WHAT REALLY IS pWCET? A RIGOROUS AXIOMATIC PROPOSAL

RTSS 2023 6 December 2023

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg





technische universität dortmund



European Research Council Established by the European Commission



THIS PAPER IN A NUTSHELL

What **exactly** is pWCET? And how does it relate to pET?

Probabilistic Worst-Case Execution Time (pWCET)

Probabilistic Execution Time (pET)







THIS PAPER IN A NUTSHELL

We propose

- → First **fully formal** definitions of pET and pWCET
- → Adequacy property capturing the notion of "IID upper bound on pET" → **Prove** that our proposal of pWCET is adequate in this sense

Independent and identically distributed

What **exactly** is pWCET? And how does it relate to pET?

Probabilistic Worst-Case Execution Time (pWCET)

> Probabilistic Execution Time (pET)









THIS PAPER IN A NUTSHELL

We propose

- → First **fully formal** definitions of pET and pWCET
- → Adequacy property capturing the notion of "IID upper bound on pET" → **Prove** that our proposal of pWCET is adequate in this sense



We formalized our proposal with the Coq proof assistant

- → Semantics of stochastic real-time systems
- → Definitions of pET, pWCET, and the adequacy property
- → Machine-checked proof that pWCET is adequate

The Cog Proof Assistant

<u>coq.inria.fr</u>

Independent and identically distributed

What **exactly** is pWCET? And how does it relate to pET?

Probabilistic Worst-Case Execution Time (pWCET)

> Probabilistic Execution Time (pET)







ANALYSIS OF REAL-TIME SYSTEMS: THE BIG PICTURE

THE BIG PICTURE



Predictions

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg







To get the predictions right, we need:

- → Model with the right **specification**
 - \rightarrow E.g., model must include WCETs to allow (classical) response-time analysis

THE BIG PICTURE

Model of the system



Predictions

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg



To get the predictions right, we need:

- → Model with the right **specification**
 - → E.g., model must include WCETs to allow (classical) response-time analysis
- → Correct model **derivation**
 - → Optimistic WCET bound \implies Wrong predictions

THE BIG PICTURE



Predictions

s



Worst-Case Execution

Time (WCET)



To get the predictions right, we need:

- → Model with the right **specification**
 - → *E.q.*, model must include WCETs to allow (classical) response-time analysis
- → Correct model **derivation** → Optimistic WCET bound \implies Wrong predictions
- → Correct analysis

 \rightarrow Flawed analysis \implies Wrong predictions

THE BIG PICTURE





Worst-Case Execution

Time (WCET)



To get the predictions right, we need:

- → Model with the right **specification**
 - → *E.q.*, model must include WCETs to allow (classical) response-time analysis
- → Correct model **derivation** → Optimistic WCET bound \implies Wrong predictions
- → Correct analysis

 \rightarrow Flawed analysis \implies Wrong predictions

THE BIG PICTURE



Interpretation of common models is pretty straightforward in the deterministic case





SPECIFICATIONS ARE LESS OBVIOUS IN THE STOCHASTIC CASE

To get the predictions right, we need:

- → Model with the right **specification**
 - \rightarrow E.g., model must include WCETs ??? to allow (classical probabilistic) response-time analysis
- → Correct model **derivation** → Optimistic WCET bound \implies Wrong predictions
- → Correct analysis → Flawed analysis ==> Wrong predictions



Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg

11

SPECIFICATIONS ARE LESS OBVIOUS IN THE STOCHASTIC CASE

To get the predictions right, we need:

- → Model with the right **specification**
 - \rightarrow E.g., model must include WCETs ??? to allow (classical probabilistic) response-time analysis
- → Correct model **derivation** → Optimistic WCET bound \implies Wrong predictions
- → Correct analysis → Flawed analysis ==> Wrong predictions



are much less straightforward



Stochastic real-time systems

→ Model of RTSs, where workload parameters are modelled stochastically

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg



Stochastic real-time systems

→ Model of RTSs, where workload parameters are modelled stochastically

Pros:

→ Most systems can tolerate deadline misses \implies Want to take advantage of this

[1] Akesson, Benny, et al. "A comprehensive survey of industry practice in real-time systems."

Question 14 For the most time-critical functions in the system, roughly how frequently can the deadline of a function be missed without causing a system failure. (n = 101)







Stochastic real-time systems

→ Model of RTSs, where workload parameters are modelled stochastically

Pros:

- → Most systems can tolerate deadline misses \implies Want to take advantage of this
- → Allows answering **quantitative** questions

[1] Akesson, Benny, et al. "A comprehensive survey of industry practice in real-time systems."

Question 14 For the most time-critical functions in the system, roughly how frequently can the deadline of a function be missed without causing a system failure. (n = 101)







Stochastic real-time systems

→ Model of RTSs, where workload parameters are modelled stochastically

Pros:

- → Most systems can tolerate deadline misses \implies Want to take advantage of this
- → Allows answering **quantitative** questions
- → Enables analysis of transiently **overloaded systems**
 - → Ubiquitous in practice
 - \rightarrow E.g., FMTV Challenge 2016

[1] Akesson, Benny, et al. "A comprehensive survey of industry practice in real-time systems." [2] Rivas, Juan M., et al. "Calculating latencies in an engine management system using response time analysis with MAST."

Question 14 For the most time-critical functions in the system, roughly how fre-



The total utilization

of that system goes above 100%. Using response time analysis in such situation automatically yields unbounded (infinite) worst-case response times. [2]





DEPENDENCY IN STOCHASTIC RTS





Probabilistic Execution Times (pETs)





Probabilistic Execution Times (pETs) are dependent!





Probabilistic Execution Times (pETs) are dependent!

 Image processing: Two consecutive frames might take similar amounts of compute





Probabilistic Execution Times (pETs) are dependent!

 Image processing: Two consecutive frames might take similar amounts of compute
 Behavior of a prior job influences the state of the cache





Probabilistic Execution Times (pETs) are dependent!

 Image processing: Two consecutive frames might take similar amounts of compute
 Behavior of a prior job influences the state of the cache





Probabilistic Execution Times (pETs) are dependent!

→ Tia *et al.* 1995: computation times are **not** independent [1]

→ Ignoring this fact may lead to incorrect bounds

"Unfortunately, the computation times of individual requests are not statistically independent. [...] As a consequence, the probability of meeting deadlines thus computed may be overly optimistic."

[1] Tia, T-S., et al. "Probabilistic performance guarantee for real-time tasks with varying computation times."





Probabilistic Execution Times (pETs) are dependent!

- → Tia *et al.* 1995: computation times are **not** independent [1]
 - → Ignoring this fact may lead to incorrect bounds

"Unfortunately, the computation times of individual requests are not statistically independent. [...] As a consequence, the probability of meeting deadlines thus computed may be overly optimistic."

→ Limits the application of probability theory tools
 → E.g., convolution is not applicable

[1] Tia, T-S., et al. "Probabilistic performance guarantee for real-time tasks with varying computation times."





PRIOR WORK: pWCET





We note that the actual execution times for a sequence of jobs of a task, which exercise the same or different paths, may well show strong correlations and dependences. It is the *modelling* of the execution times via an appropriate pWCET distribution which enables probabilistic independence to be assumed. (This is similar to the conventional case of a single WCET which can similarly be used in this way, even though the actual execution times of different jobs have strong dependences).

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."





28

We note that the actual execution times for a sequence of jobs of a task, which exercise the same or different paths, may well show strong correlations and dependences. It is the *modelling* of the execution times via an appropriate pWCET distribution which enables probabilistic independence to be assumed. (This is similar to the conventional case of a single WCET which can similarly be used in this way, even though the actual execution times of different jobs have strong dependences).

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."





29

pWCET is a convenient model abstraction to regain independence

→ Goal: enable[~]"IID reasoning"

Independent and identically distributed

We note that the actual execution times for a sequence of jobs of a task, which exercise the same or different paths, may well show strong correlations and dependences. It is the *modelling* of the execution times via an appropriate pWCET distribution which enables probabilistic independence to be assumed. (This is similar to the conventional case of a single WCET which can similarly be used in this way, even though the actual execution times of different jobs have strong dependences).

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."







pWCET is a convenient model abstraction to regain independence → Goal: enable^{*}"IID reasoning" → The mainstream approach to hiding dependence → Unlocks powerful probability theory techniques → Such as convolution, Chernoff bound, etc.

Independent and identically distributed

We note that the actual execution times for a sequence of jobs of a task, which exercise the same or different paths, may well show strong correlations and dependences. It is the *modelling* of the execution times via an appropriate pWCET distribution which enables probabilistic independence to be assumed. (This is similar to the conventional case of a single WCET which can similarly be used in this way, even though the actual execution times of different jobs have strong dependences).

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."





31

pWCET is a convenient model abstraction to regain independence → Goal: enable "IID reasoning" → The mainstream approach to hiding dependence → Unlocks powerful probability theory techniques → Such as convolution, Chernoff bound, etc. → but when **exactly** is a pWCET distribution "appropriate"? Independent and identically distributed

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."



32

THE STATE-OF-THE-ART pWCET DEFINITION

Definition 2. The probabilistic Worst-Case Execution Time (pWCET) distribution for a task is the least upper bound, in the sense of the greater than or equal to operator \succeq (defined below), on the execution time distribution of the jobs of the task for every valid scenario of operation, where a *scenario* of operation is defined as an infinitely repeating sequence of input states (including) both input values and software state variables) and initial hardware states that characterise a feasible way in which recurrent execution of the task may occur.

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."



THE STATE-OF-THE-ART pWCET DEFINITION

► Definition 2. The probabilistic Worst-Case Execution Time (pWCET) distribut a task is the least upper bound, in the sense of the greater than or equal to operator $\succeq 7$ elow), on the execution time distribution of the jobs of the task for every valid scenario is operation, where a scenario of operation is defined as an infinitely repeating sequence of input states (including both input values and software state variables) and initial hardware states that characterise a feasible way in which recurrent execution of the task may occur.

Side note: dominance relation \leq

- → Proposed by Diaz *et al.* in 2004 [2]
- → Partial order on random variables
 - → Similar to stochastic dominance

$\rightarrow \mathscr{A} \leq \mathscr{B} := \forall x, \mathbb{P}[\mathscr{A} \leq x] \geq \mathbb{P}[\mathscr{B} \leq x]$

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems." [2] Diaz, Jose Luis, et al. "Pessimism in the stochastic analysis of real-time systems: Concept and applications."



Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg

34

THE STATE-OF-THE-ART pWCET DEFINITION

▶ Definition 2. The *probabilistic Worst-Case Execution Time (pWCET)* distribution for a task is the least upper bound, in the sense of the greater than or equal to operator \succeq (defined below), on the execution time distribution of the jobs of the task for every valid scenario of operation, where a *scenario* of operation is defined as an infinitely repeating sequence of input states (including both input values and software state variables) and initial hardware states that characterise a feasible way in which recurrent execution of the task may occur.

Pros

- → Gives the right intuition
- → Identifies that "scenario of operation" is the key notion

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."



pWCET DEFINITION IS OPEN TO INTERPRETATION

▶ Definition 2. The *probabilistic Worst-Case Execution Time (pWCET)* distribution for a task is the least upper bound, in the sense of the greater than or equal to operator \succeq (defined below), on the execution time distribution of the jobs of the task for every valid scenario of operation, where a *scenario* of operation is defined as an infinitely repeating sequence of input states (including both input values and software state variables) and initial hardware states that characterise a feasible way in which recurrent execution of the task may occur.

Pros

- → Gives the right intuition
- → Identifies that "scenario of operation" is the key notion

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."

Cons

- → Open to interpretation
 - → Key aspects are stated in **prose** only
 → Not suitable for formal verification


pWCET DEFINITION IS OPEN TO INTERPRETATION

Definition 2. The probabilistic Worst-Case Execution Time (pWCET) distribution for a task is the least upper bound, in the sense of the greater than or equal to operator \succeq (defined below), on the execution time distribution of the jobs of the task for every valid scenario of operation, where a *scenario* of operation is defined as an infinitely repeating sequence of input states (including both input values and software state variables) and initial hardware states that characterise a feasible way in which recurrent execution of the task may occur.

Pros

- \rightarrow Gives the right intuition
- → Identifies that "scenario of operation" is the key notion

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."

Cons

- → Open to interpretation
 - → Key aspects are stated in **prose** only
 - → **Not** suitable for formal verification
- → **Does not** necessarily enable IID-based analyses





[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic timing analysis techniques for real-time systems."

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg

Already noted in [1]



A toy system: [1]

MPI-SWS

- → Time-predictable hardware
- → System has **four** states
- → State cycling through its four possible values
- → Small variability in each of the states
- → Starts with random state



[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic timing analysis techniques for real-time systems."

Already noted in **[1**]

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg



A toy system: [1]

MPI-SWS

- → Time-predictable hardware
- → System has **four** states
- → State cycling through its four possible values
- → Small variability in each of the states
- → Starts with random state



[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic timing analysis techniques for real-time systems."

→ Resulting pET distribution: [1] $\begin{pmatrix} 10 \pm 2 & 20 \pm 2 & 30 \pm 2 & 40 \pm 2 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$

→ Valid pWCET distribution: [1]

 $\begin{pmatrix} 12 & 22 & 32 & 42 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg



A toy system: [1]

MPI-SWS

- → Time-predictable hardware
- → System has **four** states
- → State cycling through its four possible values
- → Small variability in each of the states
- → Starts with random state



Except that → Smallest workload of four consecutive jobs: → (10-2) + (20-2) + (30-2) + (40-2) = 92

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic timing analysis techniques for real-time systems."

→ Resulting pET distribution: [1] $\begin{pmatrix} 10 \pm 2 & 20 \pm 2 & 30 \pm 2 & 40 \pm 2 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$

→ Valid pWCET distribution: [1]

 $\begin{pmatrix} 12 & 22 & 32 & 42 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$



 $\mathbb{P}\left[\sum_{i} \mathsf{pET} \ge 92\right] = 1$

A toy system: [1]

MPI-SWS

- → Time-predictable hardware
- → System has **four** states
- → State cycling through its four possible values
- → Small variability in each of the states
- → Starts with random state



Except that

→ Smallest v → (10 - 2)→ Sum of for → E.g., 12

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic timing analysis techniques for real-time systems."

→ Resulting pET distribution: [1] $\begin{pmatrix} 10 \pm 2 & 20 \pm 2 & 30 \pm 2 & 40 \pm 2 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$

→ Valid pWCET distribution: [1] $(12 \ 22 \ 32 \ 42)$

 $\begin{pmatrix} 12 & 22 & 32 & 42 \\ 1/4 & 1/4 & 1/4 & 1/4 \end{pmatrix}$

workload of four consecutive jobs:

$$P\left[\sum_{4} pET \ge 92\right]$$

$$P\left[\sum_{4} pWCET \ge 92\right]$$

$$P\left[\sum_{4} pWCET \ge 92\right]$$

$$P\left[\sum_{4} pWCET \ge 92\right]$$

 \rightarrow E.g., 12 + 12 + 12 + 12 = 48 has nonzero probability







A toy system: [1]

MPI-SWS

- → Time-predictable hardware
- → System has **four** states
- → State cycling through its four possible values
- → Small variability in each of the states
- → Starts with random state



Except that

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic timing analysis techniques for real-time systems."





MPI-SWS



[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic timing analysis techniques for real-time systems."





OUR PROPOSAL: AXIOMATIC pWCET



Formal definitions of pET and pWCET Definitions that are mathematically formal and unambiguous



Formal definitions of pET and pWCET Definitions that are mathematically formal and unambiguous

Adequacy property: pWCET enables IID analysis

Any sound analysis assuming IID costs must result in a valid estimation





Formal definitions of pET and pWCET Definitions that are mathematically formal and unambiguous

Precise enough to be mechanisable in Coq • pET, pWCET and adequacy property with its proof must be formalisable in Coq proof assistant

Adequacy property: pWCET enables IID analysis

Any sound analysis assuming IID costs must result in a valid estimation







AXIOMATIC pWCET

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."



[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."

Definition 2. The probabilistic Worst-Case Execution Time (pWCET) distribution for a task is the least upper bound, in the sense of the greater than or equal to operator \succeq (defined below), on the execution time distribution of the jobs of the task for every valid scenario of operation, where a scenario of operation is defined as an infinitely repeating sequence of input states (including both input values and software state variables) and initial hardware states that characterise a 1

AXIOMATIC pWCET



Def. 7 ().

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."

Definition 2. The probabilistic Worst-Case Execution Time (pWCET) distribution for a task is the least upper bound, in the sense of the greater than or equal to operator \succeq (defined below), on the execution time distribution of the jobs of the task for every valid scenario of operation, where a *scenario* of operation is defined as an infinitely repeating sequence of input states (including both input values and software state variables) and initial hardware states that characterise a 1

AXIOMATIC pWCET



Def. 7 (**\clubsuit**). A monotonically increasing function $F_i: \mathbb{W} \rightarrow \mathbb{W}$ [0,1] with $F_i(0) = 0$ and $\lim_{t\to\infty} F_i(t) = 1$ is an axiomatic pWCET for a task τ_i if

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."

▶ **Definition 2.** The probabilistic Worst-Case Execution Time (pWCET) distribution for a task is the least upper bound, in the sense of the greater than or equal to operator \succeq (defined below), on the execution time distribution of the jobs of the task for every valid scenario of operation, where a scenario of operation is defined as an infinitely repeating sequence of input states (including) both input values and software state variables) and initial hardware states that characterise a

AXIOMATIC pWCET

Def. 7 (**\clubsuit**). A monotonically increasing function $F_i : \mathbb{W} \rightarrow \mathbb{W}$ [0,1] with $F_i(0) = 0$ and $\lim_{t\to\infty} F_i(t) = 1$ is an axiomatic pWCET for a task τ_i if, for every $J \in \tau_i$ and every fixed arrival sequence $\xi \in \Xi$, there exists a partition \mathfrak{S} (Def. 4) such that

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."







[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg







$$\begin{aligned} \forall S_l \in \mathfrak{S} \text{ s.th. } \mathbb{P}[S_l \wedge \xi] &> 0 : \\ \mathbb{P}\left[\mathcal{C}_J = \vec{c}_J \wedge \forall J' \in G : \mathcal{C}_{J'} = \vec{c}_{J'} | S_l \wedge \xi\right] \\ &= \mathbb{P}\left[\mathcal{C}_J = \vec{c}_J | S_l \wedge \xi\right] \cdot \mathbb{P}\left[\forall J' \in G : \mathcal{C}_{J'} = \vec{c}_{J'} | S_l \wedge \xi\right]. \end{aligned}$$

[1] Davis, Robert Ian, and Liliana Cucu-Grosjean. "A survey of probabilistic schedulability analysis techniques for real-time systems."

