Algebraic Reasoning About Timeliness

Seyed Hossein HAERI
IOG, Belgium
University of Bergen, Norway
hossein.haeri@iohk.io

Peter W. THOMPSON
PNSol, UK
Peter.Thompson@pnsol.com

Peter VAN ROY
Université catholique de Louvain, Belgium
pvr@info.ucl.ac.be

Magne HAVERAAEN
University of Bergen, Norway
Magne.Haveraaen@uib.no

Neil J. DAVIES
PNSol, UK
Neil.Davies@pnsol.com

Mikhail BARASH
University of Bergen, Norway
mikhail.barash@uib.no

Kevin HAMMOND
IOG, UK
kevin.hammond@iohk.io

James CHAPMAN
IOG, UK
james.chapman@iohk.io

Designing distributed systems to have predictable performance under high load is difficult because of resource exhaustion, non-linearity, and stochastic behaviour. Timeliness, i.e., delivering results within defined time bounds, is a central aspect of predictable performance. In this paper, we focus on timeliness using the ΔQ Systems Development paradigm (ΔQSD, developed by PNSol), which computes timeliness by modelling systems observationally using so-called outcome expressions. An outcome expression is a compositional definition of a system's observed behaviour in terms of its basic operations. Given the behaviour of the basic operations, ΔQSD efficiently computes the stochastic behaviour of the whole system including its timeliness.

This paper formally proves useful algebraic properties of outcome expressions w.r.t. timeliness. We prove the different algebraic structures the set of outcome expressions form with the different ΔQSD operators and demonstrate why those operators do not form richer structures. We prove or disprove the set of all possible distributivity results on outcome expressions. On our way for disproving 8 of those distributivity results, we develop a technique called *properisation*, which gives rise to the first body of maths for **improper** random variables. Finally, we also prove 14 equivalences that have been used in the past in the practice of ΔQSD.

An immediate benefit is rewrite rules that can be used for design exploration under established timeliness equivalence. This work is part of an ongoing project to disseminate and build tool support for ΔQSD. The ability to rewrite outcome expressions is essential for efficient tool support.

1 Introduction

Designing distributed systems to have predictable performance under high load is difficult. At high load, resources such as network, memory, storage, or CPU capacity will be exhausted, causing a dramatic effect on performance. Prediction is difficult because the behaviour of system components and their interactions are both nonlinear and stochastic. For over 20 years, a small group of people associated with the company PNSol has worked on diagnosing and designing systems to predict and correct performance problems [17]. PNSol has developed the ΔQ Systems Development paradigm (ΔQSD) as part of this work. ΔQSD has been used in areas as diverse as telecommunications [20] [19] [6], WiFi [14], and distributed ledgers [5]. ΔQSD has been applied to many large industrial systems, with clients including BT, Vodafone, Boeing Space and Defence, and IOG (formerly IOHK).

C. Aubert, C. Di Giusto, S. Fowler & L. Safina (Eds.):
16th Interaction and Concurrency Experience (ICE 2023)
EPTCS 383, 2023, pp. 35–54, doi:10.4204/EPTCS.383.3

This paper defines and proves algebraic properties of the ΔQSD operators w.r.t. timeliness, i.e., delivering outcomes within the acceptable time-frames. In this paper, our sole resource of concern is time, although ΔQSD includes other types of resources and their interaction.

This theoretical work is part of an ongoing project to disseminate and build tool support for ΔQSD, to make it available to the wide community of system engineers. We base our work on the ΔQSD formalisation given by Haeri et al. [11], which defines outcome expressions and their semantics, and gives a real-world example of ΔQSD taken from the blockchain domain.

Contributions

This paper gives a firm mathematical foundation for ΔQSD, and uses this to establish important algebraic properties of the ΔQSD operators with respect to timeliness, i.e., when the relevant resource is time. This paper is based on a general model theory of resource analysis for systems specified using outcome expressions [12]. That model theory is the first of its kind and we specialise it using the timeliness analysis recipe that is commonly used in ΔQSD (Definition 3).

- We show that the set of outcome expressions forms different algebraic structures with the different ΔQSD operators (Theorems 1–4).
- We establish 3 distributivity results in Section 7 about the ΔQSD operators (Theorem 6).
- We rule out the formation of certain richer algebraic structures by the set of outcome expressions and the current ΔQSD operators (Remarks 2, 3, and 4).
- We develop two new techniques for analysing the validity of algebraic equivalences: a new technique that we call *Properisation* (Section 8) and another based on counterexamples (Section 7.2). We use those techniques to refute the remaining possible distributivity results in their full generality: 8 using properisation (Theorem 9) and 4 using counterexamples (Theorem 7).

  From a mathematical viewpoint, properisation is an important contribution of ours. As far as we know, properisation is the first body of maths developed for *improper random variable*s [21].
- We provide guidelines for studying the necessary/sufficient conditions for the distributivity results we refute the generality of (Section 7.1).
- We establish 14 equivalences that have been used in the past in the practice of ΔQSD (Section 6).

Full proofs can be found in the accompanying technical report [12], which also shows how Fig. 2 can be further elaborated using code running in a Jupyter notebook.

The primary practical results of this paper are to establish distributive properties of ΔQSD operators and other equivalences that are useful for rewriting outcome expressions. These enable common subexpressions to be moved, for example, to reduce representational complexity, with associated gains in tool performance. Rewriting can also be used to produce normal forms, and, in particular, to extract reliability/failure probabilities without fully evaluating the outcome expression. More generally, it can be used to establish equivalences between different designs with respect to their timeliness, even though their usage of other resources might differ, thereby allowing design exploration under equivalence.

2 Motivating Example: Cache Memory

We give an example of a memory system consisting of a local cache with a remote main memory. This example serves two purposes: First, it shows how outcome diagrams can be used to model nontrivial

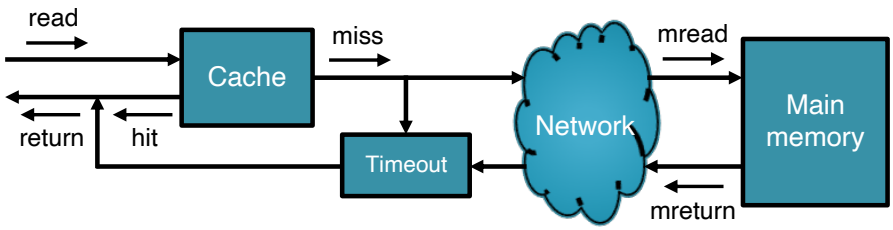


Figure 1: Block Diagram for a Cache with Networked Main Memory

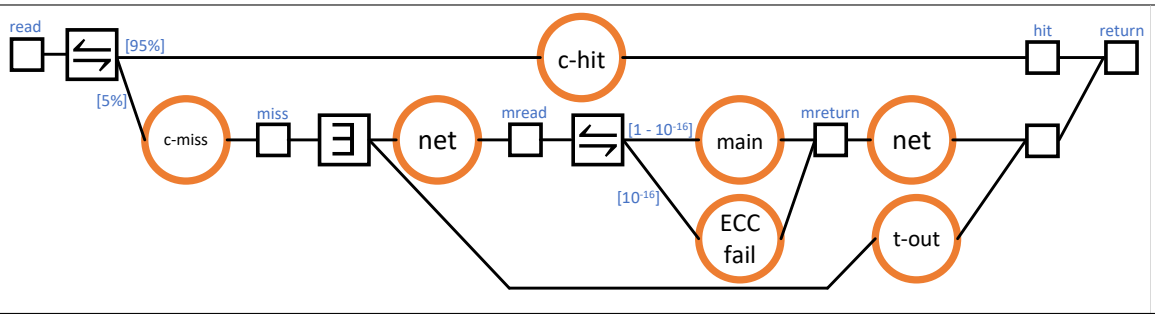


Figure 2: Outcome Diagram for the Cache of Figure 1

systems. Second, it shows the usefulness of the algebraic transformations of this paper. We give the block diagram and the outcome diagram for this example. We show how to rewrite (a simplified version of) the outcome diagram to swiftly compute failure (and, hence, success) rate of this design, giving it a 'back of an envelope' feasibility test. As we revisit this example later on, we will see that all this is possible because of the algebraic results proved in this paper.

Fig. 1 gives the block diagram of the memory system. A read message enters the cache; a cache hit – when the memory word is in the cache – results in an immediate return message; a cache miss – when the memory word is not in the cache – results in a main memory read. The main memory is across a network, so accessing it requires communication in both directions. Main memory access is guarded by a timeout in case of communication failure. The cache miss initialises the timeout timer; the *mreturn* message is passed through if it occurs before the timeout; otherwise, a *timeout* message is passed instead. Furthermore, there is a small probability that the remote main memory read fails.

Outcome Diagram for the Cache with Networked Memory Fig. 2 shows the outcome diagram for the memory system. We can define an *outcome* as what the system obtains by performing one of its tasks. Outcomes are shown using orange circles in the outcome diagrams. When there is a left-to-right path from one outcome to another, the right one is causally dependent on the left one. Small square boxes show the starting and terminating sets of events of the corresponding outcomes. Large square boxes are operators. In Fig. 2 there are two *probabilistic choices*, "$\leftrightharpoons$", and one *first-to-finish* synchronisation, "$\exists$". We assume that the cache hit rate is 95%. That is modelled using the leftmost probabilistic choice with two paths, one to each outcome ("cache hit" and "cache miss"), decorated with their corresponding probabilities. Timeout is modelled by a first-to-finish relationship between the main memory read and the timer. We assume that the main memory uses Error-Correction Codes (ECC) to catch bit errors, but nevertheless account for the possibility that a main memory access fails (e.g. because of hardware

failure) by giving it a failure rate of 10^{-16}. This assumption is modelled in Fig. 2 as a probabilistic choice between the "main" and "ECC fail" outcomes.

