_jobs_dont_execute :=
 ∀ j ∀ t,
 service sched j t ≤ job cost j.

For every job j at any time t,

...the service received by j is no larger than its cost. Theorems

Theorems are proven in small steps using **lemmas**

Definitions	
Assumptions	
Theorems	

Theorem workload_bounded_by_W :
 workload_of tsk t1 (t1 + delta) ≤ workload_bound.

Theorems are proven in small steps using **lemmas**

Definitions	
Assumptions	
heorems	



Theorems are proven in small steps using **lemmas**

Definitions Assumptions Theorems

Lemma workload_bound_many_periods_in_between : job_arrival j_lst - job_arrival j_fst t ≥ num_mid_jobs.+1 × (task_period tsk). We upper-bound the workload of a task... ...based on the minimum distance between its first and last jobs in the interval. Theorem workload_bounded_by_W : workload of tsk t1 (t1 + delta) ≤ workload bound.

Prosa covers many concepts and is well-documented



	Definition/Let	Lemma/Theorem
Total	714	699

We use short, easy-to-understand definitions.

	Specification	Proofs	Comments
Lines	6661	17104	3442

1 comment for every 2 lines of spec!

(in ~8 person months)

(in ~8 person months)

- Sporadic Task Model
 - Workload-based interference bounds for work-conserving and EDF schedulers

(in ~8 person months)

• Sporadic Task Model

- Workload-based interference bounds for work-conserving and EDF schedulers
- Definition and proofs of correctness and termination of Bertogna and Cirinei's RTA for FP scheduling
 - Same for Bertogna and Cirinei's RTA for EDF scheduling

(in ~8 person months)

• Sporadic Task Model

- Workload-based interference bounds for work-conserving and EDF schedulers
- Definition and proofs of correctness and termination of Bertogna and Cirinei's RTA for FP scheduling
 - Same for Bertogna and Cirinei's RTA for EDF scheduling
- Implementation of a work-conserving scheduler to test for contradictory assumptions

(in ~8 person months)

• Sporadic Task Model

- Workload-based interference bounds for work-conserving and EDF schedulers
- Definition and proofs of correctness and termination of Bertogna and Cirinei's RTA for FP scheduling
 - Same for Bertogna and Cirinei's RTA for EDF scheduling
- Implementation of a work-conserving scheduler to test for contradictory assumptions
- Extensions
 - ➡ Same definitions and proofs for workloads with release jitter
 - Same definitions and proofs for workloads with parallel jobs

novel

results

What we have proven so far (in ~8 person months)

- Sporadic Task Model
 - Workload-based interference bounds for work-conserving and EDF schedulers
 - Definition and proofs of correctness and termination of Bertogna and Cirinei's

Thanks to <u>mature proof assistants and libraries</u>, Prosa allows mechanizing recent and complex schedulability analyses in reasonable time.

- Extensions
 - Same definitions and proofs for workloads with release jitter
 - Same definitions and proofs for workloads with parallel jobs

- 1. Correct recently refuted proofs
 - a) APA scheduling (done! see prosa.mpi-sws.org/apa)
 - b) Self-suspending tasks

- 1. Correct recently refuted proofs
 - a) APA scheduling (done! see prosa.mpi-sws.org/apa)
 - b) Self-suspending tasks
- 2. Verify practical results
 - a) Semi-partitioned scheduling (e.g. C=D)
 - b) Blocking analysis
 - c) Overhead accounting

- 1. Correct recently refuted proofs
 - a) APA scheduling (done! see prosa.mpi-sws.org/apa)
 - b) Self-suspending tasks
- 2. Verify practical results
 - a) Semi-partitioned scheduling (e.g. C=D)
 - b) Blocking analysis
 - c) Overhead accounting
- 3. Investigate how to integrate Prosa with analysis tools and scheduler implementations

Not every proof has to be formalized.

Not every proof has to be formalized.

Pen-and-paper proofs are still useful.

Not every proof has to be formalized.

Pen-and-paper proofs are <u>still useful.</u>

We aim for <u>readable specifications</u>, but writing formal proofs <u>remains non-trivial</u>.

formally proven **PROSA** schedulability analysis

More info at prosa.mpi-sws.org

Mechanized proofs provide an opportunity to avoid the correctness pitfalls in real-time scheduling.

By focusing on <u>readability</u> and by <u>maintaining the</u> <u>established proof culture</u>, mechanized proofs can reach the community at large.

Thanks to <u>mature proof assistants and libraries</u>, Prosa allows mechanizing recent and complex schedulability analyses in reasonable time.

Backup slides

Generality of discrete time

Theorem 6 A sporadic arbitrary-deadline task system T is feasible with respect to continuous schedules iff it is feasible with respect to discrete schedules.

Results about dense time could still be formalized with Coq libraries for real numbers, e.g. Coquelicot.

^[2] Bonifaci, V. and Marchetti-Spaccamela, A., "Feasibility analysis of sporadic real-time multiprocessor task systems," in Proc. of the 18th Annual European Symposium on Algorithms (ESA'10), 2010.

Working with Real Numbers

Coquelicot:

A User-Friendly Library of Real Analysis for Coq

Formalization of limits, continuity, differentiability, Riemann integrals, series, etc.