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg





$$\begin{aligned} \forall S_l \in \mathfrak{S} \text{ s.th. } \mathbb{P}[S_l \wedge \xi] &> 0 : \\ \mathbb{P}\left[\mathcal{C}_J = \vec{c}_J \wedge \forall J' \in G : \mathcal{C}_{J'} = \vec{c}_{J'} | S_l \wedge \xi\right] \\ &= \mathbb{P}\left[\mathcal{C}_J = \vec{c}_J | S_l \wedge \xi\right] \cdot \mathbb{P}\left[\forall J' \in G : \mathcal{C}_{J'} = \vec{c}_{J'} | S_l \wedge \xi\right]. \end{aligned}$$





$$\begin{aligned} \forall S_l \in \mathfrak{S} \text{ s.th. } \mathbb{P}[S_l \wedge \xi] &> 0 : \\ \mathbb{P}\left[\mathcal{C}_J = \vec{c}_J \wedge \forall J' \in G : \mathcal{C}_{J'} = \vec{c}_{J'} | S_l \wedge \xi\right] \\ &= \mathbb{P}\left[\mathcal{C}_J = \vec{c}_J | S_l \wedge \xi\right] \cdot \mathbb{P}\left[\forall J' \in G : \mathcal{C}_{J'} = \vec{c}_{J'} | S_l \wedge \xi\right]. \end{aligned}$$



ADEQUACY

What Really is pWCET? A Rigorous Axiomatic Proposal

ADEQUACY: FORMAL BASIS FOR IID REASONING

How do we know that an IID-based analysis that uses axiomatic pWCET will obtain a sound bound?





How do we know that an IID-based analysis that uses axiomatic pWCET will obtain a sound bound?





How do we know that an IID-based analysis that uses axiomatic pWCET will obtain a sound bound?





How do we know that an IID-based analysis that uses axiomatic pWCET will obtain a sound bound?

pRT of $J_{i,i}$ obtained by **any** valid IID-based analysis

using axiomatic pWCET









How do we know that an IID-based analysis that uses axiomatic pWCET will obtain a sound bound?

pRT of $J_{i,i}$ obtained by **any** valid IID-based analysis

using axiomatic pWCET









Formal statement is **surprisingly** tricky and involves the notion of "replacement" of pETs with pWCETs

How do we know that an IID-based analysis that uses axiomatic pWCET will obtain a sound bound?

pRT of $J_{i,i}$ obtained by **any** valid IID-based analysis

using axiomatic pWCET





: Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz r.

is file serves for the most part to relate the prob namely to establish that $\mathbb{P} < \mu_{of S} \in [Exc]$

o start the proof. First, we do a case analysis or

 ξ part of the sample space $\Omega_of S$ of system fferent arrival sequence

partition_on_ξ μ.

ndexed) set of events, v t of Ω where the arrival s n technical detail may l

arrival sequences them ders.) paper, one can transfe For further details see

fer.extend_partit extend_partition

nd <mark>ξpart '</mark> are so similar ξi' that are "pickle equi ry strong notion of equiv o elements of similar typ pickle_bij.)

present the proof in a to

, part) (ξi' : Ι ξpart j ξi ξi' → ξi = ξi'.

or any two elements of uivalent, it holds that P{

ineq_ξ_partitioned : part) (ξi' : I ξpart') (ξEQU : pickle_b {[Exc ∩ ξpart⊲{ξi}]} ≤ ℙ<μ_of S'>{[E

t P<µ_of S>{[Exc1]} ≤ P<µ_of S'>{[Exc2 the lemma statement to the hypothesis stateme rmation_is_pRT_monotone_step1 : Exc]} ≤ P<µ_of S'>{[Exc']}.

if we have the inequality with partitions on arriva ioned, then we are done. But how do we prove rove the required inequality by introducing a new blished. We then continue in this manner until r to prove without introducing new hypotheses.

we replace the partition over all arrival sequenc al sequence.

e premise of our hypothesis H_ineq_ξ_partit

artition Epart of the sample space of system fferent arrival sequences...

partition_on_ξ μ.

stem S' and denote it as ξ part'. extend_partition S ξpart j_rep.

ırbitrary pickle-equivalent elements ξi and ξi' : I ξpart) (ξi' : I ξpart'). equivalence : pickle_bij ξi ξi'.

ed, pickle-equivalence is quite strong and we ca correspond to the same arrival sequence ξ . Be sequences are constructed, we can extract an n.

arrival_sequence Job.

dices ξi : I ξpart and ξi' : I ξpart', we w f' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

 c_{1} P<µ_of S>{[Exc n ξpart⊲{ξi}]} ≤ P<µ_of S'>{[Exc' n ξpart'⊲{ξi'}]}.

... we can show that $\mathbb{P} < \mu_{of} S > \{ [Exc1] \} \leq \mathbb{P} < \mu_{of} S' > \{ [Exc2] \}$. Or, in other words, we reduced the lemma statement to the hypothesis statement.

Lemma transformation_is_pRT_monotone_step1 : P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}.

End Step1

Step 2

Now, we know that if we have the inequality with partitions on arrival sequences H ineg E partitioned, then we are done. But how do we prove such an inequality? In

AXIOMATIC pWCET IS ADEQUATE

the next step, we prove the required inequality by introducing a new hypothesis from which it can be established. We then continue in this manner until reaching a hypothesis

Theorem (*paraphrased*). Consider a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the pRT of $J_{i,j}$ in the initial system and $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a simplified system obtained via pWCET F_i . Then $\mathscr{R}_{i,j} \leq \mathscr{R}_{i,j}^{\star}$.

Let $\xi f := (\lambda \omega, arr_seq \omega == \xi) : pred (\Omega_of S)$. Let $\xi f' := (\lambda \omega, arr_seq (proj1 S j_rep \omega) == \xi) : pred (\Omega_of S').$

During this step, we replace indices with predicates.

Hypothesis H_ineq_ξ_fixed : $\mathbb{P}<\mu_{of} S>\{[\xif]\} > 0 \rightarrow \mathbb{P}<\mu_{of} S>\{[Exc \cap \xif]\} \le \mathbb{P}<\mu_{of} S'>\{[Exc' \cap \xif']\}.$

Lemma transformation_is_pRT_monotone_step2 : $\mathbb{P} < \mu_{of S} \{ [Exc \cap \overline{\xi} part \triangleleft \{\overline{\xi}i\}] \} \leq \mathbb{P} < \mu_{of S} < \{ [Exc' \cap \xi part' \triangleleft \{\xii'\}] \}.$

End Step2.

Section Step3.

Step 3

Now, we can forget about ξ part and ξ part' and use ξ , ξ f, and ξ f' instead. Notice that here we use ξ to denote the deterministic arrival sequence. Predicates ξf and $\xi f'$ denote events that result in ξ in systems S and S⁺, respectively.

Variable ξ : arrival_sequence Job. Let $\xi f := (\lambda \omega, arr_seq \omega == \xi)$: pred ($\Omega_of S$). Let $\xi f' := (\lambda \omega, arr_seq (proj1 S j_rep \omega) == \xi) : pred (\Omega_of S').$

Without loss of generality, we can assume that ξ appears with positive probability (otherwise the LHS of the inequality from Step 2 is equal to 0).

Hypothesis H_ ξ _pos_prob : P< μ >{[ξf]} > 0.

In this step, we introduce a partition (that guarantees partition-independence and partitiondominance) of the probability space by exploiting H_axiomatic_pWCET, the assumption of axiomatic pWCET.

Consider some countable type I and define a family of predicates part : $I \rightarrow pred \Omega$. Consider an index Pi : i and assume that part Pi has positive probability. Note that part might depend on Ef.

Next, let us assume the following: we have an event Sf := part Pi in system S such that the event ensures (1) the validity of the pWCET bound and (2) the partition-independence of job j_rep.

Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj1 simply returns the first component of a tuple).

With this, let us assume that the inequality that now includes Sf and Sf' holds:

part_unpack_ ξ : match ξ i with exist ξ I $\mathbb{P} < \mu_o f S > \{ [(Exc1 n \xi f) n S f] \} \le \mathbb{P} < \mu_o f S' > \{ [(Exc2 n \xi f') n S f'] \}$.

Hvpothesis H ineq_Ω_part : \forall (I : countType) (part : I \rightarrow pred Ω) (Pi : I) (ρ : PosProb μ ($\xi f \cap$ part Pi)), let Sf := part Pi : pred (Ω_of S) in let Sf' := $(\lambda \ \omega, \ Sf \ (proj1 \ \overline{S} \ j_rep \ \omega))$: pred $(\Omega_of \ S')$ in

(∀ (x : nat), $\mathbb{F}_{\mu} \{ [odflt0 (\mathcal{C} j_{rep}) | \xi f \cap Sf] \}(x) \ge pWCET_cdf (job_task j_{rep}) x) \rightarrow \mathbb{F}_{\mu} \}$



Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz

is file serves for the most part to relate the prot namely to establish that $\mathbb{P} < \mu$ of $S > \{ [Exc] \}$

o start the proof. First, we do a case analysis or

Epart of the sample space **O** of S of system

partition_on_ξ μ.

ndexed) set of events t of Ω where the arriva

arrival sequences the paper, one can transf . For further details s

fer.extend_parti

nd <mark>ξpart '</mark> are so simila ξi' that are "pickle equ ry strong notion of equiv o elements of similar pickle_bij.)

, part) (ξi' : Ι ξpart j ξi ξi' → ξi = ξi'.

present the proof in a t

or any two elements o uivalent, it holds that P

s pRT monotone step1 Exc]} ≤ P<µ of S'>{[Exc

if we have the inequality with partitions on arriv ioned, then we are done. But how do we prov

e replace the partition over all arrival sequenc

e premise of our hypothesis H_ineq_ξ_partit

artition Epart of the sample space of system fferent arrival sequences.

S' and denote it as Epart extend partition S Epart j rep

irditrary pickle-equivalent elements ξ_1 and ξ_1 : I ξpart) (ξi' : I ξpart'). equivalence : pickle_bij ξi ξi'.

ed, pickle-equivalence is quite strong and we ca correspond to the same arrival sequence ξ . Be sequences are constructed, we can extract an n.

arrival sequence Job.

dices ξi : I ξpart and ξi' : I ξpart', we w f' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

כpart ⊲זכד זןז... P<μ_of S>{[Exc ∩ ξpart⊲{ξi}]} ≤ P<μ_of S'>{[Exc' ∩ ξpart'⊲{ξi'}]}

. we can show that $\mathbb{P}<\mu_{of S} \in \mathbb{E} \times \mathbb{I} \leq \mathbb{P}<\mu_{of S} \in \mathbb{E} \times \mathbb{I}$. Or, in othe words, we reduced the lemma statement to the hypothesis statemen

_emma transformation_is_pRT_monotone_step1 P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}. Ind Step1

Step 2 Now, we know that *if* we have the inequality with partitions on arrival sequence H ineq E partitioned, then we are done. But how do we prove such an inequality? the next step, we prove the required inequality by introducing a new hypothesis from

which it can be established. We then continue in this manner until reaching a hypothesis

AXIOMATIC pWCET IS ADEQUATE

Theorem (*paraphrased*). Consider a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the pRT of $J_{i,j}$ in the initial system and $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a simplified system obtained via pWCET F_i . Then $\mathcal{R}_{i,i} \leq \mathcal{R}_{i,i}^{\star}$.

part) (ξi' : I ξpart') (ξEQU pickle_o

1. Use axiomatic pWCET to construct a "copy" of the initial Now, we can proset about Epart and Eport and use E Ef, and E interact. Notice that Set Set to control determining in systems and provide the effective of the e

Let $\xi f' := (\lambda \omega, arr_seq (proj1 S j_rep \omega) == \xi) : pred (\Omega_of S').$ Vitrue as office protection of the protection of

In this step, we introduce a partition (that guarantees partition-independent and partition-durantees pathy mace y exploring Rx ipmatic of the assumption of axig matic by the support of the assumption section Step3.

Consider some countable type I and define a family of predicates part : $I \rightarrow pred G$ Strip dependent of the validity of the pWCET bound and (2) the partition-independence

of job j_rep.

Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj1 simply returns the first component of a tuple).

With this, let us assume that the inequality that now includes Sf and Sf' holds:

part_unpack_ ξ : match ξ i with exist ξ I $\mathbb{P}<\mu_of S>\{[(Exc1 \cap \xi f) \cap Sf]\} \leq \mathbb{P}<\mu_of S'>\{[(Exc2 \cap \xi f') \cap Sf']\}$

Hypothesis H ineq Ω part : \forall (I : countType) (part : I \rightarrow pred Ω) (Pi : I) (ρ : PosProb μ ($\xi f \cap$ part Pi)), let Sf := part Pi : pred (Ω_of S) in let Sf' := $(\lambda \ \omega, \ Sf \ (proj1 \ \overline{S} \ j_rep \ \omega))$: pred $(\Omega_of \ S')$ in

(∀ (x : nat), $\mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \} \rightarrow \mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \}$ <u>rgey Bozhko</u>, Filip Markovic, Georg von der Brüggen, and Björn Brandenburg

: Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz

is file serves for the most part to relate the prot namely to establish that $\mathbb{P} < \mu_{of S} \in [Exc]$

o start the proof. First, we do a case analysis or

Epart of the sample space Ω of S of system

partition on_ξ μ.

ndexed) set of events t of Ω where the arrival on technical detail mav

arrival sequences the paper, one can transf . For further details s

fer.extend_parti

nd <mark>ξpart '</mark> are so simila ξi' that are "pickle equ ry strong notion of equiv o elements of similar t pickle_bij.)

, part) (ξi' : Ι ξpart j ξi ξi' → ξi = ξi'.

present the proof in a to

or any two elements o uivalent, it holds that P

ation is pRT monotone step1 Exc]} $\leq \mathbb{P} < \mu$ of S'>{[Exc

if we have the inequality with partitions on arriv ioned, then we are done. But how do we prov ablished. We then continue in this manner until r

ve replace the partition over all arrival sequenc

e premise of our hypothesis H_ineq_ξ_partit

artition Epart of the sample space of system fferent arrival sequences..

m S' and denote it as **Epart**'. extend partition S Epart j rep

irditrary pickle-equivalent elements ξ_1 and ξ_1 : I ξpart) (ξi' : I ξpart'). equivalence : pickle_bij ξi ξi'.

ed, pickle-equivalence is quite strong and we ca correspond to the same arrival sequence ξ . Be sequences are constructed, we can extract an n.

arrival sequence Job.

dices ξi : I ξpart and ξi' : I ξpart', we w f' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

כpart ⊲זכד זןז... P<μ_of S>{[Exc n ξpart⊲{ξi}]} ≤ P<μ_of S'>{[Exc' n ξpart'⊲{ξi'}]}

. we can show that \mathbb{P}_{4} of S>{ [Exc1]} $\leq \mathbb{P}_{4}$ of S'>{ [Exc2]}. Or, in other words, we reduced the lemma statement to the hypothesis statement

_emma transformation_is_pRT_monotone_step1 P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}. End Step1

Step 2 Now, we know that *if* we have the inequality with partitions on arrival sequence H ineg E partitioned, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from

which it can be established. We then continue in this manner until reaching a hypothesis

AXIOMATIC pWCET IS ADEQUATE

Theorem (*paraphrased*). Consider a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the pRT of $J_{i,j}$ in the initial system and $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a simplified system obtained via pWCET F_i . Then $\mathcal{R}_{i,i} \leq \mathcal{R}_{i,i}^{\star}$.

part) (ξi': I ξpart') (ξEQU pickle_0

t \mathbb{P}_{μ_0} of $S \in \mathbb{E} \times \mathbb{I}$ burning this step, we replace indices with predicates. the lemma statement to the hypothesis statemed \mathbb{P}_{μ_0} of $S \in \mathbb{I} \times \mathbb{I}$ burning this step, we replace indices with predicates. The lemma statement to the hypothesis statemed \mathbb{P}_{μ_0} of $S \in \mathbb{I} \times \mathbb{I}$ burning this step, we replace indices with predicates. The lemma statement to the hypothesis statemed \mathbb{P}_{μ_0} of $S \in \mathbb{I} \times \mathbb{I} \times \mathbb{I}$ burning this step, we replace indices with predicates.

1. Use axiomatic pWCET to construct a "copy" of the initial Now, we can proset about Epart and Eport and use E Ef, and E interact. Notice that Set Set to control determining in systems and provide the effective of the e

In this step, we introduce a partition (that guarantaes partition-independence and partition-dependence of the repair type active active active active and partition-of axic matic by the comparison of the simplified system section Step3.

Consider some countable type I and define a family of predicates part : $I \rightarrow pred \Omega$ Strip dependent of the validity of the pWCET bound and (2) the partition-independence

of job j_rep.

Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj1 simply returns the first component of a tuple).

With this, let us assume that the inequality that now includes Sf and Sf' holds:

part_unpack_ ξ : match ξ i with exist ξ I $\mathbb{P}<\mu_of S>\{[(Exc1 \cap \xi f) \cap Sf]\} \leq \mathbb{P}<\mu_of S'>\{[(Exc2 \cap \xi f') \cap Sf']\}$

Hypothesis H ineq Ω part : \forall (I : countType) (part : I \rightarrow pred Ω) (Pi : I) (ρ : PosProb μ ($\xi f \cap$ part Pi)), let Sf := part Pi : pred (Ω_of S) in let Sf' := $(\lambda \ \omega, \ Sf \ (proj1 \ \overline{S} \ j_rep \ \omega))$: pred $(\Omega_of \ S')$ in

(∀ (x : nat), $\mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \} \rightarrow \mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \}$ <u>rgey Bozhko</u>, Filip Markovic, Georg von der Brüggen, and Björn Brandenburg





Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz

is file serves for the most part to relate the prot namely to establish that $\mathbb{P} < \mu$ of $S > \{ [Exc] \}$

o start the proof. First, we do a case analysis or

Epart of the sample space **O** of S of system

paper, one can transf . For further details s

fer.extend part:

nd **ξpart'** are so simila ξi' that are "pickle equ ry strong notion of equi o elements of similar pickle_bij.)

, . part) (ξi' : I ξpart j ξi ξi' → ξi = ξi'.

present the proof in a t

or any two elements o uvalent, it holds that

s_pRT_monotone_step1 Exc]} $\leq \mathbb{P} < \mu_o f S' > \{ [Exc'] \}$

if we have the inequality with partitions on arriv ioned, then we are done. But how do we prov

e replace the partition over all arrival sequenc

e premise of our hypothesis H_ineq_ξ_partit

artition Epart of the sample space of system fferent arrival sequences.

n S' and denote it as Epart extend partition S ξpart j rep

irditrary pickle-equivalent elements ξ_1 and ξ_1 : I ξpart) (ξi' : I ξpart'). equivalence : pickle_bij ξi ξi'.

ed, pickle-equivalence is quite strong and we ca correspond to the same arrival sequence ξ . Be sequences are constructed, we can extract an n.

arrival sequence Job.

dices ξi : I ξpart and ξi' : I ξpart', we w f' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

we can show that $\mathbb{P}<\mu_{of S} \in \mathbb{E} \times \mathbb{I} \leq \mathbb{P}<\mu_{of S} \in \mathbb{E} \times \mathbb{I}$. Or, in other words. we reduced the lemma statement to the hypothesis statement

_emma transformation_is_pRT_monotone_step1 P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}. End Step1

Step 2 Now, we know that *if* we have the inequality with partitions on arrival sequence H ineq E partitioned, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from

which it can be established. We then continue in this manner until reaching a hypothesis

AXIOMATIC pWCET IS ADEQUATE

Theorem (*paraphrased*). Consider a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the pRT of $J_{i,j}$ in the initial system and $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a simplified system obtained via pWCET F_i . Then $\mathcal{R}_{i,i} \leq \mathcal{R}_{i,i}^{\star}$.

part) (ξi': I ξpart') (ξEQU pickle_0

 $P<\mu_{of S>{[Exc1]} \leq P<\mu_{of S'>{[Exc2]}}$ During this step, we replace indices with predicates. $P<\mu_{of S>{[Exc1]} \leq P<\mu_{of S} = P$

1. Use axiomatic pWCET to construct a "copy" of the initial Now, we can proset about Epart and Eport and use E Ef, and E interest. Notice that Solves to the first determining in systems and provide the estimated in the system is an estimated in the estimated in the system is an estimated in the estimated in the system is an estimated in the estin the estin the es

2. In this step, we introduce a partition (that guarantees partition-independent and partition-demonstrate products and partition-independent and partition-of axid matic by the trade of the same trade of the same time to the simplified system section Step3.