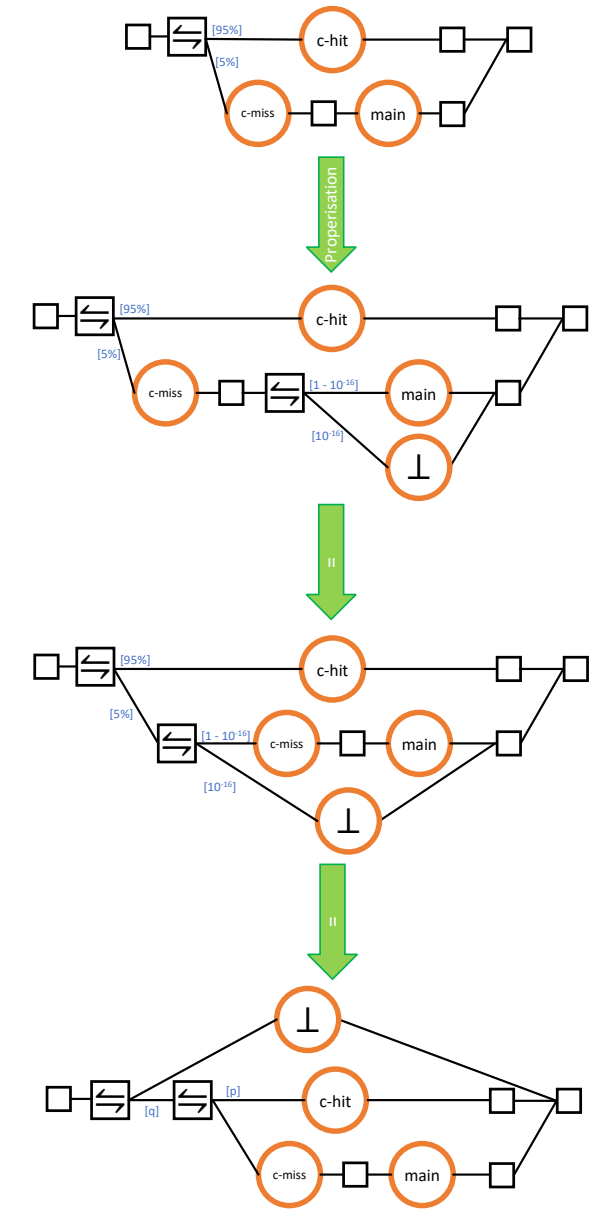


Figure 3: Steps for Swiftly Calculating the Failure Rate

Failure Rate Let us now compute the failure rate by doing algebraic transformations as defined in this paper. Without loss of generality, we can assume that the network has zero delay and the timeout is infinite. One can then simplify Figure 2 to the outcome diagram at the top of Figure 3. In that diagram, the ECC failure is hidden in the failure rate assigned to *main* in the timeliness analysis of the diagram. However, as we will prove in this paper, one can also *properise main* and explicitly demonstrate that failure rate as a probabilistic choice, **whilst retaining the level of timeliness**. The result of that properisation is shown in the second diagram from the top, where "$\perp$" represents *(unconditional) failure*.

According to the developments of this paper, one can rewrite the second diagram from the top to the third and then to the bottom one, again, **whilst retaining the level of timeliness**. What is important about the bottom diagram of Fig. 3 is that it comprises of a probabilistic choice between failure and everything else in the diagram. As will be proved later, for some p, and for $q = (1-0.05\times10^{-16}) = 0.9999999999999999995$, we have swiftly obtained the failure rate. Those numbers immediately tell the system engineer that, under the current assumptions about cache hit and main memory failure rates, **every** implementation will be infeasible if the overall success rate must be greater than q.

The techniques used for this example generalise in a straightforward fashion to any system modelled using an outcome diagram.

In the remainder of this paper, Examples 1–6 will come back to the developments of this section by supplying syntax, semantics, timeliness analysis, and authorising the rewrite steps taken here.

Closing Remarks on the Example Realistic cache memories are often more complex than this example, which gives rise to more complicated outcome diagrams in which "$\perp$" will appear at multiple depths. Thanks to the results we prove in this paper, techniques such as that of this section can be used to accumulate those $\perp$s.

While the probabilities in this example may seem small, they can combine with probabilities from other parts of the system, and it is important to be able to keep track of them. Dismissing them as 'minimal' risks missing potentially serious failures when many 'small' probabilities aggregate.

3 Background

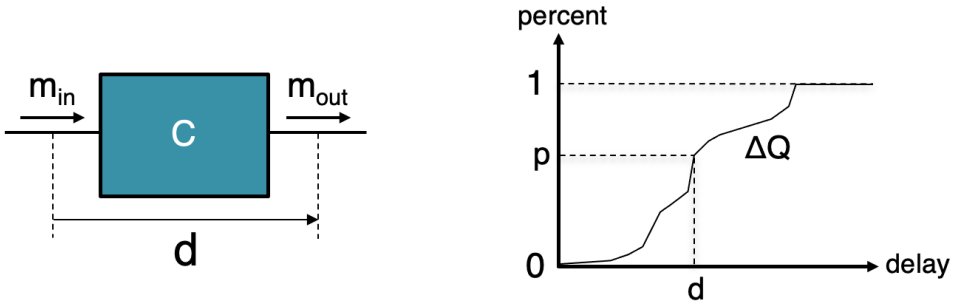
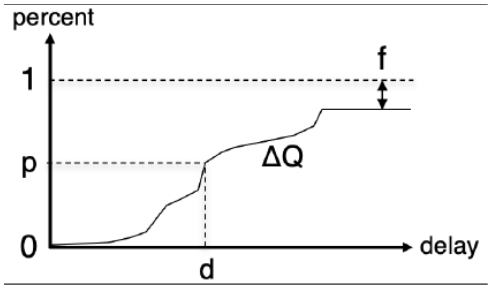


Figure 4: A Component's Operation and its Cumulative Delay Function

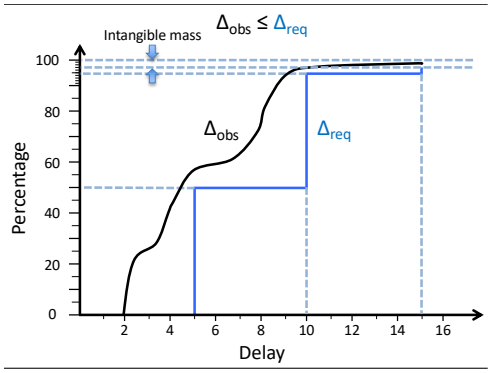
Outcome and Quality Attenuation Consider a component C which inputs message m_{in} and outputs message m_{out} after a delay d. Doing this many times will usually give different delays. We define a cumulative delay function so that p percent of delays are less or equal to d. Figure 4 gives an illustration.

The ΔQSD paradigm generalises this simple measurement. We measure delay not only for messages, but for all system behaviours that have a starting event and a terminating event. Given a starting event e_{in} and a terminating event e_{out}, what the system gains within the (e_{in}, e_{out}) time frame is called an instance of an *outcome*. We also generalise the property that we measure: we measure not only delay, but any property that makes the system less than perfect. The cumulative distribution function of the property is then called a *quality attenuation* and is denoted by a ΔQ. In what follows, we will consistently use the terms outcome and quality attenuation.

Failure It is straightforward to generalise the quality attenuation to model both delay and failure. It suffices to allow the cumulative delay function's limit to be less than 1. Figure 5a illustrates this possibility. There is an f percent probability that the delay is infinite, which corresponds precisely to a failure. For the component, it means simply that there is an



(a) Failure is modelled as a quality attenuation whose limit is less than 1.



(b) Timeliness: ΔQ_{obs} (the observed quality attenuation ΔQ) is always to the left and above ΔQ_{req} (the required ΔQ).

Figure 5: Failure and Timeliness

input message m_{in} with no corresponding output message m_{out}. Mathematically, the delay is modelled by a random variable that is allowed to be **improper**: The probability that it is infinite can be greater than 0. This probability is called the intangible mass of the Improper Random Variable (IRV) [21].

The ability to model delay and failure as a single quantity is a key strength of ΔQSD. It makes it easy to explore trade-offs between delay and failure in the system design. This ability shows up clearly in the algebra presented in this paper.

Timeliness We define *timeliness* as a relation (defined in [20]) between an observed ΔQ_{obs} and a required ΔQ_{req}. We say that the system *satisfies timeliness* for a given outcome if $\Delta Q_{obs} \leq \Delta Q_{req}$. Figure 5b illustrates this condition.

Outcome Expressions For a system consisting of multiple interconnected components, one can define a graph that combines all the components' outcomes. This graph defines the causal relationships between the outcomes and is called an *outcome diagram*. Each outcome diagram has a corresponding *outcome expression* – a mathematical description of the diagram [1]. Given an outcome expression and the quality attenuations of all its components, it is possible to compute the quality attenuation of the complete system. The reverse process can also be fruitful: given an outcome expression and the required quality attenuation of the complete system, one can estimate the required quality attenuations of its components. This gives the system designer a powerful tool for both design and diagnosis.

Outcome expressions can be manipulated according to algebraic rules, in particular those presented in this paper, which are useful to system designers using ΔQSD. As part of an ongoing project, we are building software tools to support ΔQSD, which can use the algebraic rules presented here for symbolic manipulation of outcome expressions.

ΔQSD

ΔQSD is a systems development paradigm that is able to compute many system properties early on in the design process, such as performance (latency and throughput), timeliness, resource consumption, risks, and feasibility. ΔQSD is used both for diagnosis and design:

- *System Diagnosis*. ΔQSD can analyse an existing system, to pinpoint anomalous behaviours so their origin can be found and the system can be corrected.
- *System Design*. ΔQSD can estimate performance trade-offs during the design process. At **every** step of the design process, performance of the complete system can be estimated by a computation on the partial design. This computation also determines whether or not the system is feasible, i.e., whether it can or cannot meet the requirements.

While historically ΔQSD has primarily been used to diagnose and correct problems in large industrial systems, PNSol has recently used ΔQSD to design the Shelley block diffusion algorithm as used in the Cardano blockchain [11]. More information on ΔQSD can be found in a tutorial given at HiPEAC 2023 [22].

[1] In this paper, we take the equivalence between the outcome expressions and outcome diagrams for granted. That equivalence is not the focus of this paper.