More info at <u>coquelicot.saclay.inria.fr</u>

Library: Probability Theory

Total/conditional probability, Bayes' theorem, random variables and finite distributions

Lemma prob_decomp: forall A B, $Pr_d[A] = Pr_d[A : \&: B] + Pr_d[A : \&: ~:B].$

Moreira, D. Finite Probability Distributions in Coq (2012).

Related Work

Formalisms for schedulability analysis

Based on the Duration Calculus (DC) interval logic

- Proof of EDF optimality [Yuhua and Chaochen 1994] — improved version [Zhan 2000]
- Schedulability condition of RM [Schuzhen et al. 1999]
- Simplified proofs and review [Xu and Zhan 2008]

Formalisms for schedulability analysis

Based on the Duration Calculus (DC) interval logic

- Proof of EDF optimality [Yuhua and Chaochen 1994]
 improved version [Zhan 2000]
- Schedulability condition of RM [Schuzhen et al. 1999]
- Simplified proofs and review [Xu and Zhan 2008]

+ Formalism reduces ambiguity

Formalisms for schedulability analysis

Based on the Duration Calculus (DC) interval logic

- Proof of EDF optimality [Yuhua and Chaochen 1994] — improved version [Zhan 2000]
- Schedulability condition of RM [Schuzhen et al. 1999]
- Simplified proofs and review [Xu and Zhan 2008]

+ Formalism reduces ambiguity

- Complex logics and manual proofs
- Only uniprocessor scheduling
Earlier mechanized proofs

- Proof of **EDF optimality** using *Nqthm* [Wilding 1998]
- Analysis of the Priority Ceiling and Priority Inheritance Protocols [Zhang et al. 1999] [Dutertre 1999] [Dutertre and Stavridou 2000]
- Schedulability conditions based on task phase using *Coq* [*De Rauglaudre 2012*]
- Certified Computations of Network Calculus in Isabelle/HOL [Mabille et al. 2013]
- Implementation and proof of **EDF optimality** with Propositional Projection Temporal Logic (PPTL) in Coq [*Zhang et al. 2014*]

Earlier mechanized proofs

- Proof of **EDF optimality** using *Nqthm* [Wilding 1998]
- Analysis of the Priority Ceiling and Priority Inheritance Protocols [Zhang et al. 1999] [Dutertre 1999] [Dutertre and Stavridou 2000]
- Schedulability conditions based on task phase using *Coq* [*De Rauglaudre 2012*]
- Certified Computations of Network Calculus in Isabelle/HOL [Mabille et al. 2013]
- Implementation and proof of **EDF optimality** with Propositional Projection Temporal Logic (PPTL) in Coq [Zhang et al. 2014]

+ Mechanically-checked

- No results about multiprocessors

Earlier mechanized proofs

- Proof of **EDF optimality** using *Nqthm* [Wilding 1998]
- Analysis of the Priority Ceiling and Priority Inheritance Protocols [Zhang et al. 1999] [Dutertre 1999] [Dutertre and Stavridou 2000]
- Schedulability conditions based on task phase using *Coq* [*De Rauglaudre 2012*]
- Certified Computations of Network Calculus in Isabelle/HOL [Mabille et al. 2013]
- Implementation and proof of **EDF optimality** with Propositional Projection Temporal Logic (PPTL) in Coq [*Zhang et al. 2014*]

+ Mechanically-checked

- No results about multiprocessors
- Not widely adopted by our community

Model checking and timed automata

- Analysis of uniprocessor FP scheduling using UPPAAL [Fersman at al. 2006]
- Analysis of multiprocessor FP and EDF scheduling of periodic tasks using UPPAAL and NuSMV
 [Guan et al. 2007] [Guan et al. 2008] [Cordovilla et al. 2011]
- Analysis of sporadic tasks based on state exploration and automata reachability [Baker and Cirinei 2007] [Geeraerts et al. 2012] [Burmyakov et al. 2015] [Sun and Lipari 2015]

Model checking and timed automata

- Analysis of uniprocessor FP scheduling using UPPAAL [Fersman at al. 2006]
- Analysis of multiprocessor FP and EDF scheduling of periodic tasks using UPPAAL and NuSMV [Guan et al. 2007] [Guan et al. 2008] [Cordovilla et al. 2011]
- Analysis of sporadic tasks based on state exploration and automata reachability [Baker and Cirinei 2007] [Geeraerts et al. 2012] [Burmyakov et al. 2015] [Sun and Lipari 2015]

+ Multiprocessor, exact schedulability analysis

Model checking and timed automata

- Analysis of uniprocessor FP scheduling using UPPAAL [Fersman at al. 2006]
- Analysis of multiprocessor FP and EDF scheduling of periodic tasks using UPPAAL and NuSMV [Guan et al. 2007] [Guan et al. 2008] [Cordovilla et al. 2011]
- Analysis of sporadic tasks based on state exploration and automata reachability [Baker and Cirinei 2007] [Geeraerts et al. 2012] [Burmyakov et al. 2015] [Sun and Lipari 2015]

+ Multiprocessor, exact schedulability analysis - State explosion (\leq 10 tasks or 4 processors)

Avoiding contradictory assumptions

- 1. **Implement a scheduler function S** using the proof assistant (take pending jobs, sort by priority, assign to CPUs, ...).
- Prove that scheduler S satisfies every requirement of the analysis (work-conserving, enforces priority, etc.) in an assumption-free context.
- 3. Since S is an actual algorithm, it is **impossible that two contradictory assumptions are satisfied by S.**