Consider some countable type I and define a family of predicates part : $I \rightarrow pred \Omega$ Consider an inder P1: 1 and assume that per Pi has popitive probability. Note that Strip dependent Stically dominates the submate of the probability of the pWCET bound and (2) the partition-independence the event ensures (1) the validity of the pWCET bound and (2) the partition-independence

of job j_rep.

Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj1 simply returns the first component of a tuple).

With this, let us assume that the inequality that now includes Sf and Sf' holds:

part_unpack_ ξ : match ξ i with exist ξ I $\mathbb{P}<\mu_of S>\{[(Exc1 \cap \xi f) \cap Sf]\} \leq \mathbb{P}<\mu_of S'>\{[(Exc2 \cap \xi f') \cap Sf']\}$

Hypothesis H ineq Ω part : \forall (I : countType) (part : I \rightarrow pred Ω) (Pi : I) (ρ : PosProb μ ($\xi f \cap$ part Pi)), let Sf := part Pi : pred (Ω_of S) in let Sf' := $(\lambda \ \omega, \ Sf \ (proj1 \ \overline{S} \ j_rep \ \omega))$: pred $(\Omega_of \ S')$ in

(∀ (x : nat), $\mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \} \rightarrow \mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \}$





: Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz

is file serves for the most part to relate the prob namely to establish that $\mathbb{P}<\mu_{of S} \{ [Exc] \}$

o start the proof. First, we do a case analysis or

Epart of the sample space Ω of S of system

partition on ξ μ.

ndexed) set of events. t of **Ω** where the arrival n technical detail may

arrival sequences the ders.) paper, one can transfe For further details set

fer.extend_parti extend_partition

nd <mark>ξpart '</mark> are so simila ξi' that are "pickle equi ry strong notion of equiv o elements of similar ty pickle_bij.)

çart) (ξi' : I ξpart j ξi ξi' → ξi = ξi',

present the proof in a to

or anv two elements of uivalent, it holds that P

 $P<\mu_{of S>{[Exc1]} \leq P<\mu_{of S'>{[Exc:}] }$ During the step, we represent the lemma statement to the hypothesis statement to the hypothesis statement is provided at the statement is prov

if we have the inequality with partitions on arriva ioned, then we are done. But how do we prove ove the required inequality by introducing a nev ablished. We then continue in this manner until r

we replace the partition over all arrival sequenc al sequence

e premise of our hypothesis H_ineq_ξ_partit

artition Epart of the sample space of system fferent arrival sequences...

partition_on_ξ μ.

stem S' and denote it as Epart' extend partition S Epart j rep

rbitrary pickle-equivalent elements ξi and ξi : I ξpart) (ξi' : I ξpart'). equivalence : pickle_bij ξi ξi'.

ed, pickle-equivalence is quite strong and we ca correspond to the same arrival sequence ξ . Be sequences are constructed, we can extract an n.

arrival sequence Job.

dices ξi : I ξpart and ξi' : I ξpart', we w f' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

chair dict ili … P<μ_of S>{[Exc ∩ ξpart⊲{ξi}]} ≤ P<μ_of S'>{[Exc' ∩ ξpart'⊲{ξi'}]}.

. we can show that $\mathbb{P} < \mu_{of} S > \{ [Exc1] \} \leq \mathbb{P} < \mu_{of} S' > \{ [Exc2] \}$. Or, in other words, we reduced the lemma statement to the hypothesis statement.

Lemma transformation_is_pRT_monotone_step1 P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}. End Step1

Step 2 Now, we know that *if* we have the inequality with partitions on arrival sequence H ineg E partitioned, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from

which it can be established. We then continue in this manner until reaching a hypothesis

AXIOMATIC pWCET IS ADEQUATE

Theorem (*paraphrased*). Consider a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the pRT of $J_{i,j}$ in the initial system and $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a simplified system obtained via pWCET F_i . Then $\mathscr{R}_{i,i} \leq \mathscr{R}_{i,i}^{\star}$.

part) (ξ i' : I ξ part') (ξ EQU : pickle_p Let ξ f := ($\lambda \omega$, arr seq $\omega = \xi$) : pred (Ω of S). [[Exc $n \xi$ partd{ ξ i}]} $\leq P < \mu_0 f S'>$ {[E

1. Use axiomatic pWCET to construct a "copy" of the initial Now, we can proset about Epart and Eport and use E Ef, and E interest. Notice that Set Set to can be the determining in systems and provide the estimated in the system is an esti

Vertures of generative S as une has Experiment of the productive model of S'). (otherwise the Existence quarty from Step 2) sequence of the productive productive of the productive the Existence of the interpretative of the productive of the pro

2. In this step, we introduce a partition (that guarantees partition-independent) and partition-d monoportise repeating action of the product of the assumption of axiomatic well. The assumption of axiomatic well.

Consider some countable type I and define a family of predicates part : $I \rightarrow pred \Omega$. Consider an inder Pi : i and assume that per Pi has positive probability. Note that Strip dependent Strip and assume that per Pi has positive probability. Note that in system 5 such that the value of the pwcet bound and (2) the partition-independence of in b i non

of job j_rep.

Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj1 simply returns the first component of a tuple).

With this, let us assume that the inequality that now includes Sf and Sf' holds:

part_unpack_ ξ : match ξ i with exist ξ I $\mathbb{P}<\mu_of S>\{[(Exc1 \cap \xi f) \cap Sf]\} \leq \mathbb{P}<\mu_of S'>\{[(Exc2 \cap \xi f') \cap Sf']\}$

Hypothesis H ineq Ω part : \forall (I : countType) (part : I \rightarrow pred Ω) (Pi : I) (ρ : PosProb μ ($\xi f \cap$ part Pi)), let Sf := part Pi : pred (Ω_of S) in let Sf' := $(\lambda \ \omega, \ Sf \ (proj1 \ \overline{S} \ j_rep \ \omega))$: pred $(\Omega_of \ S')$ in

(∀ (x : nat), $\mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \} \rightarrow \mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \}$ rgey Bozhko, Filip Markovic, Georg von der Brüggen, and Björn Brandenburg

Step-by-step Proof of Theorem text {Task : TaskType}
 {pwCET pmf : ProbWCET Task t S := {| Ω_of := Ω; μ_of := μ; A_of := A; e_of := e |}. /ariable j_rep : Job. .et S' := replace_job_pET j_rep or convenience, let tsk denote i 's and let **µ_tsk** denote the measu t µ_tsk := match pWCET_pmf with | Build_ProbWCET pWCET nonneg : {| pmf := pWCET tsk; pmf_pos := nonneg tsk; pmf_sum1 := sum1 tsk |} Next, consider an arbitrary job j of any tas pickle_bij ξi ξi' → ξi = ξi' Now, recall that we present the proof in a top-down ($C \rightarrow Goal$) fashion, starting with the overall theorem. So, assuming that for any two elements of partitions $\xi i : I \xi_{part} and \xi i' : I \xi_{part} that are "pickle"-equivalent, it holds that P{[Exc1 \cap \xi_{part} \{\xi_i\}]} <math>\leq P{[Exc2 \cap \xi_{part}]}$... we can show that $P<\mu_{of S}{Exc1} > P<\mu_{of S'}{Exc2}$. Or, in other words, we reduced the lemma statement to the hypothesis statement. Lemma transformation_is_pRT_monotone_step1 : Psu of S>{[Exc]} < Psu of S'>{[Exc']}. Step 2 Now, we know that if we have the inequality with partitions on arrival sequences it_ineq_fpartitioned, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from which it can be established. We then continue in this manner unit reaching a hypothesis that is easy enough to prove without introducing new hypotheses.



: Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz

is file serves for the most part to relate the prob namely to establish that $\mathbb{P}<\mu_{of S} \in [Exc]$

o start the proof. First, we do a case analysis or

Epart of the sample space Ω of S of system

partition on ξ μ.

ndexed) set of events. t of **Ω** where the arrival n technical detail may

arrival sequences the ders.) paper, one can transfe . For further details se

fer.extend_parti extend_partition

nd <mark>ξpart '</mark> are so simila ξi' that are "pickle equi ry strong notion of equiv o elements of similar ty pickle_bij.)

ς . part) (ξi' : Ι ξpart j ξi ξi' → ξi = ξi'.

present the proof in a to

or anv two elements of uivalent, it holds that P-

 $P<\mu_{of S>{[Exc1]} \leq P<\mu_{of S'>{[Exc2]}$ During tills Step, no represent to the hypothesis station is provided in the p

if we have the inequality with partitions on arriva ioned, then we are done. But how do we prove rove the required inequality by introducing a nev ablished. We then continue in this manner until r

we replace the partition over all arrival sequenc al sequence.

e premise of our hypothesis H_ineq_ξ_partit

artition Epart of the sample space of system fferent arrival sequences...

partition_on_ξ μ.

stem S' and denote it as ξpart'. extend partition S Epart i rep

irbitrary pickle-equivalent elements ξi and ξi : I ξpart) (ξi' : I ξpart'). equivalence : pickle_bij ξi ξi'.

ed, pickle-equivalence is quite strong and we ca correspond to the same arrival sequence ξ . Be sequences are constructed, we can extract an n.

arrival sequence Job. part_unpack_ξ : match ξi with exist ξ Ι

dices ξi : I ξpart and ξi' : I ξpart', we w f' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

chair dict ili … P<μ_of S>{[Exc ∩ ξpart⊲{ξi}]} ≤ P<μ_of S'>{[Exc' ∩ ξpart'⊲{ξi'}]}.

. we can show that $\mathbb{P} < \mu_{of} S > \{ [Exc1] \} \leq \mathbb{P} < \mu_{of} S' > \{ [Exc2] \}$. Or, in other words, we reduced the lemma statement to the hypothesis statement.

End Step1

Step 2 Now, we know that *if* we have the inequality with partitions on arrival sequence H ineg E partitioned, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from

which it can be established. We then continue in this manner until reaching a hypothesis

AXIOMATIC pWCET IS ADEQUATE

Theorem (*paraphrased*). Consider a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the pRT of $J_{i,j}$ in the initial system and $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a simplified system obtained via pWCET F_i . Then $\mathscr{R}_{i,i} \leq \mathscr{R}_{i,i}^{\star}$.

part) (ξ i' : I ξ part') (ξ EQU : pickle_p Let ξ f := ($\lambda \omega$, arr seq $\omega = \xi$) : pred (Ω of S). [[Exc $n \xi$ partd{ ξ i}]} $\leq P < \mu_0 f S'>$ {[E

1. Use axiomatic pWCET to construct a "copy" of the initial Now, we can proset about Epart and Eport and use E Ef, and E interest. Notice that Solves to the first determining in systems and provide the estimated in the system is an estimated in the estimated in the system is an estimated in the estimated in the system is an estimated in the estin the estin the es

Let $\xi_{f'} := (\lambda, \omega, arr_{seq}(proj1 s j_{rep}, \omega) := \xi)$; pred (Q_of s'). verburyes of perturbed S assume the sequence of the probability D and have distribution F_i .

2. In this step, we introduce a partition (that guarantees partition-independent) and partition-d monoportise repeating action of the product of the assumption of axiomatic well. The assumption of axiomatic well.

Consider some countable type I and define a family of predicates part : $I \rightarrow pred \Omega$. Consider an inder P1: 1 and assume that part P1 has positive probability. Note that restrict dree por a Stical V dominates the provide the positive probability. Note that the event ensures (1) the validity of the pWCET bound and (2) the partition-independence of inb 1 rep.

of job j_rep.

Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj1 simply returns the first component of a tuple).

With this, let us assume that the inequality that now includes Sf and Sf' holds:

 $\mathbb{P} < \mu_{of S} \{ [(Excl n \xi f) n S f] \} \leq \mathbb{P} < \mu_{of S'} \{ [(Excl n \xi f') n S f'] \}$

Hypothesis H ineq Ω part : \forall (I : countType) (part : I \rightarrow pred Ω) (Pi : I) (ρ : PosProb μ ($\xi f \cap$ part Pi)), let Sf := part Pi : pred (Ω_of S) in let Sf' := $(\lambda \ \omega, \ Sf \ (proj1 \ \overline{S} \ j_rep \ \omega))$: pred $(\Omega_of \ S')$ in

(∀ (x : nat), $\mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \} \rightarrow \mathbb{F}_{\mu} = \{ odflt0 (c_j_rep) \mid \xi f \cap Sf \} (x) \ge pWCET_cdf (job_task j_rep) x \}$ <u>rgey Bozhko</u>, Filip Markovic, Georg von der Brüggen, and Björn Brandenburg



this file, we prove Theorem 1 presented in the paper "What really is pWCET? / igorous Axiomatic Definition of pWCET" by Bozhko et al. (RTSS'23).

Step-by-step Proof of Theorem 1

text {Task : TaskType}
 {pWCET pmf : ProbWCET Task}

ontext {Job : finType}
 {job task : JobTask Job Task

/ariable Ω : countType. /ariable μ : measure Ω. /ariable A : JobArrivalRV Job Ω μ /ariable C : JobCostRV Job Ω μ.

st S := {| Ω_of := Ω; μ_of := μ; A_of := A; e_of := e |}.

vpothesis H_axiomatic_pWCET : axiomatic_pWCET (µ_of S) (job_arrival := A_of S) (job_cost := e_of

/ariable j_rep : Job. .et S' := replace_job_pET j_rep S

or convenience, let tsk denote i 's task tsk := job_task j_rep. and let <u>µ_tsk</u> denote the measure induced by

tt µ_tsk :=
match pWCET_pmf with
| Build_ProbWCET pWCET nonneg sun
{| pmf := pWCET tsk;
 pmf_pos := nonneg tsk;
 pmf_sum1 := sum1 tsk |}

Next, consider an arbitrary job j of any tas

(job_arrival := A_of S') (sched S') horizon i w)

pickle_bij Ei Ei' - Ei = Ei'. Now, recall that we present the proof in a top-down (C - Goal) fashion, starting with the overall theorem.

typothesis H_ineq_ξ_partitioned : Ψ (ξi : I ξpart) (ξi ' : I ξpart') (ξΕQU : pickle_bij ξi ξi'), ℙeq_of S>{[Exc ∩ ξpart⊲{ξi}]} ≤ ℙeq_of S'>{[Exc' ∩ ξpart'⊲{ξi'}] ... we can show that $P<\mu_{0}f S>{[Exc1]} = P<\mu_{0}f S'>{[Exc2]}. Or, in other words, we reduced the lemma statement to the hypothesis statement.$ Lemma transformation_is_pRT_monotone_step1 :
 P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}.

Now, we know that *if* we have the inequality with partitions on arrival sequences $H_{_ineq__partitioned}$, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by intruducing a new hypothesis from which it can be established. We then continue in this manner until reaching a hypothesis that is easy enough to prove without introducing new hypotheses.

Step 2

Step 2

et Epart := partition on ξ μ.

et Epart' := extend_partition S Epart j_re

ariables (ξi : Ι ξpart) (ξi' : Ι ξpart'). ypothesis ξ_equivalence : pickle_bij ξi ξi

ypothesis H_ineq_E_fixed : Pcu of S>{[Ef]} > 0 \rightarrow Pcu of S>{[Exc n Ef]} \leq Pcu of S'>{[Exc' n E

to denote the deterministic arrival sequence. Provide the sequence of the sequ Tiddle (, and the sequence solution of a solution of s). et $\xi f := (\lambda \omega, \operatorname{arr_seq} \omega = \xi) : \operatorname{pred} (\Omega_o f S).$ et $\xi f' := (\lambda \omega, \operatorname{arr_seq} (\operatorname{proj1} S j_rep \omega) == \xi) : \operatorname{pred} (\Omega_o f S)$

thesis H_ξ_pos_prob : P<μ>{[ξf]} > 0.

vith this, let us assume that the inequality that n

P<u>{[e_fix C [:: j_rep] n e_fix C jobs | Ef n Sf]] <µ_of S>{[(Exc n ξf) n Sf]} ≤ P<µ_of S'>{[(Exc' n ξf') n S

emma transformation_is_pRT_monotone_step3 : P<μ_of S>{[Exc ∩ ξf]} ≤ P<μ_of S'>{[Exc' ∩ ξf']]

Let Sf := part Pi : pred ($\Omega_o f$ S). Let Sf' := ($\lambda \omega$, Sf (proil S i rep ω)) : pred (Ω of S'

ariable p1 : PosProb (µ_of S) (ξf n Sf). ariable p2 : PosProb (µ of S') (ξf' n Sf'). pothesis H_pWCET_bounds_cond_cdf : ∀ (x : nat), F<μ>{[odflt0 (@ j_rep) | ξf n Sf]}(x)

]_rep (notif jobs → P<µ>{[@_fix C [:: j_rep] ∩ @_fix C jobs | = P<µ>{[@ fix C [:: i rep] | Ef ∩ Sf]} ×

fypothesis H_ineq_conditional :
 P<µ_of S>{[Exc | ξf ∩ Sf]} ≤ P<µ_of S'>{[Exc' | ξf' ∩ Sf

emma transformation_is_pRT_monotone_step4 : P=u_of S>{[Exc n ξf n Sf]} ≤ P=u_of S'>{[Exc' n ξf' n Sf']

example, given $\omega \ in \ \Omega_{of} \ S, \ compute_costs \ returns a vector of all costs fixed for this specific evolution <math display="inline">\omega.$

Let compute_costs (ω : Ω_{of} S) := job.compute_costs ω (job_cost := e_{of} S). Let compute_costs' (ω : Ω_{of} S') := job.compute_costs ω (job_cost := e_{of} S'). For simplicity, let ${\cal R}$ denote a function that computes the response time of any job for given fixed vectors A and C. Let *R* := schedulerAC_to_rtAC horizon ζ.

In this step, we replace events Exc and Exc⁺ with events $\lambda \omega = \text{exceeds}$ (arrival times. Note that previously we had a general random variable describing the response-time distribution, but now we have algorithm *R* instead. ction Step5.

Note that here we assume that we are given any vector A describing job Now use there easure the true that we are given any vector A describing job arrivals witho estriction that it must agree with ξ . Inside of the proof, we indeed construct A as a ransformation of ξ ; however, for further proofs, it is not relevant, so we just forget this iformation and use a generic function Job – option instant.

ypothesis H_ineq_algorithmic_R : Y (A : Job - option instant), Pq_uof S>{[A w_ exceeds (X A (compute_costs w) j) r | ξf n Sf]} s P<u of S>{[A w_ exceeds (X A (compute costs 'w) j) r | ξf' n Sf]}; uality involving Exc and Exc' is implied by the inequality involving a



: Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz r.

is file serves for the most part to relate the prob namely to establish that $\mathbb{P}<\mu_{of S} \in [Exc]$

o start the proof. First, we do a case analysis or

Exact ξ part of the sample space $\Omega_of S$ of system

partition on ξ μ.

ndexed) set of events. t of Ω where the arrival n technical detail may l

arrival sequences the ders.) paper, one can transfe . For further details se

fer.extend_partit extend_partition

nd <mark>ξpart'</mark> are so simila ξi' that are "pickle equi ry strong notion of equiv o elements of similar ty pickle_bij.)