4 An Algebraic Perspective on Timeliness

4.1 Syntax of Outcome Expressions

Definition 1 (Haeri et al. [11])**.** Assume a set $\overline{\mathbb{B}}$ of primitive outcomes. We use variables $\beta \in \overline{\mathbb{B}}$ to represent individual primitive outcomes. We define the abstract syntax of outcome expressions as follows:

$$
\begin{array}{rcll}
\mathbb{O} \ni o & ::= & \beta & \text{primitive outcome} \\
& | & o \bullet\!\!\rightarrow\!\!\bullet\, o' & \text{sequential composition} \quad | \quad (o \parallel^{\forall} o') \quad \text{all-to-finish (a.k.a. last-to-finish)} \\
& | & o \underset{m'}{\overset{m}{\leftrightharpoons}} o' & \text{probabilistic choice} \quad | \quad (o \parallel^{\exists} o') \quad \text{any-to-finish (a.k.a. first-to-finish).}
\end{array}
$$

This defines outcome expressions as combinations of primitive outcomes β and four composition operators. In the case of probabilistic choice, m and m' are numeric weights which give the probabilities of choosing the left or right alternative, respectively. For convenience, we also introduce another notation $o \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} o'$ where the probability $(1-p)$ for the right alternative is implied. We distinguish two constant outcomes: $\top$ for "perfection" and $\bot$ for "unconditional failure."

Note that the operator "$\exists$" in the outcome diagrams is "$\parallel^{\exists}$" in the outcome expressions. That is to signify that when two outcomes are connected by first-to-finish, they are performed concurrently; hence the "$\parallel$" sign. One need not emphasise that concurrency in the outcome diagrams because our left-to-right directional convention on causal dependency already implies concurrency when forking off an "$\exists$" in the outcome diagrams. Similarly, for "$\parallel^{\forall}$" in the outcome expressions, the sign in the outcome diagrams is simply "$\forall$".

Example 1. Getting back to our motivating example, we can now transcribe the outcome diagram of Fig. 2 into an outcome expression:

$$
\mathit{c\text{-}hit} \underset{\rightharpoonup}{\overset{[95\%]}{\leftharpoondown}} (\mathit{c\text{-}miss} \bullet\!\!\rightarrow\!\!\bullet\, ((\mathit{net} \bullet\!\!\rightarrow\!\!\bullet\, (\mathit{main} \underset{\rightharpoonup}{\overset{[1-10^{-16}]}{\leftharpoondown}} \bot) \bullet\!\!\rightarrow\!\!\bullet\, \mathit{net}) \parallel^{\exists} \mathit{t\text{-}out})). \tag{1}
$$

We will use this outcome expression in further examples. □

4.2 Timeliness Semantics for Outcome Expressions

Let $\Delta\mathrm{Q}(x)$ denote the probability that an outcome occurs in a time $t \leq x$. In order to represent both delay and failure in a single quantity, a ΔQ is represented by an improper random variable (IRV), allowing the total probability not to reach 100% [21]. The *intangible mass* of such an IRV is $\Im(\Delta\mathrm{Q}) = 1 - \lim_{x\to\infty} \Delta Q(x)$. For a given ΔQ, the intangible mass $\Im(\Delta\mathrm{Q})$ encodes the probability of exceptions or failure occurring.

Denote the set $\mathbb{I}$ of all IRVs that are differentiable and the values of which are always greater than or equal to zero. Statistically speaking, every $\iota \in \mathbb{I}$ can be represented both using its Probability Density Function (PDF) or its Cumulative Distribution Function (CDF), where the former is the derivative of the latter. For convenience, we will freely switch between the two representations as the need rises. Fix a countable set of ΔQ variables Δ_v. We define $\Delta = \Delta_v \cup \mathbb{I}$ to denote both IRVs and ΔQ variables. When $\delta \in \Delta$ is in its CDF representation, we write δ' for its derivative, which is the PDF representation.

We first define a mapping between primitive outcomes $\overline{\mathbb{B}}$ and ΔQs.

Definition 2. We call a function $\Delta_\circ[\![.]\!] : \overline{\mathbb{B}} \to \Delta$ a *basic assignment* when $\Delta_\circ[\![\top]\!] = \mathbf{1}$ and $\Delta_\circ[\![\bot]\!] = \mathbf{0}$, where $\mathbf{1}$ and $\mathbf{0}$ are the functions always returning the constants 1 and 0, respectively.

Example 2. Every timeliness analysis of Fig. 2 *à la* ΔQSD needs a basic assignment that at least has mappings for the five individual outcomes in Equation (1) (namely, *c-hit*, *c-miss*, *t-out*, *net*, and *main*) so that $\Delta Q_{c\text{-}hit}$, $\Delta Q_{c\text{-}miss}$, $\Delta Q_{t\text{-}out}$, ΔQ_{net}, and ΔQ_{main}, are known initially. That is generally possible because: the first two are properties of the cache;the timeout is chosen by the designer; the network performance is known; and the main memory read time (and failure rate) is also known. □

We now define the semantics of an outcome expression as a mapping between the outcome expression and an IRV, for a given basic assignment.

Definition 3 (Haeri et al. [11])**.** Given a basic assignment $\Delta_\circ[\![.]\!] : \overline{\mathbb{B}} \to \Delta$, define $\Delta Q[\![.]\!]_{\Delta_\circ} : \mathbb{O} \to \mathbb{I}$ such that

$$\begin{aligned}
\Delta Q[\![\beta]\!]_{\Delta_\circ} &= \begin{cases} \mathbf{1} & \text{when } \Delta_\circ[\![\beta]\!] \notin \mathbb{I} \\ \Delta_\circ[\![\beta]\!] & \text{otherwise} \end{cases} \\
\Delta Q[\![o \bullet\!\!\rightarrow\!\!\bullet\, o']\!]_{\Delta_\circ} &= \Delta Q[\![o]\!]_{\Delta_\circ} * \Delta Q[\![o']\!]_{\Delta_\circ} \\
\Delta Q[\![o \overset{m}{\underset{m'}{\rightleftharpoons}} o']\!]_{\Delta_\circ} &= \tfrac{m}{m+m'} \Delta Q[\![o]\!]_{\Delta_\circ} + \tfrac{m'}{m+m'} \Delta Q[\![o']\!]_{\Delta_\circ} \\
\Delta Q[\![o \parallel^{\forall} o']\!]_{\Delta_\circ} &= \Delta Q[\![o]\!]_{\Delta_\circ} \times \Delta Q[\![o']\!]_{\Delta_\circ} \\
\Delta Q[\![o \parallel^{\exists} o']\!]_{\Delta_\circ} &= \Delta Q[\![o]\!]_{\Delta_\circ} + \Delta Q[\![o']\!]_{\Delta_\circ} - \Delta Q[\![o]\!]_{\Delta_\circ} \times \Delta Q[\![o']\!]_{\Delta_\circ}
\end{aligned}$$

Here, the notation $*$ denotes the convolution of two ΔQs. In the above formulae, the random variables are always represented using their CDFs except for sequential composition, where the representation is PDFs on both sides. Note that the PDF of $\top$ is the Dirac δ function. In what follows, we will drop $\Delta_\circ$ whenever the basic assignment is fixed throughout a computation.

One way to interpret Definition 3 is that $\Delta Q[\![.]\!]_{\Delta_\circ}$ is a homomorphism from the term algebra of outcome expressions $\mathbb{O}$ to an algebra of probability distributions $\mathbb{I}$.

Remark 1. Note that, according to Definition 3, we get $\Delta Q[\![o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2]\!] = \Delta Q[\![o_2 \bullet\!\!\rightarrow\!\!\bullet\, o_1]\!]$. This may seem counter-intuitive because $o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2 \neq o_2 \bullet\!\!\rightarrow\!\!\bullet\, o_1$. $\Delta Q[\![o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2]\!] = \Delta Q[\![o_2 \bullet\!\!\rightarrow\!\!\bullet\, o_1]\!]$ is, nonetheless, valid because, intuitively, $o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2$ is just as timely as $o_2 \bullet\!\!\rightarrow\!\!\bullet\, o_1$. See the proof of Theorem 2 [12] for the mathematical justification of that intuition. □

4.3 Motivating Example: Timeliness Analysis

Example 3. Given the developments of Example 2, here is how to work out the quality attenuation of Fig. 2 using Definition 3: Take $mem = net \bullet\!\!\rightarrow\!\!\bullet\, (main \overset{[1-10^{-16}]}{\underset{}{\rightleftharpoons}} \bot) \bullet\!\!\rightarrow\!\!\bullet\, net$ to be the outcome of the networked main memory read. We start by computing ΔQ_{mem}:

$$\Delta Q_{mem} = \Delta Q_{net} * ((1 - 10^{-16}) \times \Delta Q_{main} + 10^{-16} \times \Delta Q_{\bot}) * \Delta Q_{net} \tag{2}$$

which, because $\Delta Q_{\bot} = \mathbf{0}$, we can simplify to:

$$\Delta Q_{mem} = \Delta Q_{net} * (1 - 10^{-16}) \times \Delta Q_{main} * \Delta Q_{net} \tag{3}$$

The overall ΔQ_{obs} is then given by:

$$\Delta Q_{obs} = 0.95 \times \Delta Q_{c\text{-}hit} + 0.05 \times (\Delta Q_{c\text{-}miss} * (\Delta Q_{mem} + \Delta Q_{t\text{-}out} - \Delta Q_{mem} \times \Delta Q_{t\text{-}out})). \tag{4}$$

This computation gives us the CDF for the execution time of a memory read. The numeric computation is easily performed by a software tool. For readers interested in seeing fully worked-out numerical examples, we recommend looking up the tutorial [22]. □

Recall that, in Section 3, we defined timeliness as $\Delta Q_{obs} \leq \Delta Q_{req}$ (this relation is a partial order, defined in [11]). Definition 3 gives this more context. Using Definition 3, the systems engineer can work out the ΔQ_{obs} of an outcome so they can compare the result against the required ΔQ_{req}.

Example 4. Given the developments of Example 3, we can now get back to the plots in Fig. 5b. Taking the blue plot for ΔQ_{req}, the cache outcome diagrams developed in Section 2 are timely so long as 50% of the queries submitted to the cache can be handled within 5 units of time, 95% of them in 10 units, and 97% in 15 units. Furthermore, the cache is fine to drop 3% of the queries.[2]

Taking the black plot in Fig. 5b as that of Equation (4) after insertion of real numbers, our designed cache is timely enough because it always handles the queries within the acceptable time frame and drops less queries than the acceptable maximum. In other words, it has less delay and less failure rate. Visually, that amounts to the black plot always being to the left and above the blue plot. □

4.4 Connecting Algebra to Timeliness

In our accompanying technical report [12], we give a model theoretic formulation for studying the algebraic properties of resource consumption. This paper focuses on time as its sole resource of interest and uses that formulation for time exclusively without getting into the technical details of the formulation itself.