ς . part) (ξi' : Ι ξpart j ξi ξi' → ξi = ξi'.

present the proof in a to

or any two elements of uivalent, it holds that P-

 $P<\mu_{of S>\{[Exc1]\} \le P<\mu_{of S'>\{[Exc:] \\ P<\mu_{of S>\{[Exc1]\} \le P<\mu_{of S'>\{[Exc:] \\ P<\mu_{of S>\{[\xif]\} > 0 \rightarrow P<\mu_{of S>\{[Exc n \xif]\} \le P<\mu_{of S'>\{[Exc' n \xif']\}. \\ P<\mu_{of S>\{[\xif]\} > 0 \rightarrow P<\mu_{of S>\{[Exc' n \xif']\}. \\ P<\mu_{of S>\{[\xii']\} > 0 \rightarrow P<\mu_{of S}]$

if we have the inequality with partitions on arriva ioned, then we are done. But how do we prove ove the required inequality by introducing a nev ablished. We then continue in this manner until r

we replace the partition over all arrival sequenc al sequence.

e premise of our hypothesis H_ineq_ξ_partit

artition Epart of the sample space of system : fferent arrival sequences...

partition_on_ξ μ.

stem S' and denote it as Epart'. extend_partition S ξpart j_rep.

Irbitrary pickle-equivalent elements ξi and ξi : I ξpart) (ξi' : I ξpart'). equivalence : pickle_bij ξi ξi'.

ed, pickle-equivalence is quite strong and we ca correspond to the same arrival sequence ξ . Be sequences are constructed, we can extract an n.

arrival sequence Job. part_unpack_ξ : match ξi with exist ξ I

dices ξi : I ξpart and ξi' : I ξpart', we w f' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

chair dict ili … P<μ_of S>{[Exc ∩ ξpart⊲{ξi}]} ≤ P<μ_of S'>{[Exc' ∩ ξpart'⊲{ξi'}]}.

. we can show that $\mathbb{P} < \mu_{of} S > \{ [Exc1] \} \leq \mathbb{P} < \mu_{of} S' > \{ [Exc2] \}$. Or, in other words, we reduced the lemma statement to the hypothesis statement.

Lemma transformation_is_pRT_monotone_step1 : $\mathbb{P}_{\mu_of S} [Exc] \le \mathbb{P}_{\mu_of S'} [Exc']$. End Step1

Step 2 Now, we know that if we have the inequality with partitions on arrival sequences H ineq E partitioned, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from

which it can be established. We then continue in this manner until reaching a hypothesis

AXIOMATIC pWCET IS ADEQUATE

Theorem (*paraphrased*). Consider a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the pRT of $J_{i,j}$ in the initial system and $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a simplified system obtained via pWCET F_i . Then $\mathscr{R}_{i,i} \leq \mathscr{R}_{i,i}^{\star}$.

ineq_ξ_partitioned : part) (ξi' : I ξpart') (ξEQU : pickle_b Let ξf' := (λ ω, arr_seq ω == ξ) : pred (Ω_of S). {[Exc ∩ ξpart⊲{ξi}]} ≤ P<μ_of S'>{[E

1. Use axiomatic pWCET to construct a "copy" of the initial Now, we can proset about Epart and Eport and use E Ef, and E interest. Notice that Solves to the first determining in systems and provide the estimated in the system is an estimated in the estimated in the system is an estimated in the estimated in the system is an estimated in the estin the estin the es

Let $\xi_{f'} := (\lambda, \omega, arr_{seq}(proj1 s j_{rep}, \omega)) == \xi_{f'}$: pred (Q_of S'). verburyes of pertrait the Sassine has the predictive protability D and have distribution F_{i} .

2. In this step, we introduce a partition (that guarantees partition-independence and partition-dependence of the repair of the repair of the sumption of the simplified system of axis matic by the transformatic of the sumption the simplified system section Step3.

Consider some countable type I and define a family of predicates part : $I \rightarrow pred \Omega$. Consider an indet P1: 1 and assume that part P1 has positive probability. Note that is the open of Stically dominates the original pRT $\mathscr{P}_{i,j}$ the event ensures (1) the validity of the pWCET bound and (2) the partition-independence

of job j_rep.

Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj1 simply returns the first component of a tuple).

With this, let us assume that the inequality that now includes Sf and Sf' holds:

 $\mathbb{P} < \mu_{of S} \in \{ (Exc1 \cap \xi f) \cap Sf \} \leq \mathbb{P} < \mu_{of S} > \{ (Exc2 \cap \xi f') \cap Sf' \}$

Hypothesis H ineq Ω part : \forall (I : countType) (part : I \rightarrow pred Ω) (Pi : I) (ρ : PosProb μ ($\xi f \cap$ part Pi)), let Sf := part Pi : pred (Ω_of S) in let Sf' := $(\lambda \ \omega, \ Sf \ (proj1 \ \overline{S} \ j_rep \ \omega))$: pred $(\Omega_of \ S')$ in

(∀ (x : nat), $\mathbb{F}_{\mu} \{ [odflt0 (\mathcal{C} j_{rep}) | \xi f \cap Sf] \}(x) \ge pWCET_cdf (job_task j_{rep}) x) \rightarrow \mathbb{F}_{\mu} \}$ <u>rgey Bozhko</u>, Filip Markovic, Georg von der Brüggen, and Björn Brandenburg



prosa.model Require Import processor.idea

Step 2

ain, consider a partition Epart of the sample space of system S into events

Variable ξ : arrival_sequence Job. Hypothesis H part uppack E : match Ei with exist E IN \rightarrow E end =

Let $\xi f := (\lambda \ \omega, \ arr_seq \ \omega == \xi) : pred (\Omega_of S).$ Let $\xi f' := (\lambda \ \omega, \ arr_seq (proj1 S j_rep \ \omega) == \xi) : pred (\Omega_of S'$

use ξ to denote the deterministic arrival sequence. Here events that result in ξ in systems S and S ', respectively.

Vith this, let us assume that the inequality that now includes Sf and Sf

(C : Job → option work) (jobs : seq Job),

Lemma transformation_is_pRT_monotone_step3 : P<µ_of S>{[Exc n Ef]} ≤ P<µ_of S'>{[Exc' n Ef']}.

_independence : tion work) (jobs : seq Job), j_rep \notin jobs → P<µ>{[e_fix c [:: j_rep] ∩ e_fix c jobs | ξf ∩ Sf]} = P<µ>{[e_fix c [:: j_rep] | ξf ∩ Sf]} × P<µ>{[e_fix

this step, we transform our inequality by moving $\xi f \cap S f$ and $\xi f' \cap S f'$ to the

Hypothesis H_ineq_conditional : P<μ_of S>{[Exc | ξf ∩ Sf]} ≤ P<μ_of S'>{[Exc' | ξf' ∩ Sf']}

Lemma transformation_is_pRT_monotone_step4 : P<U_of S>{[Exc n ξf n Sf]} ≤ P<U_of S'>{[Exc' n ξf' n Sf']}.

example, given $\omega \setminus \text{in } \Omega_of S, \text{ compute_costs}$ returns a vector of all costs fixed for this specific evolution $\omega.$

For simplicity, let \mathscr{R} denote a function that computes the response time of any job for given fixed vectors A and C.

In this step, we replace events Exc and Exc' with events $\lambda \omega \rightarrow \text{exceeds}$ ($\Re A$ (compute costs ω) j) r, where the arrival times are fixed to be a *specific vector* of

response-time distribution, but now we have algorithm *s* instead.

Note that here we assume that we are given any vector A describing job arrivals withour restriction that it must agree with ξ . Inside of the proof, we indeed construct A as a transformation of ξ ; however, for further proofs, it is not relevant, so we just forget this information and use a generic function Job – option instant.

uality involving Exc and Exc' is implied by the inequality involving

typothesis H_ineq_algorithmic_R : V (A: Job → option instant), P<u_of S<[{ \u03c6 w, exceeds (R A (compute_costs w) j) r | ξf n Sf]} ≤ P<u_of S'>{[\u03c6 w, exceeds (R A (compute_costs' w) j) r | ξf' n Sf'

Let \mathcal{R} := schedulerAC_to_rtAC horizon ζ .

ection Step5.

Let compute_costs (ω : Ω_{of} S) := job.compute_costs ω (job_cost := e_{of} S). Let compute_costs' (ω : Ω_{of} S') := job.compute_costs ω (job_cost := e_{of} S').

 Remark ξf_and_Sf_eq_prob :

 P<µ_of S>{[ξf ∩ Sf]} = P<µ_of S'>{[ξf' ∩ Sf']}

j_rep \notin jobs → P<µ>{[e_fix C [:: j_rep] ∩ e_fix C jobs | ξf ∩ Sf]} = P<µ>{[e_fix C [:: j_rep] | ξf ∩ Sf]} × P<µ>{[e_fix C jobs

P<μ_of S>{[(Exc n ξf) n Sf]} ≤ P<μ_of S'>{[(Exc' n ξf') n Sf'

pthesis H E pos prob : $P < u > \{[Ef]\} > 0$.

et $\xi f := (\lambda \omega, arr_seq \omega == \xi) : pred (\Omega_of S).$ et $\xi f' := (\lambda \omega, arr_seq (proj1 S j_rep \omega) == \xi) : pred (\Omega_of S')$

typothesis H_ineq_5_fixed :
 P<μ_of S>{[Exc ∩ ξf]} ≤ P<μ_of S'>{[Exc' ∩

Let Epart := partition_on_E µ.

et Epart' := extend_partition S Epart j_rep

/ariables (ξi : I ξpart) (ξi' : I ξpart'). <mark>łypothesis</mark> ξ_equivalence : pickle_bij ξi ξi'

nsider a pair of arbitrary pickle-equivalent elements E1 and

n this file, we prove Theorem 1 presented in the paper "What really is pWCET? A Rigorous Axiomatic Definition of pWCET" by Bozhko et al. (RTSS'23). Step-by-step Proof of Theorem 1

on defines (an upper bound on) the terr

text {Task : TaskType}
 {pWCET_pmf : ProbWCET Task}.

ontext {Job : finType}
 {job task : JobTask Job Task

For brevity, let sched denot

Variable Ω : countType. Variable μ : measure Ω. Variable 4 : JobArrivalRV Job Ω μ. Variable C : JobCostRV Job Ω μ.

st S := {| Ω_of := Ω; μ_of := μ; A_of := A; e_of := e |}.

Hypothesis H_axiomatic_pWCET : axiomatic_pWCET (µ_of S) (job_arrival := A_of S) (job_cost := e_of S)

Variable j_rep : Job. Let S' := replace_job_pET j_rep S or convenience, let tsk denote i 's task tsk := job_task j_rep.

and let µ_tsk denote the measure induced by st µ_tsk := match pWCET_pmf with | Build_ProbWCET_pWCET nonneg sum1 {| pmf := pWCET tsk; pmf_pos := nonneg tsk; pmf_sum1 := sum1 tsk |}

Let Epart := partition_on_E µ.

(ξi : I ξpart) (ξi' : I ξpar pickle_bij ξi ξi' → ξi = ξi'

So, assuming that for any two elements of partitions $\xi i : I \xi part and \xi i' : I \xi part' that are "pickle"-equivalent, it holds that <math>P\{[Exc1 \cap \xi part < \xi i]\} \Rightarrow P\{[Exc2 \cap \xi part < \xi i]\}$

typothesis H_ineq_ξ_partitioned : Ψ (ξi : I ξpart) (ξi ' : I ξpart') (ξΕQU : pickle_bij ξi ξi'), ℙeq_of S>{[Exc ∩ ξpart⊲{ξi}]} ≤ ℙeq_of S'>{[Exc' ∩ ξpart'⊲{ξi'}] ... we can show that $P<\mu_{0}f S>{[Exc1]} = P<\mu_{0}f S'>{[Exc2]}. Or, in other words, we reduced the lemma statement to the hypothesis statement.$

Step 2

Assume that we are given a list of job arriv Variable (A : Job - option instant

Let μr := restrict ($\mu_0 f$ S) ($\xi f \cap S f$) : measure ($\Omega_0 f$ S). Let $\mu r'$:= restrict ($\mu_0 f$ S') ($\xi f' \cap S f'$) : measure ($\Omega_0 f$ S Next, we define random variables, which are the same as the or the measure (μr instead of μ and $\mu r'$ instead of μ' .

Let Cr j := mkRvar µr (C_of S j). Let Cr' j := mkRvar µr' (C_of S' j).

efinition projw : \forall (ω : Ω_o of S'), (ξ f' n Sf') $\omega \rightarrow (\xi$ f n Sf) (proj1 S j_rep s

low, we want to fix the job costs of all jobs except j_rep (recall that j_rep is th

which we want to replace pET with pWCET). For this, again, we intro-nto subsets, each containing a unique job cost assignment (and the ju sit unspecified). Let CSpart := partition_on_@s (job_cost := @r) μr (rem j_rep (enum Job)). Let CSpart' := extend_partition' S j_rep (ξf n Sf) CSpart (ξf' n Sf') pro

n the next step, we introduce CSpa

(, exceeds (R A (compute_costs ω) j) r) (Sparte(Cs) | Ef a Sf]) P<μ_of S'>{[(λ ω, exceeds (ℛ A (compute_costs' ω) j) r) n CSpart'⊲{Cs'} | Ef' n Sf']}.

then we can derive the inequality also without conditioning on cost

emma transformation_is_pRT_monotone_step6 : P<µ_of S>{[λ ω, exceeds (ℛ A (compute_costs ω) j) r | ξf ∩ Sf]} ≤ P<µ_of S'>{[λ ω, exceeds (ℛ A (compute_costs' ω) j) r | ξf ∩ Sf]

as before, we now proceed to derive the premise of the prior step o this end, consider two equivalent events corresponding to the same so the two systems. riable (Cs : I CSpart) (Cs' : I CSpart') (CsEOU : pickle bij Cs Cs'

Let Cpart := partition_on_C µr (job_cost := Cr) j_rep. Let Cpart' := partition on C µr' (job cost := Cr') j rep

with a vector C. However, what do we do with the cost of the u

lote that on the LHS of the below hypothesis, we now (j) r is a boolean value (true/false) and event C subset of Ω_of S such that all job costs are fixed.

To extract a vector of job costs from the partition CSpart<(Cs), we find one ω CSpart<(Cs) and compute the vector of costs for this ω . The resulting vector agree with the jobs in the system, given CSpart<(Cs) is true.

Note that we use the construction update C j_rep c to update the cost of job j rector C with a new value c. This is a way to define the notion of "reassembling"

emma transformation_is_pRT_monotone_step7 : P⊂μ_of S>{[(λ ω, exceeds (ℛ A (compute_costs ω) j) r) n CSpart⊲{CS} | ξf n Sf]} ≤ P⊂μ_of S'>{[(λ ω, exceeds (ℛ A (compute_costs' ω) j) r) n CSpart'⊲{CS'} | ξf 'n Sf'

λωο. We can use ωο to compute C. Such a cost assignment agrees with because ωο satisfies the initial predicate. Variable ωo : Ω_of S. Hypothesis H_ωo_in_Cs : CSpart⊲{Cs} ωo. Let C := (fun j → C_of S j ωo) : Job → option work.

Similarly to Step 7, which focused on the LHS, we now repla lotice that both sides now have all costs fully fixe

typothesis H_ineq_cost_partitioned : ∑[=]_(c<-option work) P<µ_of 5>{[λ ω, [δ& exceeds (𝔅 𝐴 (update C j_rep c) j) CSpart=((cs) ω δ______

m the above hypothesis, we can obtain the premise of

Step 9

Let us define a random variable following the pWCE1 distribution. This random variable describes j_rep's cost after its pET has been replaced. For more details, see the file rt/analysis/pETs_to_pMCETs and specifically the transformation replace_job_pET. Recall that μ _tsk is the measure induced by the given pWCET Let C_pWCET := mkRvar µ_tsk Some

Finally, we can use the independence property H_cond_independent HS. At the same time, we can factorize the RHS just by construction

rition-independence property of ST. Note that the providing of Structure $\omega == c$ factors into two probabilities: CSpart-{CS} and \mathfrak{C} j_rep == c. The second the second second

$$\label{eq:product} \begin{split} \mathbf{r}_{\mathbf{c}}(\mathbf{c}) & [\texttt{fn}(\mathbf{c}')] \\ & \text{fn}(\mathbf{c}') & [\texttt{fn}(\mathbf{c}')] \\ & \text{fn}(\mathbf{c}') & [\texttt{fn}(\mathbf{c}')] \\ & \text{fn}(\mathbf{c}') & \text{fn}(\mathbf{c}') \\ & \text{fn}(\mathbf{c}') & \text{fn}(\mathbf{c}') \\ & \text{fn}(\mathbf{c}') & \text{fn}(\mathbf{c}') \\ & \text$$

 Remark HBs_factorization :
 ∑|=|_(cc-option work)

 I[exceeds (&A (update c j_rep c) j) r]
 x Peq.of Si>{[A u, CSpart'-q(Cs') = u 66 (Cpart'-q(c) u) | ξf' n Sf']}

 y [w].(cc-option work)
 x [w].

[]]_(<-option work) I[exceeds (ℛ A (update c j_rep c) j) r] ≪ (P<u>{[Cspart⊲(Cs) | ξf ∩ Sf]} × P<μ_tsk>{[e_pWCET (=) c]}). As usual, let us state a new premise.

LUSUAL defU Stame a new premime. [j=[_<<<-option vork] [i=(<<<option vork) [i=(c<<option vork) [i=(c<<ption vork) [i=(c</ption vork) om the preceding factorized inequality, we can derive the premise of the previous step.

Lemma transformation_is_pRT_monotone_step9 : ∑[w]_(c-option work) Peµ_of 5<{1 & w. [66 exceeds (# A (update c j_rep c) j) r, Cpart=(c) w 6 Cpart=(c) w 1 { fr n Sf }}

(job_arrival := A_of S') (job_cost (sched S') horizon j ω) Let Sf := part Pi : pred (Ω_{of} S). Let Sf' := ($\lambda \omega$, Sf (proj1 S j_rep ω)) : pred (Ω_{of} S'). Variable p1 : PosProb (µ_of S) (ξf n Sf). Variable p2 : PosProb (µ of S') (ξf' n Sf').

between two elements of similar types; for further detail

Now, we know that *I* we have the inequality with partitions on arrival sequences $H_{_ineq__partItioned}$, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from which it can be established. We then continue in this manner until reaching a hypothesis that is easy enough to prove without introducing new hypotheses.

Lemma transformation_is_pRT_monotone_step1 :
 P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}.

Next, consider an arbitrary job j of any task a

Now, recall that we present the proof in a top-down (C - Goal) fashion, starting with the overall theorem.


ote the event that j's response time exceeds r

: Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz r.

is file serves for the most part to relate the prob namely to establish that $\mathbb{P}<\mu_{of S} \in [Exc]$

o start the proof. First, we do a case analysis or

Exact ξ part of the sample space $\Omega_of S$ of system

partition on ξ μ.

ndexed) set of events. t of Ω where the arrival n technical detail may l

arrival sequences the ders.) paper, one can transfe . For further details se

fer.extend_partit extend_partition

nd <mark>ξpart'</mark> are so simila ξi' that are "pickle equi ry strong notion of equiv o elements of similar ty pickle_bij.)

part) (ξi' : I ξpar j ξi ξi' → ξi = ξi

present the proof in a to

or any two elements of uivalent, it holds that P-

rmation is pRT monotone step1 Exc]} ≤ P<µ_of S'>{[Exc'

if we have the inequality with partitions on arriva ioned, then we are done. But how do we prove ove the required inequality by introducing a nev ablished. We then continue in this manner until r

we replace the partition over all arrival sequenc al sequence.

e premise of our hypothesis H_ineq_ξ_partit

artition Epart of the sample space of system : fferent arrival sequences...

partition_on_ξ μ.

stem S' and denote it as ξpart'. extend_partition S ξpart j_rep.

irbitrary pickle-equivalent elements ξi and ξi : I ξpart) (ξi' : I ξpart'). equivalence : pickle_bij ξi ξi'.

ed, pickle-equivalence is quite strong and we ca correspond to the same arrival sequence ξ . Be sequences are constructed, we can extract an n.

arrival sequence Job. part_unpack_ξ : match ξi with exist ξ I

dices ξi : I ξpart and ξi' : I ξpart', we w f' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

cpart ⊲ici ili... Hypothesis H_ineq_ξ_partitioned :
 V (ξi : I ξpart) (ξi' : I ξpart') (ξEQU : pickle_bij ξi ξi'), P<μ_of S>{[Exc ∩ ξpart⊲{ξi}]} ≤ P<μ_of S'>{[Exc' ∩ ξpart'⊲{ξi'}]}.

. we can show that $\mathbb{P} < \mu_{of S} \in [Exc1] \le \mathbb{P} < \mu_{of S} \in [Exc2]$. Or, in other words, we reduced the lemma statement to the hypothesis statement.

Lemma transformation_is_pRT_monotone_step1 : $\mathbb{P}_{\mu_of S} [Exc] \le \mathbb{P}_{\mu_of S'} [Exc']$. End Step1

Step 2 Now, we know that *if* we have the inequality with partitions on arrival sequence H ineq E partitioned, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from

which it can be established. We then continue in this manner until reaching a hypothesis

AXIOMATIC pWCET IS ADEQUATE

Theorem (*paraphrased*). Consider a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the pRT of $J_{i,j}$ in the initial system and $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a simplified system obtained via pWCET F_i . Then $\mathcal{R}_{i,i} \leq \mathcal{R}_{i,i}^{\star}$.

ineq_ξ_partitioned : part) (ξi' : I ξpart') (ξEQU : pickle_b Let ξf' := (λ ω, arr_seq ω == ξ) : pred (Ω_of S). {[Exc ∩ ξpart⊲{ξi}]} ≤ P<μ_of S'>{[E λ ω, arr_seq (proj1 S j_rep ω) == ξ) : pred (Ω_of S').

During this step, we replace indices with predicates.

t $\mathbb{P}_{\mu_0f} S > \{ [Exc1] \} \le \mathbb{P}_{\mu_0f} S' > \{ [Exc:] \}$ the lemma statement to the hypothesis statement is pRT monotone step1: The prochesis H_ineq_ ξ_{fixed} : $\mathbb{P}_{\mu_0f} S > \{ [\xif] \} > 0 \rightarrow \mathbb{P}_{\mu_0f} S > \{ [Exc n \xif] \} \le \mathbb{P}_{\mu_0f} S' > \{ [Exc' n \xif'] \}.$

1. Use axiomatic pWCET to construct a "copy" of the initial Now, we can proset about Epart and Eport and use E Ef, and E interest. Notice that Solves to the first determining in systems and provide the estimated in the system is an estimated in the estimated in the system is an estimated in the estimated in the system is an estimated in the estin the estin the es

Let $\xi_{f'} := (\lambda \omega, arr_seq (proj1 S j_rep \omega) == \xi) : pred (Q_of S').$ Vet uses of the probability of the probability D and have distribution F_j :

2. In this step, we introduce a partition (that guarantees partition-independence and partition-dependence of the pathetype action of the simplified system axis matic by the transformatic of the assumption section Step3.

Consider some countable type I and define a family of predicates part : $I \rightarrow pred \Omega$. Consider an inder P1: 1 and assume that part P1 has positive probability. Note that restrict dree por a Stical V dominates the provide the positive probability. Note that the event ensures (1) the validity of the pWCET bound and (2) the partition-independence of inb 1 rep.

of job j_rep.

Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj1 simply returns the first component of a tuple).

With this, let us assume that the inequality that now includes Sf and Sf' holds:

 $\mathbb{P} < \mu_{of S} \in \{ (Exc1 \cap \xi f) \cap Sf \} \leq \mathbb{P} < \mu_{of S} > \{ (Exc2 \cap \xi f') \cap Sf' \}$

Hypothesis H ineq Ω part : \forall (I : countType) (part : I \rightarrow pred Ω) (Pi : I) (ρ : PosProb μ ($\xi f \cap$ part Pi)), let Sf := part Pi : pred (Ω_of S) in let Sf' := $(\lambda \ \omega, \ Sf \ (proj1 \ \overline{S} \ j_rep \ \omega))$: pred $(\Omega_of \ S')$ in

(∀ (x : nat), $\mathbb{F}_{\mu} \{ [odflt0 (\mathcal{C} j_{rep}) | \xi f \cap Sf] \}(x) \ge pWCET_cdf (job_task j_{rep}) x) \rightarrow \mathbb{F}_{\mu} \}$ <u>rgey Bozhko</u>, Filip Markovic, Georg von der Brüggen, and Björn Brandenburg



prosa.model Require Import processor.idea n this file, we prove Theorem 1 presented in the paper "What really is pWCET? A Rigorous Axiomatic Definition of pWCET" by Bozhko et al. (RTSS'23). Step-by-step Proof of Theorem 1

Assume horizon defines (an upper bound on) the termination time (norizon = None, the system does not necessarily terminate. Note, h case our proof assumes there to be a finite number of jobs for techni

sider any type of tasks with a notion of pWC

text {Task : TaskType}
 {pWCET_pmf : ProbWCET Task}. nd their jobs.

ontext {Job : finType}
 {job_task : JobTask Job Task}

For brevity, let sched denote

Variable Ω : countType. Variable μ : measure Ω. Variable μ : JobArrivalRV Job Ω μ Variable c : JobCostRV Job Ω μ.

st S := {| Ω_of := Ω; μ_of := μ; A_of := A; e_of := e |}.

Next, we assume that the aforementioned pWCET is an axiomatic pWCET. That is, for Hypothesis H_axiomatic_pWCET : axiomatic_pWCET (µ_of S) (job_arrival := A_of S) (job_cost := e_of S)

analysis/pETs_to_pWCETs to replace the execution cost of given job j_rep. Let S anote the resulting system. Variable j_rep : Job. Let S' := replace_job_pET j_rep S. For convenience, let tsk denote i's task.

tsk := job_task j_rep. and let µ_tsk denote the measure induced by pw

st µ_tsk := match pWCET_pmf with | Build_ProbWCET pWCET nonneg sum1 -{| pmf := pWCET tsk; pmf_pos := nonneg tsk; pmf_sum1 := sum1 tsk |}

Next, consider an arbitrary job j of any task and

(job_arrival := A_of S') (job_cost (sched S') horizon j ω)

Let Epart := partition_on_E µ.

As discussed in the paper, one can transfer the partition fenote it as Epart'. For further details see the function

between two elements of similar types; for further detail

(ξi : I ξpart) (ξi' : I ξpar pickle_bij ξi ξi' - ξi = ξi'

Now, recall that we present the proof in a top-down (C - Goal) fashion, starting with the overall theorem. So, assuming that for any two elements of partitions $\xi i : I \xi part and \xi i' : I \xi part' that are "pickle"-equivalent, it holds that <math>P\{[Exc1 \cap \xi part < \xi i]\} \Rightarrow P\{[Exc2 \cap \xi part < \xi i]\}$

typothesis H_ineq_E_partitioned :
 Y (ξi : I Epart) (ξi' : I Epart) (ξEQU : pickle_bij ξi ξi'),
 Peq_of S>([Exc n ξpart⊲(ξi)]) ≤ Peq_of S'>([Exc' n ξpart⊲(ξi') ... we can show that $P<\mu_0f S>{[Exc1]} = P<\mu_0f S'>{[Exc2]}. Or, in other words, we reduced the lemma statement to the hypothesis statement.$ Lemma transformation_is_pRT_monotone_step1 : P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}.

Step 2 Now, we know that *I* we have the inequality with partitions on arrival sequences $H_{_ineq__partItioned}$, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from which it can be established. We then continue in this manner until reaching a hypothesis that is easy enough to prove without introducing new hypotheses. Step 2

ain, consider a partition Epart of the sample space of system S into events Let Epart := partition_on_E µ.

et Epart' := extend_partition S Epart j_rep

onsider a pair of arbitrary pickle-equivalent elements E1 and Variables (ξi : I ξpart) (ξi' : I ξpart'). Hypothesis ξ_equivalence : pickle_bij ξi ξi'

Variable ξ : arrival_sequence Job. Hypothesis H part uppack E : match E with exist E IN \Rightarrow E end =

Let $\xi f := (\lambda \omega, arr_seq \omega == \xi) : pred (\Omega_of S).$ Let $\xi f' := (\lambda \omega, arr_seq (proj1 S j_rep \omega) == \xi) : pred (\Omega_of S')$

typothesis H_ineq_5_fixed :
 P<μ_of S>{[Exc ∩ ξf]} ≤ P<μ_of S'>{[Exc' ∩ Lemma transformation_is_pRT_monotone_step2 : P<µ_of S>{[Exc n Epart⊲{Ei}]} ≤ P<µ_of S'>{[Exc' n Epart'⊲{Ei'}]}

Now, we can forget about $\xi part$ and $\xi part'$ and use $\xi, \xi f, ar here we use <math display="inline">\xi$ to denote the deterministic arrival sequence. Pre denote events that result in ξ in systems 5 and 5', respectively

et ξ f := (λ ω , arr_seq ω == ξ) : pred (Ω _of S). et ξ f' := (λ ω , arr_seq (proj1 S j_rep ω) == ξ) : pred (Ω _of S') othesis H_ξ_pos_prob : P<μ>{[ξf]} > 0.

onsider an index Pi : i and assume that part Pi has positive probability. Note that prt might depend on $\xi f.$

lext, let us assume the following

et us define Sf's twin S Jith this, let us assume that the inequality that now includes Sf and Sf

V (x : nat), Ecus(odflt0 (@ i rep) | Ef p Sf l}(x) > pWCET cdf (ich) / (C : Job → option work) (jobs : seq Job), j_rep \notin jobs → PeµP{[e_fix C :: j_rep] ∩ e_fix C jobs | ξf ∩ Sf]} = PeµP{[e_fix C ::: j_rep] | ξf ∩ Sf]} × PeµP{[e_fix C jobs

 $P<\mu_of S>{[(Exc n \xif) n Sf]} \le P<\mu_of S'>{[(Exc' n \xif') n Sf']}$ pothesis. In particular, we have to present a

Lemma transformation_is_pRT_monotone_step3 : P<µ_of S>{[Exc n Ef]} ≤ P<µ_of S'>{[Exc' n Ef']}.

countType) (part : Idx → pred Ω) (Pi : Idx Let Sf := part Pi : pred (Ω_{of} S). Let Sf' := ($\lambda \omega$, Sf (proj1 S j_rep ω)) : pred (Ω_{of} S'). Variable p1 : PosProb (µ_of S) (ξf n Sf). Variable p2 : PosProb (µ of S') (ξf' n Sf').

_independence : ption work) (jobs : seq Job),

j_rep \notin jobs → P<µ>{[e_fix c [:: j_rep] ∩ e_fix c jobs | ξf ∩ Sf]} = P<µ>{[e_fix c [:: j_rep] | ξf ∩ Sf]} × P<µ>{[e_fix In this step, we transform our inequality by moving $\xi f \cap S f$ and $\xi f' \cap S f'$ to the

Hypothesis H_ineq_conditional : P<μ_of S>{[Exc | ξf ∩ Sf]} ≤ P<μ_of S'>{[Exc' | ξf' ∩ Sf']} Lemma transformation_is_pRT_monotone_step4 : P<U_of S>{[Exc n ξf n Sf]} ≤ P<U_of S'>{[Exc' n ξf' n Sf']}.

example, given $\omega \setminus \text{in } \Omega_of S, \text{ compute_costs}$ returns a vector of all costs fixed for this specific evolution $\omega.$ Let compute_costs (ω : Ω_{of} S) := job.compute_costs ω (job_cost := e_{of} S). Let compute_costs' (ω : Ω_{of} S') := job.compute_costs ω (job_cost := e_{of} S').

For simplicity, let \mathscr{R} denote a function that computes the response time of any job for given fixed vectors A and C. Let \mathcal{R} := schedulerAC_to_rtAC horizon ζ .

In this step, we replace events Exc and Exc ⁱ with events $\lambda \omega = exceeds$ (*S* (compute_costs ω) j) r, where the arrival times are fixed to be a *specific* response-time distribution, but now we have algorithm *s* instead. ection Step5.

Note that here we assume that we are given any vector A describing job arrivals withour restriction that it must agree with ξ . Inside of the proof, we indeed construct A as a transformation of ξ ; however, for further proofs, it is not relevant, so we just forget this information and use a generic function Job – option instant.

ypothesis H_ineq_algorithmic_% : ¥ (A : Job → option instant), P=u_of S// A w, exceeds (% A (compute_costs w) j) r | ξf n Sf]} ≤ P=u_of S'>{[λ w, exceeds (% A (compute_costs' w) j) r | ξf' n Sf' uality involving Exc and Exc' is implied by the inequality involving a

Assume that we are given a list of job arriva Variable (A : Job - option instant)

Let μr := restrict ($\mu_0 f$ S) ($\xi f \cap S f$) : measure ($\Omega_0 f$ S). Let $\mu r'$:= restrict ($\mu_0 f$ S') ($\xi f' \cap S f'$) : measure ($\Omega \circ f$ S' asure (µr instead of µ and µr' instead of

Let Cr j := mkRvar µr (C_of S j). Let Cr' j := mkRvar µr' (C_of S' j).

Definition projw : $\forall (\omega : \Omega_of S'), (\xi f' \cap S f') \omega \rightarrow (\xi f \cap S f) (proj1 S j_rep \omega$

low, we want to fix the job costs of all jobs except j_rep (recall that j_ Let CSpart := partition_on_@s (job_cost := @r) μr (rem j_rep (enum Job)). Let CSpart' := extend_partition' S j_rep (ξf n Sf) CSpart (ξf' n Sf') proj

n the next step, we introduce CSpart<{Cs} and CSpart'<{C

w, exceeds (R A (compute_costs ω) j) r) CSpart⊲{Cs} | ξf n Sf]} P<μ_of S'>{[(λ ω, exceeds (ℛ A (compute_costs' ω) j) r) n CSpart'⊲{Cs'} [ξf' n Sf']}.

then we can derive the inequality also without conditioning on costs emma transformation_is_pRT_monotone_step6 : P<µ_of S>{[λ ω, exceeds (ℛ A (compute_costs ω) j) r | ξf ∩ Sf]} ≤ P<µ_of S'>{[λ ω, exceeds (ℛ A (compute_costs' ω) j) r | ξf' ∩ Sf']

As before, we now proceed to derive the premise of the prior step To this end, consider two equivalent events corresponding to the same setting of job c in the two systems. ariable (Cs : I CSpart) (Cs' : I CSpart') (CsEOU : pickle bij Cs Cs')

Let Cpart := partition_on_@ µr (job_cost := @r) j_rep. Let Cpart' := partition on @ µr' (job cost := @r') j rep milarly to how we replaced job arrivals with A, we can do a similar tr sts with a vector C. However, what do we do with the cost of the un

lote that on the LHS of the below hypothesis, we now have the foll An important result of this transformation is that now exc (j) r is a boolean value (true/false) and event C subset of Ω_of S such that all job costs are fixed.

To extract a vector of job costs from the partition CSpart<[Cs], we find one a 6 CSpart<[Cs] and compute the vector of costs for this ω . The resulting vector of agree with the jobs in the system, given CSpart<[Cs] is true. Note that we use the construction update C j_rep c to update the cost of job j_rep rector C with a new value c. This is a way to define the notion of "reassembling" two j

ypothesis H_ineq_cost_partitioned : ¥ (wo : Ω of S), CSpart-{CS} wo → let c := (fun j → e_of S j wo) in

emma transformation_is_pRT_monotone_step7 : P≈q_of S>{[(λ ω, exceeds (ℛ A (compute_costs ω) j) r) n CSpart⊲{Cs} ≈ P≈q_of S`>{[(λ ω, exceeds (ℛ A (compute_costs' ω) j) r) n CSpart'

wo. We can use wo to compute C. Such a cost assignment agrees with because wo satisfies the initial predicate. Variable wo : $Q_of S$. Hypothesis H_wo_in_Cs : CSpart<{Cs} wo. Let C := (fun j + $e_of S j wo$) : Job + option work.

Similarly to Step 7, which focused on the LHS, we now replace lotice that both sides now have all costs fully fixe

CSpart=(Cs) ω δ [mart=(c) ω] { fn Sf } ≤ [[u]_{cc-option work] Pcu of St=(1 λ ω, [66 exceeds (# A (update C j_rep c) j) CSpart*=(Cs)* J ω Cpart*=(c)* J ω

m the above hypothesis, we can obtain the premise of § Lemma transformation_is_pRT_monotone_step8 : [1]_c-option work) PhyLode exceeds (#A (update C j_rep c) j) r. CSpart=(CS) u & Prop of Supric(C) u) [cf n Sf]) s Ptop of Supric(C) u) [cnopute_costs' u)]) n (Spart=(CS) [r n Sf']).

Step 9

e a random variable tollowing the pwcE1 distribution. This random variable _rep's cost after its pET has been replaced. For more details, see the file is/pETs_to_pWCETs and specifically the transformation ob_pET. Recall that µ_tsk is the measure induced by the given pWCET Let C_pWCET := mkRvar µ_tsk Some

Finally, we can use the independence property H_cond_independen LHS. At the same time, we can factorize the RHS just by construction.

rep w == c factors into two pro ilities: CSpart<{Cs} and & j_rep ==

× P<µ_of S>{[ℓ j_rep (= Next, we show that the probability part of the RHS can be factorized into two terms: $P_{SUS}\{(S_{Part} \in (S_{S}) | Ef \cap Sf\}\}$ and $P_{SU} = f_{SUS}\{(f_{UD} \cap (S_{S}) | Ef \cap Sf)\}$

the line is a ratio is to only a probability min respect to 3 - it is a probability with respect to 5. For the second term, it is a probability over the probability space described in the file pETs_to_pWCETs. Remark HRs_factorization : ∑i=__(cc-option work) Ilexceds (&A (update c j_rep c) j) r] × Peq_of S'>{[A w__CSpart'⊲(Cs'} w && (Cpart'⊲(c) w) | ξf' n Sf']} = Vie_1(cc-ontion work)

 $\begin{array}{l} \underbrace{[e]_{(c-option work)}}_{[e]exceeds (R A (update c _rep c) j) r]} \\ \underbrace{[e]_{(c-option d)}}_{(c-option d)} \underbrace{[e]_{(c-option d)}}_{[c-option d)} \underbrace{[e]_{($ As usual, let us state a new premise.