An algebraic structure often consists of a carrier set, a few operations on the carrier set, and a finite set of identities that those operations need to satisfy. Given our focus on timeliness à la ΔQSD, the carrier set will always be $\mathbb{O}$ in this paper. The full set of operators on $\mathbb{O}$ is $\{\bullet\!\!\rightarrow\!\!\bullet, \|^{\forall}, \|^{\exists}, \leftrightharpoons\}$. However, most algebraic structures do not need all those operators. Different structures work with different number of operations; for example, a monoid works with only one operation; whilst a group works with two. Finally, the identities are of the form $o_l = o_r$.

We take $\Delta Q[\![.]\!]$ (Definition 3) to be the model of time consumption for $\mathbb{O}$. We write

- $\odot\!\odot\, time \vDash o_l = o_r$ when $\Delta Q[\![o_l]\!] = \Delta Q[\![o_r]\!]$. That is when o_l and o_r are as timely.
- $\odot\!\odot\, time \vDash (\mathbb{O}, P) : s$ for an algebraic structure s and a set of ΔQSD operators P when $\odot\!\odot\, time \vDash o_l = o_r$, for every equation $o_l = o_r$
  - that is constructed using the operators in P, and
  - that is required for the formation of s.

With time being our solo resource of interest in this paper, we will drop the initial "$\odot\!\odot\, time \vDash$" from the above formulation hereafter.

5 Algebraic Structures

This section establishes several important properties on $\mathbb{O}$:

- probabilistic choice forms a magma (Theorem 1);
- sequential composition forms a commutative monoid with $\top$ and $\bot$ as the identity and absorbing elements (Theorem 2);
- all-to-finish forms a commutative monoid with $\top$ and $\bot$ as the identity and absorbing elements (Theorem 3);

[2] This is an instance of the "better-than" part of the Quantitative Timeliness Agreement (QTA) [20].

- any-to-finish forms a commutative monoid with $\bot$ and $\top$ as the identity and absorbing elements (Theorem 4); and
- neither all-to-finish nor any-to-finish nor their combination form the familiar richer algebraic structures (Remarks 2, 3, and 4).

Theorem 1. $(\mathbb{O}, \leftrightharpoons)$ *forms a magma when observing time.*

A magma is the weakest algebraic structure. That is because $\leftrightharpoons$ is not even associative. Despite this, expressions containing two consecutive occurrences of $\leftrightharpoons$ can still be re-associated. However, in this case the coefficients will change. Lemmas 2 and 3 give the exact formulae.

Theorem 2. $\circledcirc$ *time* $\vDash (\mathbb{O}, \bullet\!\!\rightarrow\!\!\bullet)$: *forms a commutative monoid with* $\top$ *and* $\bot$ *as the identity and absorbing elements, respectively.*

Theorem 3. $\circledcirc$ *time* $\vDash (\mathbb{O}, \|^{\forall})$: *forms a commutative monoid with* $\top$ *and* $\bot$ *as the identity and absorbing elements, respectively.*

Remark 2. It is important to notice that, when observing time, $(\mathbb{O}, \|^{\forall})$ does *not* form a group. That is because, in general, an outcome has no inverse element - intuitively, one can never undo an outcome!

In order to prove that claim formally, suppose otherwise. That is, suppose that there exist a pair of outcomes o_1 and o_2 such that $o_1 \|^{\forall} o_2 = \top$. Then, $\Delta\mathrm{Q}[\![o_1 \|^{\forall} o_2]\!] = \Delta\mathrm{Q}[\![\top]\!]$ which implies $\delta_1 \times \delta_2 = \mathbf{1} \Rightarrow \delta_2 = \frac{\mathbf{1}}{\delta_1}$. However, given that $\delta_1 \leq \mathbf{1}$, we get $\delta_2 \geq \mathbf{1}$. The latter inequality can only be satisfied when $o_1 = \top$. Restricting the application of ΔQSD to perfection is not practical. □

Theorem 4. $\circledcirc$ *time* $\vDash (\mathbb{O}, \|^{\exists})$: *forms a commutative monoid with* $\bot$ *and* $\top$ *as the identity and absorbing elements, respectively.*

Remark 3. Similar to the case for $\|^{\forall}$, it is important to note that, when observing time, $(\mathbb{O}, \|^{\exists})$ does not form a group. Again, it is the lack of an inverse element that is causing the trouble. Following our previous result, suppose that there exist a pair of outcomes o_1 and o_2 such that $o_1 \|^{\exists} o_2 = \bot$. Then, $\Delta\mathrm{Q}[\![o_1 \|^{\exists} o_2]\!] = \Delta\mathrm{Q}[\![\bot]\!]$ which implies $\delta_1 + \delta_2 - \delta_1 \times \delta_2 = \mathbf{0} \Rightarrow \delta_2 = \frac{\delta_1}{\delta_1 - 1}$. However, because $\delta_1 \leq \mathbf{1}$, we get $\delta_2 \leq \mathbf{0}$. But, only $\bot$ can satisfy the latter inequality. There is no reason to develop a system in which all the outcomes will fail unconditionally! □

Having established that both $(\mathbb{O}, \|^{\forall})$ and $(\mathbb{O}, \|^{\exists})$ form commutative monoids for time, a natural question is whether $(\mathbb{O}, \|^{\forall}, \|^{\exists})$ or $(\mathbb{O}, \|^{\exists}, \|^{\forall})$ form semi-rings. This is not the case, since they do not distribute over one another.

Lemma 1 helps Remark 4 show how the desirable distributivities fail.

Lemma 1. $\circledcirc$ *time* $\vDash o_1 \|^{\exists} o_2 = \top$ *implies* $o_1 = \top$ *and* $o_2 = \top$.

Remark 4. Neither $(\mathbb{O}, \|^{\forall}, \|^{\exists})$ nor $(\mathbb{O}, \|^{\exists}, \|^{\forall})$ form a semi-ring when observing time: for this to be the case, $\|^{\forall}$ and $\|^{\exists}$ would need to distribute over one another. The first distributivity requirement is:

$$o_1 \|^{\exists} (o_2 \|^{\forall} o_3) \stackrel{?}{=} (o_1 \|^{\exists} o_2) \|^{\forall} (o_1 \|^{\exists} o_3) \tag{5}$$

Equating $\Delta\mathrm{Q}[\![.]\!]$s of the two sides, one eventually makes it to the requirement that either $\delta_1 = \mathbf{0}$ or $\Delta\mathrm{Q}[\![(o_1 \|^{\exists} o_3) \|^{\exists} o_2]\!] = \top$. In other words, it follows by Lemma 1 that Equation (5) can only hold under the trivial conditions when either $o_1 = \bot$ or $o_1 = o_2 = o_3 = \top$. The second distributivity requirement is

$$o_1 \|^{\forall} (o_2 \|^{\exists} o_3) \stackrel{?}{=} (o_1 \|^{\forall} o_2) \|^{\exists} (o_1 \|^{\forall} o_3) \tag{6}$$

Again, equating $\Delta\mathrm{Q}[\![.]\!]$s of the two sides, one eventually comes to observe that Equation (6) only holds when $\delta_1 = \mathbf{1} \wedge \delta_2 \neq \mathbf{0} \wedge \delta_3 \neq \mathbf{0}$, i.e., when $o_1 = \top \wedge o_2 \neq \bot \wedge o_3 \neq \bot$. □

$$\bot \leftrightharpoons \bot = \bot \qquad (o_1 \leftrightharpoons \bot) \bullet\!\!\rightarrow\!\!\bullet\, o_2 = (o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2) \leftrightharpoons \bot \qquad o \bullet\!\!\rightarrow\!\!\bullet\, \bot = \bot \qquad \top \leftrightharpoons \top = \top$$

$$\bot \bullet\!\!\rightarrow\!\!\bullet\, o = \bot \qquad o_1 \bullet\!\!\rightarrow\!\!\bullet\, (o_2 \leftrightharpoons \bot) = (o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2) \leftrightharpoons \bot \qquad \top \bullet\!\!\rightarrow\!\!\bullet\, o = o \qquad o \bullet\!\!\rightarrow\!\!\bullet\, \top = o$$

$$\top \|^{\forall} o = o \qquad (o_1 \leftrightharpoons \top) \bullet\!\!\rightarrow\!\!\bullet\, o_2 = (o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2) \leftrightharpoons o_2 \qquad o_1 \bullet\!\!\rightarrow\!\!\bullet\, (o_2 \leftrightharpoons \top) = (o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2) \leftrightharpoons o_1$$

$$\bot \|^{\exists} o = o \qquad o_1 \underset{\leftharpoondown}{\overset{[p]}{\rightharpoonup}} (o_2 \underset{\leftharpoondown}{\overset{[q]}{\rightharpoonup}} \top) = o_2 \underset{\leftharpoondown}{\overset{[q(1-p)]}{\rightharpoonup}} (o_1 \underset{\leftharpoondown}{\overset{\left[\frac{p}{1-q(1-p)}\right]}{\rightharpoonup}} \top) \qquad \bot \underset{\leftharpoondown}{\overset{[p]}{\rightharpoonup}} (\bot \underset{\leftharpoondown}{\overset{[q]}{\rightharpoonup}} o) = \bot \underset{\leftharpoondown}{\overset{[p+(1-p)q]}{\rightharpoonup}} o$$

Figure 6: Equivalences Containing $\top$ and $\bot$

6 Equivalences Containing Constant Outcomes

ΔQSD is already in use by its practitioners, who, amongst other usages, simplify outcome expressions according to their timeliness analysis. In particular, Figure 6 distils a list of equivalences that are used in such simplifications. Those equivalences all contain constant outcomes ($\top$ or $\bot$).

Equivalences of Figure 6 provide the basis for rewrite rules that are useful for construction of normal forms, such as expressing a given system as a convolution of probabilistic choices or a probabilistic choice of convolutions. Such rewriting allows for: extraction of common sub-expressions permitting aggregation of failure rates (distinguishing between conditional and non-conditional failure); identifying minimal delays; and highlighting branching probabilities to identify issues of relative criticality. This is useful for quickly assessing whether a particular outcome decomposition is *feasible* without having to compute the complete ΔQ. See Section 2, for example. In addition, the equivalences of Figure 6 are very handy in the proofs of properties such as those established in this paper. Two examples, amongst many, are the proofs of Theorem 7 and Lemma 5.