Usuam, et us sale a new permise. [[e]_(c<-option vork) liexceds (x A (update c]_rep c))) r] × [Po_u of S<[[Spart-c(SS] [ξ f n Sf] } × Poµ_of S>[[¢]_rep [=) c] * [Po_u of S>[[Spart-c(SS] [ξ f n Sf]] × Poµ_tsk>[[¢]_NCET (=) c] * (Poµ_tof S<[[Spart-c(SS] [ξ f n Sf]] × Poµ_tsk>[[¢]_NCET (=) c] * (Poµ_tof S<[[Spart-c(SS] [ξ f n Sf]] × Poµ_tsk>[[¢]_NCET (=) c]

m the preceding factorized inequality, we can derive the premise of the previous step

<pre>I[exceeds (% A (update c j_rep c) j) r]</pre>	
From the preceding factorized inequality, we can derive the premise of the previous step.	
[im](c<-option work) Pe⊥of S>{{} X w, [66 exceeds (3 A (update c j_rep c) j) r,	
CSpart-d(CS) ⊌ & 6 Cpart-d(C) ຟ ξf ∩ Sf]} ≤ ∑[=]_(c<-option work)	
[≤4_0] S ≥4(1 ∧ W; [56 exceeds (# A (update C j_rep c) j) r, (Spart'd(S') w δ (Cpart'd(C') w)] [f' ∩ Sf']}.	
End Step9.	
Step 10	
In this step, we remove P<µ_of S>{[CSpart⊲{Cs} ξf ∩ Sf]} on the LHS and P<µ_of S>{[CSpart⊲{Cs} ξf ∩ Sf']} on the RHS.	
Section Step10.	
Let us assume that we can prove the following inequality, where $P < \mu_0 f S < [CSpart < CS + [f \cap Sf]] on both sides cancel out,$	
Hypothesis H_ineg_without_other_costs : ∑[@]_(c<-option work) I[exceeds (% A (update C j_rep c) j) r] × P<µ_of S>{[€ j_rep (=) c ξf n Sf]}	
≤ ∑[e]_(c<-option work) I[exceeds (ℛ A (update C j_rep c) j) r] × P<µ_tsk>{[e_pWCET (=) c]}.	
then we can easily arrive at the preceding step's premise. Lemma_transformation_is_pRT_monotone_step10:	
2[#]_(C==option work/ [[exceeds (& A (update c j_rep c) j) r] × (P=μ_of S>{[Cspart={Cs}] ξf n Sf]} × P=μ_of S>{[c j_rep (=) c ξf n Sf]}) < Σ[#] (c==notion work)	
<pre>[] Sigesteed (R A (update c j_rep c) j) r]</pre>	
End Step10.	
Step 11	
We have successfully removed constraints on job costs of jobs that are not equal to j_rep. Depending on the cost of j_rep, we can still get different behavior. In this section, we do a case analysis on the "critical" value of the cost of j rep.	
Section Step11.	
There are three possibilities: (1) whatever cost of [ob]_rep we pick, the response time of job] aver exceeds r, (2) whatever cost of [ob]_rep we pick, the response time of job] always exceeds r, and (3) there is some critical value c6 such that]s response time will exceed r if and only if]_rep is job cost is larger than r.	
Lemma cost_causing_exceedance_of_r: (* 3 *) [3] (c0 : option work), ∀ (c : option work), is_true (c0 (<) c) * exceeds (X A (update C]_rep c) j) r)	
<pre>v (= 2 **) (¥ (c : option work), exceeds (% A (update C j_rep c) j) r) v (= 1 **) (¥ (c : option work), ¬ exceeds (% A (update C j_rep c) j) r). Note that case (1) is easy since the inequality reduces to 0 = 0.</pre>	
Remark jobs_rt_never_exceeds_r : (Y c: option work, - exceeds (R A (update c j_rep c) j) r) -)[a] (c=option work)	
[[exceeds (X A (update c j_rep c) j) r] × Peup-[[e]_rep (=) c [ξ f ∩ Sf]} ≤ [[=]_(c-option work)	
<pre>I[exceeds (% A (update c j_rep c) j) r]</pre>	
Note that case (2) is easy since the inequality reduces to 1 ≤ 1. <u>Remark</u> jobs_rt_always_exceeds_r :	
(∀ c : option work, exceeds (ℛ A (update C j_rep c) j) r) → ∑[∞]_{c<-option work} I[exceeds (ℛ A (update C j_rep c) j) r]	
x P=QP=(1 c)_rep (=) c { c { c } r n 5 } } \$ [m]_(c=option work} I[exceeds (# A (update c j_rep c) j) r] x P=u tskof{ c wotEr (=) c }	
Case (3) is interesting; so let us assume that c0 is the critical value, after which the response time of job j will exceed r.	
However, then I [exceeds (\Re A (update C j_rep c) j) r] can be transformed into c0 (<) c. We use the fancy (<) to account for the fact that both costs can be 1.	
Hypothesis H_ineq_c0_causes_exceedance : V (c0 : option work),	
$\begin{split} & T[\theta_{-1}(c-\text{option work}) \ T(\theta_{-1}(c) \ \times \ \theta_{-n} \ \text{of} \ S^{-1}(e_{-1}, e_{-n} \ e_{-1}) \ c_{-1}(e_{-1}, e_{-1} \ e_{-1}) \ c_{-1}(e_{-1}, e_{-1}) \ c_{-1}(e_{$	
Lemma transformation_is_pRT_monotone_stepi1 : ∑[e]_(cc-option work} I[exceeds (& A (undate c i rep.c) i) r] × Psu of S >{[c i rep.(=) c Ef.o.Sf]}	
<pre>\$[w]_[co-option work} [[exceeds (x A (update C j_rep c) j) r] × P<µ_tsk>{[e_pwCET (=) c]}.</pre>	
End Step11.	
Step 12	
Let us now assume that there is a cost c0 (of job j_rep) with the property that any cost strictly higher causes j to have a response time larger than r.	
Variable c0 : option work. Hypothesis H_c0_causes_exceedance : Y c, is_true (c0 <(: 0) = exceeds (X A (update C j_rep c) j) r.	
Variable c0 : option work. Hypothesis H_c0_cause_ccceedance : $\forall c \cdot j_{strue} (c0 < c) = ccceedas ($	
Yoriable c0 : option work. Hypothesis H_C6_causes_recedence : v c, is_true (c0 (c) c) ← exceeds (X A (update c j_rep c) j) r. Note that <u>\</u> _{c} [1[c0 c c] P[X = c] is equivalent to P[X > c0]. We use this property to simplif \[] options uses. Section Step12.	
Variable c0 : option work. Vporthasis N_(c0 causes_coccedance : V c, is_true (c0 (c) c) = exceeds (% A (update c j_rep c) j) r. Note that J_(c) I(c0 < c) P(X = c) is equivalent to P(X > c0). We use this property to simplify I() to both sides. Section Step12. Assuming that the following nequality, which is very similar to our initial assumption H_DVCT_bounds_cond_coff	
Variable c8 : option work. Hypothesis (U = causes) Y <, is_true (e causes)	
<pre>Variable c0 : option work. hypothesis IL (c0 causes_coccedance :</pre>	
Yoriable c d : option work. Wy c, is_true (c0 (c) c) = exceeds (x A (update c j_rep () j) r. Note that <u>J_(c) T[c0 < c] P(X = c) is equivalent to P(X > c0). We use this property to</u> simplify T[] to note sides. Section Step12. Assuming that the following inequality, which is very similar to our initial assumption H_pWCET_Dounds_cond_cdf Hypothesis H a linost_wCET_bounds_cond_cdf : Peq_LSA*[c j_rep (-c) c 0] {f n Sf } } we can prove the inequality assumed in the previous step. Lema transformation_is_pRT_monotone_step12 : [Lema transformation_is_pRT_monotone_step12]: [Lema transformation_is_pRT_monotone_step12]:	
<pre>Variable c0 : option work. Wpothesis (Log causes_corecaince :</pre>	
<pre>Variable c 0 : option work. You have t c 1; t (c) (c) c auses_coccedance : Y c, is_true (c0 (c) c) = exceeds (% A (update c j_rep c) j) r. Note that J_(c) I(c0 < c) P(X = c) is equivalent to P(X > c0). We use this property to simply I() to obth sides. Section Step12. Assuming that the following inequality, which is very similar to our initial assumption Hypetter_bounds_cond_cdf Hypetter_bounds_cond_cdf : P=u_ttsk-[[c j_xrep (c) c0] [f n Sf]) * P=u_ttsk-[[c j_xrep (c) c0] [f n Sf]) * = [[[(c - coption nat) I [(c) (c) (c)] * [c j_xrep (c) c] [f n Sf]) * = [[[(c - coption nat) I [(c) (c) (c)] * [c j_xrep (c) c] [f n Sf]) * = [[[(c - coption nat) I [(c) (c) (c)] * [c j_xrep (c) c] [f n Sf]) * = [[[(c - coption nat) I [(c) (c) (c)] * [c j_xrep (c) c] [f n Sf]) * = [[[(c - coption nat) I [(c) (c) (c)] * [c j_xrep (c) c] [f n Sf]) * = [[[(c - coption nat) I [(c) (c)] * [c j_xrep (c) c] [f n Sf]) * = [[[(c - coption nat) I [(c) (c)] * [c j_xrep (c) c] [f n Sf]] * = [[[(c - coption nat) I [(c) (c)] * [c j_xrep (c) c] [f n Sf]] * = [[[(c - coption nat) I [(c) (c)] * [c j_xrep (c) c] [f n Sf]] * = [[[(c - coption nat) I [(c) (c)] * [c j_xrep (c) c] [[(c - c) f n Sf]] * = [[[(c - coption nat) I [(c) (c)] * [c j_xrep (c) c]] * [[(c - c) f n Sf]] * = [[[(c - coption nat) I [(c - c)] * [(c - c) f n Sf]] * = [[[(c - coption nat) I [(c - c)] * [(c - c) f n Sf]] * = [[[(c - coption nat) I [(c - c)] * [(c - c) f n Sf]] * = [[[(c - coption nat) I [(c - c)] * [(</pre>	
<pre>Variable c 0 : option work. Wyothesis (v = cause, coccedance :</pre>	
<pre>Variable cd : option work. by c, is_true (c0 (c) c) = exceeds (% A (update c j_rep c) j) r. Note that <u>J_(c) I(c0 < c) P(X = c) is equivalent to P(X > c0).</u> We use this property to simplify I() no tools iddes. Section Step12. Assuming that the following inequality, which is very similar to our initial assumption <u>H_pWETE_Dounds_cond_cdf</u> Hypothesis H almost_wETE_Dounds_cond_cdf :</pre>	
<pre>Variable cd : option work. yc, is_true {cd : {c} - c} = exceeds (# A (update c j_rep c) ;) r. Note that J_{c} [C] [cd < c] P(X = c) is equivalent to P(X > cd). We use this property to simplify II) no toch sides. Section Step12. Assuming that the following nequality, which is very similar to our initial assumption H_pWCET_bounds_cond_cdf, Hypothesis H_almost_pWCET_bounds_cond_cdf :</pre>	
<pre>Variable c 0 : option work. You has t = c 0 : option work. Y c, is_true (c0 (c) c) = exceeds (& A (update c j_rep c) j) r. Note that J_(c) I(c0 < c) P(X = c) is equivalent to P(X > c0). We use this property to simply I() to obto sides. Section Step12. Assuming that the following inequality, which is very similar to our initial assumption H_DMPETE_bounds_cond_cdf. PMUETE_bounds_cond_cdf. PMUETE_bounds_cond_cdf. Leans transformation is_pMT_monotone.step12 : [[c] _rco) (c) [c] rcf is f is f is f is f is f is cond t = co</pre>	
<pre>Variable c 0 : option work. You have t c 0 = customer is the second of the secon</pre>	
<pre>Variable c 0 : option work. You have that \$\{c\} if (0 < c\} = exceeds (& A (update c \$_rep c) \$) r. Note that \$_{c}^{c} If (0 < c\} P(X = c) is equivalent to P(X > c\). We use this property to simply If 10 noto hidde. Section Step12. Assuming that the following inequality, which is very similar to our initial assumption HypetCT_bounds_cond_cdf. HypetCT (0 = rep (0 = c) 0 { { f n Sf } } > F=q_Ltsk-{[c \$_ymet[(c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[(c = rep (c = c) 0] { f n Sf }] > End Step12. Step13 In the last step, we exploit the top-level assumption H_pWCET_bounds_cond_cdf to final the proof. Section Step13. Notice that the following statement is very close to the pWCET guarantee H_pWCET_bounds_cond_cdf. Lemme transformation_is_RT_monotone_step13 : F=q_Ltsk-{[c \$_ymet[(c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[(c = rep (c = c) 0] { f n Sf }] > Section Step13. Notice that the following statement is very close to the pWCET guarantee H_pWCET_bounds_cond_cdf. Lemme transformation_is_RT_monotone_step13 : F=q_Ltsk-{[c \$_ymet[(c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c = rep (c = c) 0] { f n Sf }] > F=q_Ltsk-{[c \$_ymet[c =</pre>	
<pre>Variable cd : gptim work.</pre>	
<pre>Variable of : option work.</pre>	
<pre>Norise is option work:</pre>	
<pre>Variable of : gptim work.</pre>	
<pre>Variable 0 : option work. You has 1_{0} cause_coredance : Y c, is_true (0 (c) c) = exceeds (& A (update crep c)]) r. Note that T_{0} core coredance : Y c, is_true (0 (c) c) = exceeds (& A (update crep c)]) r. Note that T_{0} core core core core core core core core</pre>	
<pre>Variable cd : gption work.</pre>	
<pre>WrateLie G : gptim work:</pre>	
<pre>Number of the spring work.</pre>	
<pre>Note that is cf : gritan work. Ye, is_true (cf (c) c) = exceeds (& A (update c j_rep (c) j) r. Note that J_(c) I(cd < c) P(X = c) is equivalent to P(X > ch). We use this property to simply I() no to bid side. Section Step12. Assuming that the following inequality, which is very similar to our initial assumption H_PMCET_bounds_cond_cf. HypetTel_bounds_cond_cf. HypetTel_bounds_cond_cf. HypetTel_bounds_cond_cf. HypetTel_bounds_cond_cf. HypetTel_bounds_cond_cf. HypetTel_bounds_cond_cf. HypetTel_bounds_cond_cf. HypetTel_bounds_cond_cf. End Step12. Step13 In the last step, we exploit the top-level assumption H_PWCET_bounds_cond_cff to finish the proof. Section Step13. Notice that the following statement is very close to the pWCET guarantee H_PWCET_bounds_cond_cf. Lemma transformation_is_PTC if (f) f f) f f f f f f f f f f f f f f</pre>	
<pre>Nyministics 0: sprime work:</pre>	
<pre>Number of the spring work. Ye, is_true (dd (d) c) = exceeds (& A (update c j_rep c) j) r. Note that J_{c} I [dd < c] P X = c is equivalent to P(X > ch). We use this property to simply I [d, i, j) noto hiddes. Section Step1. Assuming that the following inequality, which is very similar to our initial assumption P Lot I = 0 noto hiddes. Section Step1. Assuming that the following inequality, which is very similar to our initial assumption P Lot I = 0 not in dise. Section Step1. Assuming that the following inequality assumed in the previous step. I = 0 the p Lot I = 0 the spring in the spring</pre>	
<pre>Write: 0 : grite unver.</pre>	
<pre>Write is if i grite unvert.</pre>	
<pre>Number of the set of the set</pre>	
<pre>Write to first provide the first provide th</pre>	
<pre>Variable : 0 : gaption work:</pre>	
<pre>Variable 0; ; grigins wet. Vie. 1s_true (cl (c) = created set (x A (update c)_rep()) r. Note that 1; (c) I(cl < cl P(x = c) is equivalent to P(x > cl). We use this property to settion Step1. Assuming that the following inequality, which is very similar to our initial assumption if (, in our both sides. Settion Step1. Assuming that the following inequality, summed in the previous step. If (, if (</pre>	
<pre>Variable 0 = regions white</pre>	
<pre>Yuring the set of the set of</pre>	
<pre>Yuribule up to prive work "Y & f a true (d (d) = exceeds (f A (update c j rep c)) r. Nee hat j (c) [d (d) = exceeds (f A (update c j rep c)) r. Nee hat j (c) [d (d) = exceeds (f A (update c j rep c)) r. Nee hat j (c) [d (d) = rep c = b a equivalent to P(X > cb). We use this property to singuly 1 [] on both adds. Section Step1: . we can prove the inequality assumed in the previous step. [m j = f (d) = f (d) = f (f) f)</pre>	
<pre>Yurbit bit if y contains the function of the section is the section in the section is the s</pre>	
<pre>vyrigthers::::::::::::::::::::::::::::::::::::</pre>	
<pre>Virgin: prive if y is prive work if y is prive work if y is prive i</pre>	



ote the event that j's response time exceeds r

: Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz r.

is file serves for the most part to relate the prob namely to establish that $\mathbb{P}<\mu_{of S}<\{[Exc]\}$

o start the proof. First, we do a case analysis or

 ξ part of the sample space Ω of S of system fferent arrival sequences.

partition on ξ μ.

ndexed) set of events, where an ξ_i -th event de t of Ω where the arrival sequence is equal to ξ_1 . on technical detail may be interested in noting th

arrival sequences themselves. This detail may ders.)

paper, one can transfer the partition ξ part to t . For further details see the function fer.extend_partition.

extend_partition S ξpart j_rep.

nd Epart' are so similar that one can prove equ ξ_i that are "pickle equivalent" can be shown t ry strong notion of equivalence; one can intuitiv o elements of similar types; for further details s pickle bij.)

part) (ξi' : I ξpart'), j ξi ξi' → ξi = ξi'.

present the proof in a top-down ($C \rightarrow Goal$) fash

or any two elements of partitions $\xi i : I \xi part$ uivalent, it holds that P{[Exc1 n Epart 4{Ei}]

ineq_&_partitioned :
part) (& i' : I & part') (& EQU : pickle_b {[Exc ∩ ξpart⊲{ξi}]} ≤ ℙ<μ_of S'>{[E

 $P < \mu_of S > \{ [Exc1] \} \le P < \mu_of S' > \{ [Exc2] \}$ the lemma statement to the hypothesis stateme rmation_is_pRT_monotone_step1 :

Exc]} ≤ P<µ_of S'>{[Exc'

if we have the inequality with partitions on arriva ioned, then we are done. But how do we prove ove the required inequality by introducing a nev blished. We then continue in this manner until r to prove without introducing new hypotheses.

we replace the partition over all arrival sequence al sequence.

stem S' and denote it as spart'.

extend_partition S ξpart _rep

Without loss of g e premise of ar has the sist in the provide (otherwise the LI Hypothesis H artition Epart of the sample space of system fferent arrival sequences...

In this step, we dominance) of t of axiomatic pW Section Step

Consider some rbitrary pickle-equivalent elements ξi and ξi' a might deper : I ξpart) (ξi' : I ξpart'). equivalence : pickle_bij ξi ξi'.

ed, pickle-equivalence is quite pong a ob je_rep. A t us define S correspond to the same arriva sequences are constructed, w t component arrival_sequence Job.

part_unpack_ξ : match ξi with exist ξ I dices ξi : Ιξpart and ξi' : Ιξpart', we w f ' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

cpart ⊲ici ili... P<μ_of S>{[Exc n ξpart⊲{ξi}]} ≤ P<μ_of S'>{[Exc' n ξpart'⊲{ξi'}]}.

.. we can show that $\mathbb{P} < \mu_{of} S > \{ [Exc1] \} \leq \mathbb{P} < \mu_{of} S' > \{ [Exc2] \}$. Or, in other words, we reduced the lemma statement to the hypothesis statement.

Lemma transformation_is_pRT_monotone_step1 : $\mathbb{P}<\mu_{of S} = \mathbb{P}<\mu_{of S} = \mathbb{P}$

End Step1

Step 2

Now, we know that if we have the inequality with partitions on arrival sequences H ineg E partitioned, then we are done. But how do we prove such an inequality? In the next step, we prove the required inequality by introducing a new hypothesis from which it can be established. We then continue in this manner until reaching a hypothesis that is easy enough to prove without introducing new hypotheses

In the second step, we replace the partition over all arrival sequences with an event encoding one arrival sequence.

Section Step2

First, let us state the premise of our hypothesis $H_{ineq_{partitioned}}$

Again, consider a partition Spart of the sample space of system S into events corresponding to different arrival sequences...

Let Epart := partition on E u.

... and *Epart* to system S' and denote it as *Epart*'.

Let ξpart' := extend_partition S ξpart j_rep.

Consider a pair of arbitrary pickle-equivalent elements ξi and $\xi i'$.

Variables (ξi : I ξpart) (ξi' : I ξpart') Hypothesis ξ_equivalence : pickle_bij ξi ξi'.

As already discussed, pickle-equivalence is guite strong and we can show that both indices ξi and $\xi i'$ correspond to the same arrival sequence ξ . Because of the way partitions on arrival sequences are constructed, we can extract an arrival sequence by unpacking a partition

Variable E : arrival sequence Job. Hypothesis H_part_unpack_ ξ : match ξ i with exist ξ IN $\rightarrow \xi$ end = ξ .

Instead of clunky indices Ei : I Epart and Ei' : I Epart', we will use "plain" predicates ξf and $\xi f'$ that correspond to a fixed arrival sequence ξ in S and S', respectively

Let $\xi f := (\lambda \omega, arr_seq \omega == \xi)$: pred ($\Omega_of S$). Let $\xi f' := (\lambda \omega, arr_seq (proj1 S j_rep \omega) == \xi) : pred (\Omega_of S').$

During this step, we replace indices with predicates.

Hypothesis H_ineq_ξ_fixed : P<µ_of S>{[ξf]} > 0 → P<µ_of S>{[Exc ∩ ξf]} ≤ P<µ_of S'>{[Exc' ∩ ξf']}.

Lemma transformation_is_pRT_monotone_step2 : $\mathbb{P}_{\mu_of S}[\text{Exc } \cap \overline{\xi}_{\mu_v}] \leq \mathbb{P}_{\mu_of S'}[\text{Exc' } \cap \overline{\xi}_{\nu_v}].$

End Step2.

Step 3

Now, we can forget about ξ part and ξ part ' and use ξ , ξ f, and ξ f' instead. Notice that here we use ξ to denote the deterministic arrival sequence. Predicates ξf and ξf denote events that result in ξ in systems S and S', respectively.

Variable ξ : arrival_sequence Job. Let $\xi f := (\lambda \omega, \operatorname{arr} \operatorname{seq} \omega == \xi) : \operatorname{pred} (\Omega_o f S).$ Let $\xi f' := (\lambda \omega, \operatorname{arr} \operatorname{seq} (\operatorname{proil} S \ i \ \operatorname{rep} \omega) == \xi)$

If pWCET satisfies our notion of axiomatic pWCET, ...

Hypothesis H_axiomatic_pWCET : axiomatic_pWCET (μ_{of} S) (job_arrival := \mathcal{A}_{of} S) (job_cost := \mathcal{C}_{of} S).

... then the response-time distribution of job j in schedule sched S is \leq -bounded by the response-time distribution of job j in schedule sched S'. That is, $\Re j \leq \Re j'$.

Lemma prob_rt_monotonic_axiomatic_pWCET_replace_pET : Rj ≤ Rj'.

P<µ_of S>{[(Exc1 n ξf) n Sf]}≤P<µ_of S'>{[(Exc2 n ξf') n Sf']

Hypothesis H_ineq_Ω_part : ∀ (I : countType) (part : I → pred Ω) (Pi : I) (ρ : PosProb μ (ξf ∩ part Pi)), let Sf := part Pi : pred (Ω of S) in let Sf' := ($\lambda \omega$, Sf (proj1 S j_rep ω)) : pred (Ω of S') in

(∀ (x : nat) $\mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 ($

AXIOMATIC pWCET IS ADEQUATE

a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a $\mathsf{ET} F_i \mathsf{Then} \, \mathscr{R}_{i,i} \preceq \mathscr{R}_{i,i}^{\star}$

truct a "copy" of the initial ad with job casts that are

<u>rgey Bozhko</u>, Filip Markovic, Georg von der Brüggen, and Björn Brandenburg



m prosa.model Require Import processor.ideal. n probsa.util Require Export nic biop.inf. n probsa.probability Require Export pred law.of_total_prob. probsa.rt.aodel Require Export task vents ariomatic_DMCE probsa.rt.analysis Require Export pETs_to_DMCETs partition

n this file, we prove Theorem 1 presented in the paper "What really is pWCET? A Rigorous Axiomatic Definition of pWCET" by Bozhko et al. (RTSS'23). Step-by-step Proof of Theorem 1

e following, we present the proof of Theorem 1 in the above-cited h of the proof and the nature of Coq, we cannot start this section theorem. Instead, we will first prove many "stepping stone" lemu pine them together to obtain a complete proof. Readers who would heat the start of the st

Therefore, we adopt a more paper-like approach in w

Assume horizon defines (an upper bound on) the termination time of the system horizon = None, the system does not necessarily terminate. Note, however, that case our proof assumes there to be a finite number of jobs for technical reasons. horizon can be chosen to be arbitrarily large (the proof does not depend on its ma e.g., hundreds of even thousands of years, assuming the existence of a finite horiz

riable horizon : option instant nsider any type of tasks with a notion of pWCE

ontext {Task : TaskType}
 {pWCET_pmf : ProbWCET Task}.

nd their jobs. ontext {Job : finType}
 {job_task : JobTask Job Task}.

Consider a response-time monotonic scheduling algorithm ζ , where response-t monotonic means the following: assuming that all arrival times are fixed, an inci-the execution cost of any job cannot cause a decrease in the response time of Recall that ζ receives two vectors: a vector of arrival times A and a vector of job Variable ζ : @schedulerAC Job.

For brevity, let sched denote the probabilistic schedule generated by ζ for a given t sched S := compute pr schedule ζ (job arrival := 𝒜 of S) (job cost :=

s before, consider four parameters that describe a system under a bace $\Omega_{\rm r}$ a measure $\mu_{\rm r}$ job arrival times ${\cal A}$, and job execution costs Variable Ω : countType. Variable μ : measure Ω. Variable Æ : JobArrivalRV Job Ω μ. Variable Ĉ : JobCostRV Job Ω μ.

et S := {| $\Omega_of := \Omega$; $\mu_of := \mu$; $\mathcal{A}_of := \mathcal{A}$; $e_of := e$ |}.

Next, we assume that the aforementioned pWCET is an axiomatic pWCET. That is, for any job j and any arrival sequence ξ , there exists a partition of Ω into positive-probability events such that both partition dominance and partition independence are satisfied. Hypothesis H_axiomatic_pWCET : axiomatic_pWCET (μ_of S) (job_arrival := ALof S) (job_cost := Cof S)

Suppose we use the construction <code>replace_job_pET</code> presented in <code>probsa/rt /analysis/pETs_to_pWCETs</code> to replace the execution cost of given job <code>j_rep</code>. Let S denote the resulting system.

Variable j_rep : Job. Let S' := replace_job_pET j_rep S. For convenience, let tsk denote i's task. tsk := job_task j_rep.

and let µ_tsk denote the measure induced by pWC

Next, consider an arbitrary job j of any task and a duration

inally, let Exc denote the event that j's response time exceeds r time units in system S

t Exc := λ ω : Ω_of S, exceeds (response_time (job_arrival := J_of S) (job_cost := @_of (sched S) horizon j ω)

response_time
 (job_arrival := A_of S') (job_cost :
 (sched S') horizon j ω) remainder of this file serves for the most part to relate the probability of Exc with the

ample space Q_of S of system S into events

(Readers focused on technical detail may be interested in noting that partition ξ part are arrival sequences themselves. This detail may be by more casual readers.) As discussed in the paper, one can transfer the partition <code>Epart</code> to the system S' and denote it as <code>Epart'</code>. For further details see the function <code>partition</code> transfer-kethed partition. Let Epart' := extend_partition S Epart j_rep.

rtitions ξ part and ξ part ' are so similar that one can prove equilibrium that and ξ i and ξ i' that are "bickle equivalent" can be chosen in

.. we can show that $P<\mu_of S>{[Exc1]} \leq P<\mu_of S'>{[Exc2]}. Or, in other vords, we reduced the lemma statement to the hypothesis statement.$ Lemma transformation_is_pRT_monotone_step1 :
 P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}.

P<µ_OT >>{[EXC]} ≤ P<µ_OT >'>{[EXC' Step 2

 $H_{ineq}\xi_{partitioned}$, then we are done. But how do we prove su the next step, we prove the required inequality by introducing a new hy which it can be established. We then continue in this manner until reac hich it can be established in the high the prove w

the second step, we replace the partition over all arrival sequences with an ever neoding one arrival sequence. first, let us state the premise of our hypothesis H_ineq_§_part

 $\$ space of system S into events orresponding to different arrival sequences... Let Epart := partition_on_E µ. and Epart to system S' and denote it as Epar

Let \$part' := extend_partition \$ \$part j_rep. consider a pair of arbitrary pickle-equivalent elements ξi and ξ Variables (ξi : I ξpart) (ξi' : I ξpart'). Hypothesis ξ_equivalence : pickle_bij ξi ξi'

s already discussed, pickle-equivalence is quite strong and we ices ξ_i and ξ_i ' correspond to the same arrival sequence ξ . Because of the way titions on arrival sequences are constructed, we can extract an arrival sequence lacking a partition.

Variable ξ : arrival_sequence Job. Hypothesis H_part_unpack_ ξ : match ξi with exist ξ IN \rightarrow ξ end = ξ . ad of clunky indices $\xi_i : I \xi_{part}$ and $\xi_i' : I \xi_{part}'$, we will use "plain" rates ξ_f and ξ_f' that correspond to a fixed arrival sequence ξ in S and S',

Let $\xi f := (\lambda \omega, arr_seq \omega == \xi) : pred (\Omega_of S).$ Let $\xi f' := (\lambda \omega, arr_seq (proj1 S j_rep \omega) == \xi) : pred (\Omega_of S').$

Hypothesis H_ineq_{_fixed :
 P<µ_of S>{[Exc n ξf]} ≤ P<µ_of S'>{[Exc' n ξf
}

Step 3

Now, we can forget about $\xi part$ and $\xi part'$ and use ξ , ξf , and $\xi f'$ instead. Notice that here we use ξ to denote the deterministic arrival sequence. Predicates ξf and $\xi f'$ denote events that result in ξ in systems S and S', respectively.

Variable ξ : arrival_sequence sour. Let $\xi f := (\lambda \omega_i \operatorname{arr}_seq \omega == \xi) : pred (\underline{\Omega}_of S).$ Let $\xi f' := (\lambda \omega_i \operatorname{arr}_seq (projl S j_rep \omega) == \xi) : pred (\underline{\Omega}_of S').$

pothesis H_ξ_pos_prob : P<μ>{[ξf]} > 0.

Consider some countable type I and define a family of predicates part : $I \rightarrow pred \Omega$. Consider an index P1 : 1 and assume that part P1 has positive probability. Note that art might depend on §f. Next, let us assume the following: we have an event Sf := part the event ensures (1) the validity of the pWCET bound and (2) th

Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj1 simply returns the first component of a tuple).

With this, let us assume that the inequality that now includes Sf and Sf' holds f S>{[(Exc1 n ξf) n Sf]} ≤ P<µ_of S'>{[(Exc2 n ξf') n Sf'

 $\begin{array}{l} \text{formess } n_\text{ineq}_\text{upart}: \\ \text{f } I : \text{countType}) (part : I \rightarrow \text{pred } \Omega) (Pi : I) (p : PosProb } \mu (\xi f) \\ \text{let } f : = part Pi : pred (\Omega_of S) in \\ \text{let } Sf' := (\lambda \ \omega, \ Sf (proj1 \ S \ j_rep \ \omega)) : pred (\Omega_of \ S') in \end{array}$ (V (x : nat), $F < u > \{ f \ o \ f \ e \ i \ rep \} \mid Ef \ o \ Sf \ \} (x) \ge pWCET \ cdf \ (iob \ task \ i$

(V (c : Job → option work) (jobs : seq Job), j_rep \notin jobs → Peup+{[e_fix c [:: j_rep] ∩ e_fix c jobs | ξf ∩ Sf]} = Peup-{[e_fix c [:: j_rep] | ξf ∩ Sf]} × Peup-{[e_fix c jobs | P<μ_of S>{[(Exc n ξf) n Sf]} ≤ P<μ_of S'>{[(Exc' n ξf') n Sf']]

unany over to apply the convolution of n_ineq_v_part, we need to provide all th premises of the hypothesis. In particular, we have to present a countable type with family of predicates where every event ensures partition-independence and partici-tion.

NCET definition. Again, essentially, we prove that we can reduce the lemma to the settlesist using the properties ensured by aviamatic pWCET.

s before, now we need to prove the inequality we assumed in the previous step. For thi t us introduce all premises of the assumed inequality as variables and hypotheses. : countType) (part : Idx → pred Ω) (Pi : Idx)

Let Sf := part Pi : pred ($\Omega_o of S$). Let Sf' := ($\lambda \omega$, Sf (proj1 S j_rep ω)) : pred ($\Omega_o of S'$). Variable p1 : PosProb (µ_of S) (ξf n Sf). Variable p2 : PosProb (µ of S') (ξf' n Sf'). ypothesis H_pWCET_bounds_cond_cdf :
∀ (x : nat), F<μ>{[odflt0 (@ j_rep) | ξf ∩ Sf]}(x) ≥ pWCET_cdf (j

_independence : ption work) (jobs : seq Job), j_rep \notin joos → Peup-{[e_fix c [:: j_rep] n e_fix c jobs | ξf n Sf]} = Peup-{[e_fix c [:: j_rep] | ξf n Sf]} × Peup-{[e_fix c jobs | : In this step, we transform our inequality by moving $\xi f \cap S f$ and $\xi f' \cap S f'$ to the

irst, note that both $\xi f \cap S f$ and $\xi f' \cap S f'$ have the same part that our transformation does not change the probabilities

and_Sf_eq_prob : :>{[ξf n Sf]} = P<µ_of S'>{[ξf' n Sf']}

Hypothesis H_ineq_conditional :
 P<µ_of S>{[Exc | Ef n Sf]}

e_costs (ω : Ω_of S) := job.compute_costs ω (job_cost := e_of S). e_costs' (ω : Ω_of S') := job.compute_costs ω (job_cost := e_of S')

In this step, we replace events Exc and Exc' with events $\lambda \omega \rightarrow excee$ val times. Note the

ion that it must agree with ξ . Inside of the proof, we indeed construct A as a

and a subset of Ω_0 f S¹ that satisfies predicate $\xi f^+ n S f^+$), we have to adapt some c is notions to these new measures. Let μr := restrict (μ_of S) ($\xi f \cap Sf$) : measure (Ω_of S). Let $\mu r'$:= restrict (μ_of S') ($\xi f' \cap Sf'$) : measure (Ω_of S') Next, we define random variables, which are the same as those introduced earlier, except for the measure (μr instead of μ and $\mu r'$ instead of μ' .

Let Cr j := mkRvar µr (C_of S j). Let Cr' j := mkRvar µr' (C_of S' j). Similarly, we need to provide a new notion of projection that accounts for the restricte

Definition projw : $\forall (\omega : \Omega_of S'), (\xi f' \cap S f') \omega \rightarrow (\xi f \cap S f) (proj1 S j_rep \omega).$

Assume that we are given a list of job arriva

Variable (A : Job - option instant)

For technical reasons, we need to distinguish between jobs co

Now, we want to fix the job costs of all jobs except j_rep (recall that j_rep is the job for which we want to replace pET with pWCET). For this, again, we introduce partitioning of (into subsets, each containing a unique job cost assignment (and the job cost of j_rep is left unspecified). Let CSpart := partition_on_@s (job_cost := @r) μr (rem j_rep (enum Job)). Let CSpart' := extend_partition' S j_rep (ξf n Sf) CSpart (ξf' n Sf') projω

In the next step, we introduce CSpart<{Cs} and CSpart'<{Cs'} into the inequality. They ensure that the costs of all jobs (except j_rep) are fixed.

Assume that the inequality holds when conditioned on job-cost partitions ypothesis H_ineq_costs_partitioned :
Y (Cs : I CSpart) (Cs' : I CSpart) (CsEQU : pickle_bij Cs Cs'),
PqLof S>{[

w, exceeds (R A (compute_costs ω) j) r) CSpart⊲{Cs} | ξf n Sf]} Pu of S'>{[
 (λ ω, exceeds (*R* A (compute_costs' ω) j) r)
 n CSpart'{Cs'} [ξf' n Sf']}.

... then we can derive the inequality also without conditioning on costs. emma transformation_is_pRT_monotone_step6 : P<µ_of S>{[λ ω, exceeds (ℛ A (compute_costs ω) j) r | ξf ∩ Sf]} ≤ P<µ_of S'>{[λ ω, exceeds (ℛ A (compute_costs' ω) j) r | ξf ∩ Sf]]

Step 7

As before, we now proceed to derive the premise of the prior step To this end, consider two equivalent events corresponding to the same setting of job costs in the two systems. 'ariable (Cs : I CSpart) (Cs' : I CSpart') (CsEOU : pickle bii Cs Cs').

onsider two partitions of $\Omega_{of} S$ and $\Omega_{of} S'$ into cost of all jobs except j_rep. Let Cpart := partition_on_C µr (job_cost := Cr) j_rep. Let Cpart' := partition on C µr' (job cost := Cr') j rep.

Similarly to how we replaced job arrivals with A, we can do a similar trick costs with a vector C. However, what do we do with the cost of the unsp

ection Step7. Note that on the LHS of the below hypothesis, we now have the following ten

An important result of this transformation is that now exceeds (@ A j) r is a boolean value (true/false) and event CSpa ubset of Ω_of S such that *all* job costs are fixed. To extract a vector of job costs from the partition CSpart<fCs}, we find one $\omega \in CSpart<fCs$ } and compute the vector of costs for this ω . The resulting vector of costs will agree with the jobs in the system, given CSpart<fCs} is true.

Note that we use the construction update C j_rep c to update the cost of job j_rep vector C with a new value C. This is a way to define the notion of "reassembling" two jo cost-fixing vectors used in the paper.

before, we show that the prior step's premise follows e; that is, how the premise of Step 6 follows from the emma transformation_is_pRT_monotone_step7 : P≈µ_of S>{[(λ ω, exceeds (ℛ A (compute_costs ω) j) r) n CSpart⊲{CS} | ≤ P≈µ_of S'>{[(λ ω, exceeds (ℛ A (compute_costs' ω) j) r) n CSpart'of

↓ wo. We can use wo to compute C. Such a cost assignment agrees with ↓ because wo satisfies the initial predicate. Variable $\omega_0 : \Omega_of S$. Hypothesis H_ $\omega_oin_CS : CSpart < \{CS\} \omega_0$. Let c := (fun j $\rightarrow e_of S$ j ω_0) : Job \rightarrow option work.