Before we delve into Figure 6, we prove a result about re-associating probabilistic choice. Given an expression with two consecutive probabilistic choices, one of which wrapped inside a pair of parentheses, the ΔQSD practitioner might be interested in wrapping the other two inside a pair of parentheses – re-associating the probabilistic choices, in effect. Lemmata 2 and 3 give the conditions on the coefficients of those probabilistic choices.

Lemma 2. $o_1 \underset{\leftharpoondown}{\overset{[p]}{\rightharpoonup}} (o_2 \underset{\leftharpoondown}{\overset{[q]}{\rightharpoonup}} o_3) = (o_1 \underset{\leftharpoondown}{\overset{[p']}{\rightharpoonup}} o_2) \underset{\leftharpoondown}{\overset{[q']}{\rightharpoonup}} o_3$ *iff* $p' = \frac{p}{1-(1-p)(1-q)}$ *and* $q' = 1-(1-p)(1-q)$.

Lemma 3. $(o_1 \underset{\leftharpoondown}{\overset{[p]}{\rightharpoonup}} o_2) \underset{\leftharpoondown}{\overset{[q]}{\rightharpoonup}} o_3 = o_1 \underset{\leftharpoondown}{\overset{[p']}{\rightharpoonup}} (o_2 \underset{\leftharpoondown}{\overset{[q']}{\rightharpoonup}} o_3)$ *iff* $p' = pq$ *and* $q' = \frac{q(1-p)}{1-pq}$.

Theorem 5. *The equivalences in Fig. 6 are correct.*

Proof. We will only present the proof of $\bot \underset{m_2}{\overset{m_1}{\leftrightharpoons}} \bot = \bot$ here. The rest of the equivalences are proved similarly:

$$\Delta\mathrm{Q}[\![\bot \underset{m_2}{\overset{m_1}{\leftrightharpoons}} \bot]\!] = \frac{m_1}{m_1+m_2}\mathbf{0} + \frac{m_2}{m_1+m_2}\mathbf{0} = \mathbf{0} = \Delta\mathrm{Q}[\![\bot]\!].$$

■

Remark 5. The very last equivalence in Fig. 6 was incorrectly formulated (though never published) prior to this paper. Thanks to the formalisation developed in [11], that mistake was corrected, and the equivalences have been given a sound footing. □

6.1 Motivating Example: Correctness of the Three Bottom Rewrites

Example 5. We are now in position to confirm the steps taken in Fig. 3. Note first that, after dismissing the back-and-forth network connections and the timeout, Equation (1) simplifies to

$$c\text{-}hit \overset{[95\%]}{\leftrightharpoons} (c\text{-}miss \bullet\!\!\rightarrow\!\!\bullet (main \overset{[1-10^{-16}]}{\leftrightharpoons} \bot)) \tag{7}$$

which, according to Theorem 5, is equivalent to

$$c\text{-}hit \overset{[95\%]}{\leftrightharpoons} ((c\text{-}miss \bullet\!\!\rightarrow\!\!\bullet main) \overset{[1-10^{-16}]}{\leftrightharpoons} \bot) \tag{8}$$

which, again, can be rewritten using Lemma 2 as

$$(c\text{-}hit \overset{[\cdot]}{\leftrightharpoons} (c\text{-}miss \bullet\!\!\rightarrow\!\!\bullet main)) \overset{[q]}{\leftrightharpoons} \bot \tag{9}$$

for $q = (1 - 0.05 \times 10^{-16}) = 0.999999999999999995$. Equations (7), (8), and (9) are the outcome expressions for the bottom three outcome diagrams of Fig. 3, respectively. □

7 Distributivity

In this section, we consider the distributivity results between the ΔQSD operators. Recall that out of the four $\mathbb{P}$ operators, three are commutative (i.e., $\bullet\!\!\rightarrow\!\!\bullet$, $\|^{\forall}$, and $\|^{\exists}$) and one is not (i.e., $\leftrightharpoons$). Hence, it is only possible for right- and left-distributivity to differ when $\leftrightharpoons$ is the outermost operator. That gives rise to $2 \times \binom{3}{1} + \binom{3}{1}\binom{3}{1} = 15$ possible ways for distributing $\mathbb{P}$ operators over each other. Theorem 6 establishes 3 of those 15. In Section 7.1, we show how the routine technique for examining the equivalence of expressions (i.e., equating the $\Delta\mathbb{Q}[\![.]\!]$ of the two sides) is not that helpful for the study of the remaining 12 distributivity results. That leads to Sections 7.2 and 8, which disprove the generality of 4 and 8 distributivity results using counterexamples (Theorem 7) and properisation (Theorem 9), respectively.

We use the following syntactic convention: when, in an equivalence, two $\leftrightharpoons$s are used without weights, each on precisely one side of the equivalence, we will assume that the weights of those $\leftrightharpoons$s are the same. We therefore do not bother to repeat those weights. For example, in the theorem below, there exist weights m_2 and m_3 such that $o_2 \overset{m_2}{\leftrightharpoons}_{m_3} o_3$ and $(o_1 \bullet\!\!\rightarrow\!\!\bullet o_2) \overset{m_2}{\leftrightharpoons}_{m_3} (o_1 \bullet\!\!\rightarrow\!\!\bullet o_3)$, but we omit these.

Theorem 6. *Let $o_1, o_2, o_3 \in \mathbb{O}$ and $p \in \{\bullet\!\!\rightarrow\!\!\bullet, \|^{\forall}, \|^{\exists}\}$. Then,*

- $\circledcirc\!\!\circledcirc\, time \vDash o_1\ p\ (o_2 \leftrightharpoons o_3) = (o_1\ p\ o_2) \leftrightharpoons (o_1\ p\ o_3)$, *and*
- $\circledcirc\!\!\circledcirc\, time \vDash (o_1 \leftrightharpoons o_2)\ p\ o_3 = (o_1\ p\ o_3) \leftrightharpoons (o_2\ p\ o_3)$.

7.1 Potential Distributivity

As we are going to see in Sections 7.2 and 8, the remaining 12 potential distributivity results do not hold **in general**. Nevertheless, this section uses the routine technique for studying the equivalence of expressions: Equating the $\Delta\mathbb{Q}[\![.]\!]$ of the two sides. That is important because:

- firstly, it shows why the routine technique does not help, thereby motivating the next sections;
- secondly, it presents some of the necessary conditions for those distributivity results to hold. Although pretty immature, such conditions help the ΔQSD practitioner to verify, under special circumstances, whether their given IRVs can satisfy the provided conditions.

We do not know of better necessary conditions for the remaining 12 results (*if indeed they are soluble at all*). In this section, we demonstrate the necessary conditions of one distributivity result out the 12.

We begin by Proposition 1, which is a simple yet handy result.

Proposition 1. *Suppose that* $o_1 = o_2 \bullet\!\!\rightarrow\!\!\bullet\, o_3$. *Then,* $\odot\!\!\odot$ *time* $\vDash \delta_1(t) = \int (\delta_2' * \delta_3')(t)\,\mathrm{d}t$.

When observing time, for

$$(o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2) \overset{m}{\underset{m'}{\leftrightharpoons}} o_3 \overset{?}{=} (o_1 \overset{m}{\underset{m'}{\leftrightharpoons}} o_3) \bullet\!\!\rightarrow\!\!\bullet\, (o_2 \overset{m}{\underset{m'}{\leftrightharpoons}} o_3) \tag{10}$$

to hold, according to Proposition 1,

$$\begin{aligned}\Delta\mathrm{Q}[\![(o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2) \overset{m}{\underset{m'}{\leftrightharpoons}} o_3]\!] &= \frac{m}{m+m'} \int (\delta_1' * \delta_2')(t)\,\mathrm{d}t + \frac{m'}{m+m'}\delta_3 \\ &= \frac{m}{m+m'} \iint \delta_1'(\tau)\delta_2'(t-\tau)\,\mathrm{d}\tau\,\mathrm{d}t + \frac{m'}{m+m'}\delta_3 \end{aligned} \tag{11}$$

and

$$\begin{aligned}\Delta\mathrm{Q}[\![(o_1 \overset{m}{\underset{m'}{\leftrightharpoons}} o_3) \bullet\!\!\rightarrow\!\!\bullet\, (o_2 \overset{m}{\underset{m'}{\leftrightharpoons}} o_3)]\!] &= \int \left(\frac{m}{m+m'}\delta_1' + \frac{m'}{m+m'}\delta_3'\right) * \left(\frac{m}{m+m'}\delta_2' + \frac{m'}{m+m'}\delta_3'\right)(t)\,\mathrm{d}t \\ &= \iint \left(\frac{m}{m+m'}\delta_1'(t) + \frac{m'}{m+m'}\delta_3'(t)\right) \times \left(\frac{m}{m+m'}\delta_2'(t-\tau) + \frac{m'}{m+m'}\delta_3'(t-\tau)\right) \mathrm{d}\tau\,\mathrm{d}t. \end{aligned} \tag{12}$$

For Equation (10) to hold, the right-hand-sides of Equations (11) and (12) need to be equal. That is,

$$\begin{aligned}&\frac{m}{m+m'} \iint \delta_1'(\tau)\delta_2'(t-\tau)\,\mathrm{d}\tau\,\mathrm{d}t + \frac{m'}{m+m'}\delta_3 = \\ &\iint \left(\frac{m}{m+m'}\delta_1'(t) + \frac{m'}{m+m'}\delta_3'(t)\right) \times \left(\frac{m}{m+m'}\delta_2'(t-\tau) + \frac{m'}{m+m'}\delta_3'(t-\tau)\right) \mathrm{d}\tau\,\mathrm{d}t \end{aligned} \tag{13}$$

This is a differential equation for which we do not know a general solution. Given particular values for δ_1, δ_2, and δ_3, however, the ΔQSD practitioner might be able to solve it.

7.2 Counterexamples

As will be worked out in Remark 6, properisation does not quite work for outcome expressions containing $\|^\exists$ because $\bot$ is not compositional under $\|^\exists$. In this section, we present a less advanced yet effective technique for refuting distributivity results: counterexamples. A single counterexample suffices to refute an equivalence. That is how Theorem 7 refutes 4 distributivity results out of the questionable 12 (in their full generality).

Theorem 7. *For every* $o_1, o_2, o_3 \in \mathbb{O}$,

$$o_1 \leftrightharpoons (o_2 \|^\exists o_3) \neq (o_1 \leftrightharpoons o_2) \|^\exists (o_1 \leftrightharpoons o_3) \qquad (o_1 \|^\exists o_2) \leftrightharpoons o_3 \neq (o_1 \leftrightharpoons o_3) \|^\exists (o_2 \leftrightharpoons o_3)$$
$$o_1 \|^\exists (o_2 \bullet\!\!\rightarrow\!\!\bullet\, o_3) \neq (o_1 \|^\exists o_2) \bullet\!\!\rightarrow\!\!\bullet\, (o_1 \|^\exists o_3) \qquad o_1 \bullet\!\!\rightarrow\!\!\bullet\, (o_2 \|^\exists o_3) \neq (o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2) \|^\exists (o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_3).$$

Proof. We only prove the last item here. The other inequalities can be proved similarly using the same technique. Take $o_2 = o_3 = \top$ and let $\Delta\mathrm{Q}[\![o_1]\!] = \delta_1$. By Theorem 5, $o_1 \bullet\!\!\rightarrow\!\!\bullet\, (o_2 \|^\exists o_3) = o_1 \bullet\!\!\rightarrow\!\!\bullet\, (\top \|^\exists \top) = o_1 \bullet\!\!\rightarrow\!\!\bullet\, \top = o_1$. Therefore,

$$\Delta\mathrm{Q}[\![o_1 \bullet\!\!\rightarrow\!\!\bullet\, (o_2 \|^\exists o_3)]\!] = \delta_1. \tag{14}$$

On the other hand, by Theorem 5, $(o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2) \|^\exists (o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_3) = (o_1 \bullet\!\!\rightarrow\!\!\bullet\, \top) \|^\exists (o_1 \bullet\!\!\rightarrow\!\!\bullet\, \top) = o_1 \|^\exists o_1$. Thus,

$$\Delta\mathrm{Q}[\![(o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2) \|^\exists (o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_3)]\!] = \delta_1 + \delta_1 - \delta_1\delta_1. \tag{15}$$

Equations (14) and (15) together imply $\delta_1 = 2\delta_1 - \delta_1^2 \Rightarrow \delta_1 = \mathbf{0} \vee \delta_1 = \mathbf{1} \Rightarrow o_1 = \bot \vee o_1 = \top$. The result follows because, for any other o_1 and $o_2 = o_3 = \top$, the two sides will not be equal. ■

8 Properisation

This section sets the stage using Theorem 8 for a technique that we call *properisation* and use for disproving equivalences (in their full generality).

Properisation is based on the following important observation: if two outcomes do not fail similarly, they are not equivalent. Properisation is an algebraic technique for swiftly extracting the failure behaviour of outcomes via rewriting but without assessing the rest of their timeliness behaviour. Once the failure parts of the timeliness behaviours are at hand for the two sides, one can check whether they are equal, and if they are not, deduce that the outcomes in question are therefore unequal.

Our intuition for the choice of name "properisation" for this technique follows: recall that ΔQs are CDFs (or PDFs) of **im**proper random variables. Properisation is a technique based on making the ΔQ of an outcome o proper (by scaling it) and restoring its amount of improperness – i.e., o's intangible mass, denoted by $\Im(\Delta Q(o))$ – as a probabilistic choice (of the right weights) between o and $\bot$. That is also the intention behind the symbol we use for properisation: "$\overline{\uparrow}$." As one can see in Figure 5a, the CDF of an improper random variable needs not to make it to the "ceiling" (i.e., 1). The symbol "$\overline{\uparrow}$" that we use is intended to resemble the act of 'sticking the CDF to the ceiling' (represented by the horizontal bar at the top of "$\overline{\uparrow}$")!

Now, the formal definitions of properisation.

Definition 4. For an $\iota \in \mathbb{I}$ such that $\Im(\iota) = i$, write $\iota' = \iota\overline{\uparrow}$ when $dom(\iota) = dom(\iota')$ and $\iota'(x) = \frac{1}{1-i}\iota(x)$ for every $x \in dom(\iota)$. Call ι' the properisation of ι.

Proposition 2. $\Im(\iota\overline{\uparrow}) = 0$, *for all* $\iota \in \mathbb{I}$.

Intuitively, for IRVs, "$.\overline{\uparrow}$" produces a scaled random variable with no intangible mass.

Definition 5. Fix two basic assignments Δ, Δ' and a base variable β such that $\Delta(\beta) = \iota$. Write $\Delta' = \Delta\overline{\uparrow}^{\beta}$ when

$$\Delta'(\beta') = \Delta(\beta') \quad \text{for } \beta' \neq \beta \qquad\qquad \Delta'(\beta') = \iota\overline{\uparrow} \quad \text{otherwise.}$$

We say $\Delta\overline{\uparrow}^{\beta}$ is the result of *properisation* of β in Δ.

Intuitively, $\Delta\overline{\uparrow}^{\beta}$ produces a new basic assignment that is the as same Δ everywhere except β, where the assigned IRV is propoerised.

Notation 1. Write $o[o'/\beta]$ for the familiar λ-Calculus notation for substitutions: o in which every instance of β is replaced by o'.

Definition 6. Fix a basic assignment Δ and a base variable β such that $\Delta(\beta) = \iota$ where $\Im(\iota) = i$. Write $(o,\Delta)\overline{\uparrow}^{\beta} = (o',\Delta')$ when $o' = o[(\beta \overset{[1-i]}{\leftrightarrows} \bot)/\beta]$ and $\Delta' = \Delta\overline{\uparrow}^{\beta}$. We say that o' is the result *properisation* of β in o according to Δ.

As a shorthand, we write $(o,\Delta)\overline{\uparrow}^{\beta_1,\beta_2}$ for $\left((o,\Delta)\overline{\uparrow}^{\beta_1}\right)\overline{\uparrow}^{\beta_2}$ and $\Delta\overline{\uparrow}^{\beta_1,\beta_2}$ for $\left(\Delta\overline{\uparrow}^{\beta_1}\right)\overline{\uparrow}^{\beta_2}$.

As one can see from Definition 6, the act of properisation of a base variable β in an outcome o is according to a given basic assignment Δ. That is, the move from the right-hand-side of $(o,\Delta)\overline{\uparrow}^{\beta} = (o',\Delta')$ to its left-hand-side is performed by taking two steps in unison:

1. scaling according to the intangible mass of $\Delta(\beta)$ so that β is no longer improper in the resulting new basic assignment Δ'; and,
2. replacing every occurrence of β in the outcome o with the probabilistic choice that is weighted according to the intangible mass of $\Delta(\beta)$, resulting in the new outcome o'.

The idea is that the intangible mass that Δ' takes away o' returns, leaving timeliness intact. Lemma 4 utilises that idea.

Lemma 4. *Suppose that* $(o,\Delta)\overline{\uparrow}^{\beta_1,\beta_2,\ldots,\beta_n} = (o',\Delta')$ *for some* $\beta_1,\beta_2,\ldots,\beta_n \in \overline{\mathbb{B}}$, $o,o' \in \mathbb{O}$ *and basic assignments* Δ *and* Δ'. *Then,* $\Delta Q[\![o]\!]_\Delta = \Delta Q[\![o']\!]_{\Delta'}$.

Theorem 8 utilises Lemma 4 for examining equivalence of pairs of outcome expressions with no properisation relationship.

Theorem 8. *Suppose* Δ *and* Δ' *are two basic assignments. Suppose also that* $o_1,o_1',o_2,o_2' \in \mathbb{O}$ *such that* $(o_1',\Delta') = (o_1,\Delta)\overline{\uparrow}^{\beta_1,\beta_2,\ldots,\beta_n}$ *and* $(o_2',\Delta') = (o_2,\Delta)\overline{\uparrow}^{\beta_1,\beta_2,\ldots,\beta_n}$, *for some* $\beta_1,\beta_2,\ldots,\beta_n \in \overline{\mathbb{B}}$. *Then,* $\Delta Q[\![o_1]\!]_\Delta = \Delta Q[\![o_2]\!]_\Delta$ *iff* $\Delta Q[\![o_1']\!]_{\Delta'} = \Delta Q[\![o_2']\!]_{\Delta'}$.

8.1 Motivating Example: Correctness of the Properisation Step

Example 6. Recall from Section 2 that we took the failure rate of our ECC to be 10^{-16}. One way to model that failure rate is to assume a basic assignment Δ such that $\Im(\Delta(\mathit{main})) = 10^{-16}$. Note also that the outcome expression for the top outcome diagram of Fig. 3 is

$$\mathit{c\text{-}hit}\,\underset{\rightharpoonup}{\overset{[95\%]}{\leftharpoondown}}\,(\mathit{c\text{-}miss} \bullet\!\!\rightarrow\!\!\bullet\, \mathit{main}).$$

Furthermore, recall from Example 5 that the outcome expression for the second diagram of Fig. 3 from the top is

$$\mathit{c\text{-}hit}\,\underset{\rightharpoonup}{\overset{[95\%]}{\leftharpoondown}}\,(\mathit{c\text{-}miss} \bullet\!\!\rightarrow\!\!\bullet\, (\mathit{main}\,\underset{\rightharpoonup}{\overset{[1-10^{-16}]}{\leftharpoondown}}\,\bot)).$$

Now, suppose another basic assignment $\Delta' = \Delta\overline{\uparrow}^{\mathit{main}}$. Observe first that the latter outcome expression above is the properisation of *main* in the former according to Δ. Finally, thanks to Lemma 4, we know that one can rewrite the former outcome expression to the latter provided that one also replaces Δ with Δ'. Hence, timeliness remains intact over taking the properisation step of Fig. 3. □

8.2 Disproving the Remaining Distributivity Results

Armed with Theorem 8, we can now outline the properisation technique:

Suppose two outcome expressions o and o' the equivalence of which is to be studied. One begins by studying the equivalence of $o\overline{\uparrow}^{\beta_1,\ldots,\beta_n}$ and $o'\overline{\uparrow}^{\beta_1,\ldots,\beta_n}$ for some $\beta_1,\ldots,\beta_n \in \overline{\mathbb{B}}$. Now, suppose that – after the application of algebraic laws – one gets to rewrite $o\overline{\uparrow}^{\beta_1,\ldots,\beta_n}$ to $(\ldots)\,\underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}}\,\bot$ and $o'\overline{\uparrow}^{\beta_1,\ldots,\beta_n}$ to $(\ldots)\,\underset{\rightharpoonup}{\overset{[p']}{\leftharpoondown}}\,\bot$. One concludes that $o \neq o'$ if one can show that $p \neq p'$.