Similarly to Step 7, which focused on the LHS, we now replace the Notice that both sides now have all costs fully fixed

Hypothesis H_ineq_cost_partitioned : ∑[=]_(c<-option work) P<µ_0 f S>(1 λ ω, [δδ exceeds (ℛ A (update C j_rep c) j) r, CSpart>(CS) ω δ CSpart=(Cs) ω δ = ∑[m]_(c<-option work) P=u_of S'>(1 ∧ u, [66 exceeds (# A (update C j_rep c) j) r, CSpart=(Cs') ω δ Cpart=(Cs') ω δ Cpart=(Cs') ω δ

om the above hypothesis, we can obtain the premise of Step

Lemma transformation_is_pRT_monotone_stepB : [1]=[.c=cution work] PMPL DE exceeds (#A (update c j_rep c) j) r, CSpart=(CS) u de CSpart=(CS) u de (Fu_cof c) (f r) (f r) (ST)) e Pku_cof c (# A (compute_costs' u) j) n (SSpart=(CS) (# r) sST)).

Step 9

sfine a random variable following the pWCET distribution. This random variable s j_reps cost after its pET has been replaced. For more details, see the file ysis/pETs_t_DMCETs and specifically the transformation _job_pETs. Recall that μ_{t} tsk is the measure induced by the given pWCET Let e_pWCET := mkRvar µ_tsk Some

Finally, we can use the independence property H_cond_independen LHS. At the same time, we can factorize the RHS just by construction.

pdate C j_rep c) j) r] . (CSpart⊲{CS} ω) && (ℓ j_rep ω === c) | ξf ∩ Sf]}

P<µ_of S>{[@ j_rep (=)

Remark RHS_facto iark RHS_factorization : [@]_{c<-option work} I[exceeds (ℛ A (update C j_rep c) j) r] × P<μ_of S'>{[λ ω, CSpart'⊲{Cs'} ω & (Cpart'⊲ isual. let us state

et R := schedulerAC_to_rtAC horizon Note that here we as:
 Step 2
 transformation of 6; however, bit number of 6; however, bit

[lexceeds (ℛ A (update c j_rep c) j) r] × (Pqu_of S>{[CSpart-q(CS} ξf ∩ Sf]} × Pqu_tsk>{[e_pWCET (=) c]}).
From the preceding factorized inequality, we can derive the premise of the previous step.
<pre>[e]_(c<-option work) P<u_of s="">(1 \u03b1) (Saeteceds (R A (update C j_rep c) j) r, Csaete(S) u 6.</u_of></pre>
Cparts(c) (u) {fn sf }} ≤ ∑[(u]_(c <option work)<br="">P<µ_of s><[(u, u)_n d unders f i see c) i) =</option>
(00 excess (x / rubate c]_rep ()))), CSpart*d(S') u & (Cpart*d(S') u) { 5f' n Sf']}.
Stan 10
In this step, we remove P<µ_of S>{[CSpart<(Cs) [f n Sf]] on the LHS and
PeqLof S>{[CSpart=4(CS] {f n Sf' }} on the RHS. Section Step10.
Let us assume that we can prove the following inequality, where $P<\mu_of S>{[CSpart$
<pre>Hypothesis H_ineq_without_other_costs : [im](=c-option work) I [exceeds (#A (update c j_rep c) j) r] × Peµ_of S>{[e j_rep (=) c ξf n Sf]} * [im](cereption work)</pre>
= 2 [t=_tc==0[ton work] I [exceeds (# A (update C j_rep c) j) r] × P<µ_ts>{[e_pWCET (=) c]}. then we can easily arrive at the preceding step's premise
Lemma transformation is pRT_monotone_step10: [im]_(c<-option work} Lexence(; 2 A (undate c i rep c) i) c]
<pre>x [Peq_pdfSs-{[CSpart=(CS]] {fn ff}} x Peq_pdfS-{[ej_rep (=) c] {fn Sf}}) x [eq_(c-exption work) [exceeds (@ A (update c j_rep (x) j) r] (CSpart=(fc) {fn Sf}) r]</pre>
End Step10.
Step 11
We have successfully removed constraints on job costs of jobs that are not equal to j_rep. Depending on the cost of j_rep, we can still get different behavior. In this section, we do a case analysis on the "critical" value of the cost of i rep.
Section Step11.
There are three possibilities: (1) whatever cost of ob j_rep we pick, the response time of job j never exceeds r, (2) whatever cost of job j_rep we pick, the response time of job j always exceeds r, and (3) there is some critical value c0 such that j's response time will
exceed rif and only if j_reps job cost is larger than r. Lemma cost_causing_exceedance_of_r: (* 3 *) (3 [c0: option work), V (c: option work),
<pre>is_true (c0 (<) () = exceeds (% A (update C j_rep () j) r) v (= 2 + (V (c : option work), exceeds (% A (update C j_rep () j) r) v (= 1 +) (V (c : option work), - exceeds (% A (update C j_rep () j) r).</pre>
Note that case (1) is easy since the inequality reduces to 0 ≤ 0. Remark jobs_rt_never_exceeds_r : (V c: option work, ¬ exceeds (R A (update C j_rep c) j) r) →
[[=]_(< <pre>ption work} I[exceeds (#A (update c j_rep c) j) r] × P<pre>ption = (c = [f ∩ Sf] } s []e] (<<pre>c<pre>ption work</pre></pre></pre></pre>
I[exceeds (# A (update c j_rep c) j) r] × P<µ_tsk>{[e_pWCET (=) c]}.
Note that case (2) is easy since the inequality reduces to 1 ≤ 1. Remark jobs_rt_always_exceeds_r: (V c: option work, exceeds (# A (update C j_rep c) j) r) →
<pre>[[m]_(c<-option work) I[exceeds (#A (update C j_rep c) j) r] * Peup-[[e] , rep [m] c Ef n Sf]} * [[m]_(c-enntion work)</pre>
T[exceeds (g A (update c j_rep c) j) r] × P<µ_tsk>{[e_pw(cEt (∈) c]}. Case (3) is interesting; so let us assume that c8 is the critical value, after which the
response time of job j will exceed r. However, then I [exceeds (
(<) C. We use the hancy (<) to account for the fact that both costs can be ⊥. Hypothesis H_ineq_c0_cause_exceedance : ¥ (c0: option work),
(* c, is_true (co (< c) + exceeds (X A (update c] = (p c)) /) → ∑[m] (co-option work) I(cd (< () × P4µ, df S+([d] = (p c) = () (c) { f n Sf }) ≤ ∑[m]_(co-option work) I(c0 (<) × P<µ, tsk-{[d] wcf (c) c }).
Given the hypothesis, we can prove the prior step's premise. Lemma transformation_is_pAT_monotone_step11 :
`I[exceeds (x A (update c j_rep c) j) r] × PeqLof S >{[ℓ j_rep (=) c ξf ∩ Sf]} ≤ [=].(c-eoption work] I[exceeds (x A (update c j_rep c) j) r] × PeqLtsk>{[ℓ_pWCET (=) c]}.
End Step11.
Step 12 Let us now assume that there is a cost c8 (of lob i rep) with the property that any cost
Variable c0 : option work.
where the state of the state o
simplify [[] on both sides. Section Step12.
Assuming that the following inequality, which is very similar to our initial assumption H_pWCET_bounds_cond_cdf,
$\begin{array}{l} \mbox{Hypothesis } H_{a}[nost_pWCET_bounds_cond_cdf: \\ \mbox{Peq_L} f \leq S - [(e_j, red) ($
we can prove the inequality assumed in the previous step.
$\begin{aligned} & z \left[e_{-} \left(c \leftarrow option nat \right) 1(c_{0}(c) < c_{1} \times e_{-} c_{0} - set(c) \right] + e_{-} \left(c_{-} c_{0} - c_{0} - c_{0} + c_{0} - c_{0} - c_{0} + c_{0} - c_{0} + c_{0} - c_$
Step 13
In the last step, we exploit the top-level assumption H_pWCET_bounds_cond_cdf to finish the proof.
Section Step13. Notice that the following statement is very close to the pWCET guarantee
H_pWCET_bounds_cond_cdf.
$P_{q_{1}}$ the first we did not make any new assumptions in this section; hence we are done
End Step13.
End StepByStepProof.
Statement and Proof of Theorem 1 Now we can combine all the steps to prove that a <i>single</i> job cost can be replaced with the
corresponding pWCET while preserving response-time monotonicity (Theorem 1 in the paper).
Assume horizon defines the termination time of the system. If horizon = None, the system does not terminate: however, as discussed at the beginning of the file, we assume
a finite number of jobs in either case. Variable horizon : option instant.
Consider any type of tasks with a notion of pWCET. Context {Task : TaskType}
(pwtel_pmt : Probmtel lask). Note that the arrivals and costs are determined by the system, which is defined next.
Context {Job : finType} {job_task : JobTask Job Task}.
Consider a response-time monotonic scheduling algorithm C, where response-time monotonic means the following: assuming that all arrival times are fixed, an increase of the execution cost of any job cannot cause a decrease of the response time of any job.
Recall that <pre>creative view ovectors: a vector of arrival times A and a vector of job costs C. Variable <pre>c := @schedulerAC Job Hypothesis H_rt_monotonic : rt_monotonic_scheduler horizon </pre></pre>
Let sched denote a schedule generated by ζ .
Let S be an arbitrary system
variable 5 : @system Job. and let j_rep be a job whose cost we want to replace.
Variable j_rep : Job. Let S' be the system where we replace i rep's cost with the corresponding nWCFT vis
replace_job_pET. Letterreplace_job_pET j_rep S.
Consider an arbitrary job)
and its response times x_j and x_j ' in schedules sched S are units', respectively. Let x_i := response time (sched S) (inb arrival := a of S) (inclusion of S) because i
Let xj := response_time (sched 5') (job_arrival := 4_0f 5') (job_cost = of 5') horizon j. If pWCET satisfies our notion of axiomatic pWCET
<pre>Hypothesis H_axiomatic_PWCET : axiomatic_PWCET (µ_of S) (job_arrival := .4_of S) (job_cost := .6_of S).</pre>
then the response-time distribution of job j in schedule sched S is \leq -bounded by the response-time distribution of job j in schedule sched S'. That is, $\Re_j \leq \Re_j^*$.
A 3 - 2 - 2 - 2 - 2 - 2 - 2 - 2 - 2 - 2 -



Ω_of S', exceeds (response_time (job_arrival := (sched S') horiz we can show that $\mathbb{P} < \mu$ of S>{[Exc1]} $\leq \mathbb{P} < \mu$ of S'>{[Exc2]}. Or, in othe emma transformation_is_pRT_monotone_step1 :
P<µ_of S>{[Exc]} ≤ P<µ_of S'>{[Exc']}. **FORMAL SPECIFICATION AND PROOFS** Step 2 o start the proof. First, we do a case analysis or titioned, then we are done. But how do we prove such an ineq ablished. Le then centime in this marker until reacting a hypothesis here over all arrival sequences with an event Specific Control Specific Control C Again, consider a part of the sample space of system 5 into events and proof step is accompanied by a link to the Of Coq specification o elements of sim pickle bij.) part) (ξi' : Ι 8 j ξi ξi' → ξi = present the proof i uvalent, it holds the part) (Ei' : I Epart') (EEQU : pickle_b [Exc ∩ ξpart⊲{ξi}]} ≤ ℙ<μ_of S'> The LHS and RHS of the inequality can be simplified to Step 13 $\mathbb{P}[\mathcal{C}_{J_o} > c_0 | \xi \wedge S_l]$ and $\mathbb{P}_f[\widehat{\mathcal{C}}_{J_o} > c_0]$, respectively. Using the fact that $\mathbb{P}[a > b] \leq \mathbb{P}[c > d] \iff \mathbb{P}[a \leq b] \geq \mathbb{P}[c \leq d]$, In the last step, we exploit the top-level assumption H_pWCET_bounds_cond_cdf to we transform the inequality to obtain (\mathbf{P}) : finish the proof. $\mathbb{P}[\mathcal{C}_{J_{\alpha}} \leq c_0 | \xi \wedge S_l] \geq \mathbb{P}_f[\widehat{\mathcal{C}}_{J_{\alpha}} \leq c_0].$ Section Step13. Finally, by construction (Def. 10), $\mathbb{P}_f[\widehat{\mathcal{C}}_{J_o} \leq c_0] = F_i(c_0)$. e replace the partition ov Notice that the following statement is very close to the pWCET guarantee Hence, we end up with $\mathbb{P}[\mathcal{C}_{J_o} \leq c_0 | \xi \wedge S_l] \geq F_i(c_0)$, which H pWCET bounds cond cdf. e premise of our hypothesis Η ineq ξ part follows () from partition-dominance (Def. 6). artition Epart of the sample space of sys Lemma transformation_is_pRT_monotone_step13 : fferent arrival sequences. P<µ_of S>{[C j_rep (<=) c0 | ξf ∩ Sf]} ≥
 P<µ_tsk>{[C_pWCET (<=) c0]}.</pre> S' and denote it as Epart extend_partition S Epart j rec redicates part : $I \rightarrow pred \Omega$. irditrary pickle-equivalent elements ξ_1 and ξ_1 part might depend on Ef. Also, note that we did not make any new assumptions in this section; hence, we are done. Next, let us *assume* the following: we have an even $S^{*} = p_{A}$: I ξpart) (ξi' : I ξpart'). Pi in system S such that equivalence : pickle_bij ξi ξi'. the event ensures (1) the validity of the pWCET bound and (2) h, partition-independence of job j_rep ed, pickle-equivalence is quite strong and we ca correspond to the same arrival sequence ξ . Be Let us define Sf's twin Sf' $\omega :=$ Sf (proj1 ω) in S' (recall that proj returns the End Step13. sequences are constructed, we can extract an first component of a tuple). With this, let us assume that the inequality that now includes Sf and Sf' holds arrival_sequence Job. part_unpack_ξ : match ξi with exist ξ I $\mathbb{P} < \mu_{of} S > \{ [(Excl n \xi f) n S f] \} \leq \mathbb{P} < \mu_{of} S' > \{ [(Excl n \xi f') n S f'] \}.$ Hypothesis H_ineq_Ω_part : dices ξi : I ξpart and ξi' : I ξpart', we w \forall (I : countType) (part : I \rightarrow pred Ω) (Pi : I) (ρ : PosProb μ ($\xi f \cap$ part Pi)), let Sf := part Pi : pred (Ω of S) in let Sf' := ($\lambda \omega$, Sf (proj1 S j_rep ω)) : pred (Ω of S') in f' that correspond to a fixed arrival sequence

ω, arr_seq ω == ξ) : pred (Ω_of S). ω, arr_seq (proj1 S j_rep ω) == ξ) : p

 $\mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge pWCET_cdf (job_task j_rep) x) \rightarrow \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]\}}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 (c j_rep) | \xif n Sf]}(x) \ge \mathbb{F}_{\mu > \{[odflt0 ($

(∀ (x : nat)













CONCLUSION

What we did:

- → First **fully formal** definitions of pET and pWCET
- → Adequacy property: formalization of "safe IID upper bound on pET"
- → **Prove** that our pWCET proposal is adequate
- → All **mechanized** with Coq



The Coq Proof Assistant





What we did:

- → First **fully formal** definitions of pET and pWCET
- → Adequacy property: formalization of "safe IID upper bound on pET"
- → **Prove** that our pWCET proposal is adequate
- → All **mechanized** with Coq

Are there **better** definitions?

- ► Maybe..?
- Please propose your preferred definition ... and present an adequacy proof



The Coq Proof Assistant





What we did:

- → First fully formal definitions of pET and pWCET
- → Adequacy property: formalization of "safe IID upper bound on pET"
- → **Prove** that our pWCET proposal is adequate
- → All **mechanized** with Coq

Are there **better** definitions?

- ► Maybe..?
- Please propose your preferred definition ... and present an adequacy proof

How to **derive** such axiomatic pWCET?

- Are existing methods compatible with it? (*MBPTA? EVT? SPTA?*)
- Can compatibility be proven in Coq?



The Coq Proof Assistant

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg





What we did:

- → First fully formal definitions of pET and pWCET
- → Adequacy property: formalization of "safe IID upper bound on pET"
- → **Prove** that our pWCET proposal is adequate
- → All **mechanized** with Coq

Are there **better** definitions?

- ► Maybe..?
- Please propose your preferred definition ... and present an adequacy proof

How to **derive** such axiomatic pWCET?

- Are existing methods compatible with it? (*MBPTA? EVT? SPTA?*)
- Can compatibility be proven in Coq?



Use axiomatic pWCET to build an **analysis**

• We prove that any sound analysis results in valid bounds. Let's verify one!

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg



What we did:

- → First **fully formal** definitions of pET and pWCET
- → Adequacy property: formalization of "safe IID upper bound on pET"
- → **Prove** that our pWCET proposal is adequate
- → All **mechanized** with Coq

Are there **better** definitions?

- ► Maybe..?
- Please propose your preferred definition ... and present an adequacy proof

How to **derive** such axiomatic pWCET?

- Are existing methods compatible with it? (*MBPTA? EVT? SPTA?*)
- Can compatibility be proven in Coq?

- Still do not like pWCET?
 - Come watch Filip's talk about pWCET-less **Correlation-Tolerant Analysis** on December 8th (Session 11@12:35pm)
- Use axiomatic pWCET to build an **analysis**
 - We prove that any sound analysis results in valid bounds. Let's verify one!

Sergey Bozhko, Filip Marković, Georg von der Brüggen, and Björn Brandenburg









BACKUP SLIDES

WHY AXIOMATIC pWCET?

Theorem (*paraphrased*). Consider a job $J_{i,j}$. Let $\mathscr{R}_{i,j}$ be the pRT of $J_{i,j}$ in the initial system and $\mathscr{R}_{i,j}^{\star}$ be the pRT of $J_{i,j}$ in a simplified system obtained via pWCET F_i . Then $\mathcal{R}_{i,i} \leq \mathcal{R}_{i,i}^{\star}$.

Hint:

1. Use axiomatic pWCET to construct a "copy" of the initial system, where pETs are replaced with job costs that are,

by construction, IID and have distribution F_i

2. Prove that pRT $\mathscr{R}_{i,j}^{\star}$ in the simplified system stochastically dominates the original pRT $\mathscr{R}_{i,i}$

Def. 7 (**\Frac{1}{2}**). A monotonically increasing function $F_i: \mathbb{W} \to \mathbb{W}$ [0,1] with $F_i(0) = 0$ and $\lim_{t\to\infty} F_i(t) = 1$ is an axiomatic **pWCET** for a task τ_i if, for every $J \in \tau_i$ and every fixed arrival sequence $\xi \in \Xi$, there exists a partition \mathfrak{S} (Def. 4) such that 1) C_J is partition-independent w.r.t. ξ and \mathfrak{S} (Def. 5), and 2) $F_i \mathfrak{S}$ -dominates \mathcal{C}_J w.r.t. ξ (Def. 6). Weakest precondition for which we could find a proof of the adequacy property





TWO TYPES OF pWCET

Dominance pWCET [1]

- $\rightarrow F_i : \mathbb{W} \rightarrow [0,1]$
- \rightarrow Given $C, F_i(c)$ defines a bound on probability of a job of task τ_i to have cost exceeding C

If $F_i(50) = 0.999$, then out of 100,000 jobs, at most 100 jobs are expected to have cost greater than 50

[1] Davis, Robert I., et al. "Analysis of probabilistic cache related pre-emption delays." [2] Edgar, Stewart, and Alan Burns. "Statistical analysis of WCET for scheduling."

Confidence pWCET[2]

 $\rightarrow F_i : \mathbb{W} \rightarrow [0,1]$

→ Given $C, F_i(c)$ defines a bound on probability that WCET of task τ_i does not exceed C

If $F_i(50) = 0.999$, no job is expected to have cost greater than 50 and we are 99.9% confident about it