We start the application of our properisation technique by obtaining some useful results. Lemma 5 paves the way for the applications of the above technique. They instruct one on how to accumulate failure at the rightmost corner when the operator between two pairs of parentheses is $\bullet\!\!\rightarrow\!\!\bullet$, $\leftrightharpoons$, and $\|^\forall$, respectively. Unfortunately, $\|^\exists$ has no such property, as will be shown by Remark 6.

Lemma 5. *For every* $o_1,o_2,o_3 \in \mathbb{O}$,

$$(o_1\,\underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}}\,\bot) \bullet\!\!\rightarrow\!\!\bullet\, (o_2\,\underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}}\,\bot) = (o_1 \bullet\!\!\rightarrow\!\!\bullet\, o_2)\,\underset{\rightharpoonup}{\overset{[p_1p_2]}{\leftharpoondown}}\,\bot$$
$$(o_1\,\underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}}\,\bot)\,\underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}}\,(o_2\,\underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}}\,\bot) = (o_1\,\underset{\rightharpoonup}{\overset{[q]}{\leftharpoondown}}\,o_2)\,\underset{\rightharpoonup}{\overset{[r]}{\leftharpoondown}}\,\bot \text{ where } q = \frac{pp_1}{p_2-pp_2+pp_1} \text{ and } r = p_2 - pp_2 + pp_1$$
$$(o_1\,\underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}}\,\bot)\|^\forall(o_2\,\underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}}\,\bot) = (o_1\|^\forall o_2)\,\underset{\rightharpoonup}{\overset{[p_1p_2]}{\leftharpoondown}}\,\bot.$$

Proof. We only prove the first equivalence here. The proof is similar for the other two equivalences.

By Theorems 6 and 5,

$$(o_1 \underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}} \bot) \bullet\!\!\rightarrow\!\!\bullet (o_2 \underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}} \bot) = ((o_1 \underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}} \bot) \bullet\!\!\rightarrow\!\!\bullet o_2) \underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}} \bot = ((o_1 \bullet\!\!\rightarrow\!\!\bullet o_2) \underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}} \bot) \underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}} \bot = (o_1 \bullet\!\!\rightarrow\!\!\bullet o_2) \underset{\rightharpoonup}{\overset{[p_1 p_2]}{\leftharpoondown}} \bot.$$

■

Remark 6. Interestingly enough, there is no p such that the following holds in its full generality:

$$(o_1 \underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}} \bot) \|^{\exists} (o_2 \underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}} \bot) \stackrel{?}{=} (o_1 \|^{\exists} o_2) \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} \bot.$$

Suppose there were such a p. One gets to observe after some calculations that equating the $\Delta\mathrm{Q}[\![.]\!]$ of the two sides implies $p = p_1 = p_2 = 1$ or $p = p_1 = p_2 = 0$. When $(o_1 \underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}} \bot) \|^{\exists} (o_2 \underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}} \bot)$ is $o_1 \|^{\exists} o_2$, in which o_1 and o_2 are being properised, that is either when $o_1 = o_2 = \top$ or $o_1 = o_2 = \bot$. □

Hereafter, we will write $o_1 \underset{\rightharpoonup}{\overset{[.]}{\leftharpoondown}} o_2$ to mean $o_1 \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} o_2$ for some unimportant p.

The desirable inequalities in Theorem 9 are all of the form $o_l \neq o_r$, with the outcome variables in o_l and o_r being o_1, o_2, and o_3. In order to show $o_l \neq o_r$, we proceed by properisation of o_1, o_2, and o_3 in o_l and o_r.

To that end, we fix a basic assignment Δ, such that $\Delta\mathrm{Q}[\![o_k]\!]_\Delta = \delta_k$ and $\Im(\delta_k) = i_k$ for $k \in \{1,2,3\}$. Then, we take $p_k = 1 - i_k$ for $k \in \{1,2,3\}$, $(o'_k, \Delta') = (o_k, \Delta)\overset{o_1,o_2,o_3}{\uparrow\!\uparrow}$ for $k \in \{l,r\}$. We show that $\Delta\mathrm{Q}[\![o'_l]\!]_{\Delta'} \neq \Delta\mathrm{Q}[\![o'_r]\!]_{\Delta'}$ to conclude that $\Delta\mathrm{Q}[\![o_l]\!]_\Delta \neq \Delta\mathrm{Q}[\![o_r]\!]_\Delta$ by Theorem 8 and the result follows.

Theorem 9. *For every $o_1, o_2, o_3 \in \mathbb{O}$,*

$$(o_1 \bullet\!\!\rightarrow\!\!\bullet o_2) \leftrightharpoons o_3 \neq (o_1 \leftrightharpoons o_3) \bullet\!\!\rightarrow\!\!\bullet (o_2 \leftrightharpoons o_3) \qquad o_1 \leftrightharpoons (o_2 \bullet\!\!\rightarrow\!\!\bullet o_3) \neq (o_1 \leftrightharpoons o_2) \bullet\!\!\rightarrow\!\!\bullet (o_1 \leftrightharpoons o_3)$$
$$(o_1 \|^{\forall} o_2) \leftrightharpoons o_3 \neq (o_1 \leftrightharpoons o_3) \|^{\forall} (o_2 \leftrightharpoons o_3) \qquad o_1 \leftrightharpoons (o_2 \|^{\forall} o_3) \neq (o_1 \leftrightharpoons o_2) \|^{\forall} (o_1 \leftrightharpoons o_3)$$
$$(o_1 \|^{\forall} o_2) \bullet\!\!\rightarrow\!\!\bullet o_3 \neq (o_1 \bullet\!\!\rightarrow\!\!\bullet o_3) \|^{\forall} (o_2 \bullet\!\!\rightarrow\!\!\bullet o_3) \qquad o_1 \bullet\!\!\rightarrow\!\!\bullet (o_2 \|^{\forall} o_3) \neq (o_1 \bullet\!\!\rightarrow\!\!\bullet o_2) \|^{\forall} (o_1 \bullet\!\!\rightarrow\!\!\bullet o_3).$$

Proof. We only prove

$$(o_1 \bullet\!\!\rightarrow\!\!\bullet o_2) \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} o_3 \neq (o_1 \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} o_3) \bullet\!\!\rightarrow\!\!\bullet (o_2 \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} o_3) \tag{16}$$

for a given p here. The rest can be proved similarly using Lemma 5.

Fix a basic assignment Δ, such that $\Im(\Delta\mathrm{Q}[\![o_k]\!]_\Delta) = i_k$ for $k \in \{1,2,3\}$. Take $p_k = 1 - i_k$ for $k \in \{1,2,3\}$. Pick o'_l and o'_r such that $(o'_l, \Delta') = ((o_1 \bullet\!\!\rightarrow\!\!\bullet o_2) \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} o_3, \Delta)\overset{o_1,o_2,o_3}{\uparrow\!\uparrow}$ and $(o'_r, \Delta') = ((o_1 \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} o_3) \bullet\!\!\rightarrow\!\!\bullet (o_2 \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} o_3), \Delta)\overset{o_1,o_2,o_3}{\uparrow\!\uparrow}$, for some basic assignment Δ'. Our target inequality now becomes $o'_l \neq o'_r$, where

- o'_l is $((o'_1 \underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}} \bot) \bullet\!\!\rightarrow\!\!\bullet (o'_2 \underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}} \bot)) \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} (o'_3 \underset{\rightharpoonup}{\overset{[p_3]}{\leftharpoondown}} \bot)$, and
- o'_r is $((o'_1 \underset{\rightharpoonup}{\overset{[p_1]}{\leftharpoondown}} \bot) \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} (o'_3 \underset{\rightharpoonup}{\overset{[p_3]}{\leftharpoondown}} \bot)) \bullet\!\!\rightarrow\!\!\bullet ((o'_2 \underset{\rightharpoonup}{\overset{[p_2]}{\leftharpoondown}} \bot) \underset{\rightharpoonup}{\overset{[p]}{\leftharpoondown}} (o'_3 \underset{\rightharpoonup}{\overset{[p_3]}{\leftharpoondown}} \bot))$.

One can rewrite o'_l using Lemma 5 as $((o'_1 \bullet\!\!\rightarrow\!\!\bullet o'_2) \underset{\rightharpoonup}{\overset{[.]}{\leftharpoondown}} o'_3) \underset{\rightharpoonup}{\overset{[q]}{\leftharpoondown}} \bot$, where $q = p_3 - pp_3 + pp_1p_2$. Likewise, o'_r can be rewritten as $((o'_1 \underset{\rightharpoonup}{\overset{[.]}{\leftharpoondown}} o'_3) \bullet\!\!\rightarrow\!\!\bullet (o'_2 \underset{\rightharpoonup}{\overset{[.]}{\leftharpoondown}} o'_3)) \underset{\rightharpoonup}{\overset{[r_1 r_2]}{\leftharpoondown}} \bot$, where $r_1 = p_3 - pp_3 + pp_1$ and $r_2 = p_3 - pp_3 + pp_2$. Should $o'_l \neq o'_r$ not hold, one gets $q = r_1 r_2$. That is $p_3 - pp_3 + pp_1p_2 = (p_3 - pp_3 + pp_1)(p_3 - pp_3 + pp_2)$. But, that is not an equation that holds in general. Inequality (16) follows by Theorem 8. ■

9 Related Work

ΔQSD has been used in practice by a small group of practitioners for a couple of decades now [20, 19, 6, 14, 5]. The first formalisation of ΔQSD was, however, done quite recently by Haeri et al. [11]. We use that formalisation as a foundation.

Teigen et al [14] use ΔQ to develop a novel model of WiFi performance that produces complete latency distributions. The model is validated by comparison with previous modeling work and real-world measurements. It would be very interesting to apply ΔQSD to an outcome description of the protocol to see if this can replicate the same results.

Elsewhere, Gajda [10] attempts to model latency distributions but allows operations that do not preserve total probability, hence, leading to incorrect conclusions about failure probabilities.

Business Process Modelling and Notation (BPMN) [18] is a diagram scheme which is closely related to Outcome Diagrams (although with some details that are not considered relevant to ΔQSD). BPMN supports all ΔQSD operators except probabilistic choice. The closest operator is their "xor" gateway, which is essentially $\overset{[0.5]}{\leftrightharpoons}$. It is less expressive to the extent that it makes it impossible to consider systems such as the example in Section 2. Of the attempts for formalising BPMN, those of Wong and Gibbons [23, 24] are the most related to our work. Wong and Gibbons use the CSP process algebra for that purpose and further develop it to enable the specification of timing constraints on concurrent systems. Their developments allow mechanical verification of behavioural properties of BPMN diagrams using the FDR2 [15] refinement checker. Whilst Wong and Gibbons prove many interesting properties of their BPMN instances, they do not consider algebraic equivalences or algebraic structures for BPMN as we do in this work for ΔQSD. A less related BPMN formalisation work is that of El Hichami et al. [8], which provides a denotational semantics based on the Max+ algebra as an execution model for BPMN. They list a handful of algebraic equivalences in Max+ only axiomatically. Nevertheless, El Hichami et al. make no attempt to study the equivalence of BPMN diagrams based on their Max+ semantics.

When it comes to timeliness analysis, an important advantage of outcome diagrams over BPMNs is Definition 3, which formally defines the timeliness analysis of outcome diagrams. Definition 3 is fundamental to the applicability of the model theory we employ in this paper (Section 4.4). We are not aware of any formally defined recipe for timeliness analysis of BPMNs. The two closest attempts that we could find are the following two: Friedenstab et al. [9] borrow constructs from Business Activity Monitoring [4] to augment BPMN with a graphical notation for describing certain timeliness matters. Likewise, Morales [16] informally describes how to transform BPMN diagrams to timed automata networks, suggesting qualitative analysis of timeliness.

Performance Evaluation Process Algebra (PEPA) [13] is an algebraic language for performance modelling of systems. PEPA is successful and well-published with a rich family of formalisations with various interesting theoretical properties. However, PEPA suffers from several shortcomings that make it difficult to apply to real-world software systems. For example, PEPA does not model open or partially-specified systems; every detail of the system needs to be determined in advance. Since PEPA does not allow goals and objectives to be specified, it offers no assistance when comparing the predicted performance with the requirements. PEPA also suffers from state explosion, rapidly making it impractical, although more recent PEPA technology employs continuous approximations of the states, which contain some of the state explosion. This is similar to the use of IRVs in ΔQSD but rather *ad hoc* compared with the systematic use of ΔQs in ΔQSD. Less conservative alternatives to PEPA like SCEL [7] allow open systems but suffer from even more state explosion. CARMA [2] addresses a lot of the problems with PEPA, using a fluid approximation to manage the state explosion.

PerformERL [3] is an Erlang toolset, which focuses on monitoring the relationship between load repeatability and internal resource allocation. The authors advertise their toolset as an assistant for making early stage performance decisions, but it is unclear how it does this. Uunlike ΔQSD, monitoring (like testing) requires implementation of the system specification up to a certain level. The closer the implementation is to the full specification, the more reliable the monitoring will become, but the analysis is then no longer early-stage. Less accurate monitoring, on the other hand, is not reliable for decision

making. The closest PerformERL gets to the work described in this paper is its lightweight theoretical work out of the monitoring overhead it imposes to the system under development.

Finally, Failure Modes Effects Analysis [1] (FMEA) considers how failures propagate through a system but, unlike ΔQSD, does not model delays. We are not aware of any formalisation of FMEA that can serve algebraic developments like those on failure in this paper.

10 Conclusion and Future Work

This paper lays down model-theoretic foundations for timeliness analysis à la ΔQSD. It establishes time as a resource that is consumed by outcomes. In doing so, it enables timeliness analysis *via* the study of quality attenuation, simultaneously capturing both delay and failure. With our focus being exclusively on timeliness, we discuss the algebraic structures that the ΔQSD operators form with outcome expressions (Theorems 1–4). We refute the formation of richer algebraic structures by the ΔQSD operators and outcome expressions (Remarks 2, 3, and 4). We consider the 15 distributivity results about the ΔQSD operators. We prove 3 (Theorem 6) and disprove 8 (Theorem 9) using the newly formalised technique developed in this paper called properisation (Theorem 8) and 4 using counterexamples (Theorem 7). We also provide guidelines for studying the existence of potential distributivity (Section 7.1). Finally, we establish 14 important equivalences that have already been used in the practice of ΔQSD over the past few decades (Lemmas 2–3 and Theorem 5).

Our immediate future work is to study the algebraic properties of other resources à la ΔQSD, with the eventual goal of providing an algebraic categorisation of resources. A sound theoretical foundation is essential for the construction of robust tool support, which is, in turn, a prerequisite for wider application of the ΔQSD paradigm. Currently, there is a numerically-based tool prototype. However, to deal effectively with large complex systems, this needs to be made more symbolic. The aim is for the expressions to be simplified before calculation, and to be able to represent performance unknowns. Algebraic structures are essential for correctly manipulating and simplifying expressions. This work informs both ongoing practical work and tool development. Conversely, consideration of specific aspects of system design and operation will inform the most productive directions for the theoretical developments.

To conclude, this paper has introduced a number of important algebraic properties for ΔQSD outcome expressions. These properties have a highly practical application in the analysis of timeliness and resource consumption. For the first time, we have shown distributivity of the ΔQSD operators over probabilistic choice, and placed a set of 'folklore' equivalences (Theorem 5) that are in common usage for ΔQSD on a sound footing. These equivalences are essential for rapid recognition of infeasibility and for sound manipulation of outcome expressions to reduce computational complexity.

Acknowledgements

This research is funded by IOG, Singapore as a part of an ongoing project for incorporating performance as a first-class factor of the software development life cycle. When the routine proof technique did not work for distributivity, Andre Knispel (of IOG) suggested that we could utilise easier properties to obtain the disproofs using contrapositive reasoning. We would like to thank him for that suggestion.

References

[1] (1980): *MIL-STD-1629A – Procedures for Performing a Failure Mode Effect and Criticality Analysis*. Technical Report, United States Department of Defense.

[2] L. Bortolussi, R. De Nicola, V. Galpin, S. Gilmore, J. Hillston, D. Latella, M. Loreti & M. Massink (2015): *CARMA: Collective Adaptive Resource-sharing Markovian Agents*. In N. Bertrand & M. Tribastone, editors: *Proc. 13th W. Quant. Aspects of Prog. Lang. and Sys.*, *EPTCS* 194, pp. 16–31, doi:10.4204/EPTCS.194.2.

[3] W. Cazzola, F. Cesarini & L. Tansini (2022): *PerformERL: A Performance Testing Framework for Erlang*. *Distributed Comp.* 35(5), pp. 439–454, doi:10.1007/s00446-022-00429-7.

[4] C. Costello & O. Molloy (2008): *Towards a Semantic Framework for Business Activity Monitoring and Management*. In: *AAAI Spring Symposium: AI meets business rules and process management*, pp. 17–27.

[5] D. Coutts, N. Davies, M. Szamotulski & P. Thompson (2020): *Introduction to the Design of the Data Diffusion and Networking for Cardano Shelley*. Technical Report, IOHK. Available at `https://hydra.iohk.io/build/20405228/download/1/network-design.pdf`.

[6] N. Davies, P. Thompson, G. Young, J. Newton, B. Teigen & M. Olden (2021): *Measuring Network Impact on Application Outcomes Using Quality Attenuation*. In: *Measuring Network Quality for End-Users*, Internet Architecture Board, pp. 43–52. Available at `https://www.iab.org/wp-content/IAB-uploads/2021/09/PNSol-et-al-Submission-to-Measuring-Network-Quality-for-End-Users-1.pdf`.

[7] R. De Nicola, D. Latella, A. L. Lafuente, M. Loreti, A. Margheri, M. Massink, A. Morichetta, R. Pugliese, F. Tiezzi & A. Vandin (2015): *The SCEL Language: Design, Implementation, Verification*, pp. 3–71. Springer, doi:10.1007/978-3-319-16310-9_1.

[8] O. El Hichami, M. Naoum, M. Al Achhab, I. Berrada & B. E. El Mohajir (2015): *An Algebraic Method for Analysing Control Flow of BPMN Models*. *iJES* 3(3), pp. 20—26, doi:10.3991/ijes.v3i3.4862. Available at `https://online-journals.org/index.php/i-jes/article/view/4862`.

[9] J.-P. Friedenstab, C. Janiesch, M. Matzner & O. Muller (2012): *Extending BPMN for Business Activity Monitoring*. In: *45th HICSS*, pp. 4158–4167, doi:10.1109/HICSS.2012.276.

[10] M. J. Gajda (2020): *Curious Properties of Latency Distributions*. *CoRR* abs/2011.05219, doi:10.1007/978-3-031-10461-9_10. Available at `https://arxiv.org/abs/2011.05219`.

[11] S. H. Haeri, P. Thompson, N. Davies, P. Van Roy, K. Hammond & J. Chapman (2022): *Mind Your Outcomes: The ΔQSD Paradigm for Quality-Centric Systems Development and Its Application to a Blockchain Case Study*. *Computers* 11(3), p. 45, doi:10.3390/computers11030045. Available at `https://www.mdpi.com/2073-431X/11/3/45`.

[12] S. H. Haeri, P. W. Thompson, P. Van Roy, M. Haveraaen, N. J. Davies, M. Barash & J. Chapman (2023): *On the Algebraic Properties of Timeliness*. Technical Report, IOG. Available at `http://www.pnsol.com/public/Algebraic-Timeliness-TR.pdf`.

[13] J. Hillston (1996): *A Compositional Approach to Performance Modelling*. Cambridge University Press, doi:10.1017/CBO9780511569951.

[14] B. Ivar Teigen, N. Davies, K. Olav Ellefsen, T. Skeie & J. Torresen (2022): *Quantifying the Quality Attenuation of WiFi*. In S. Oteafy, E. Bulut & F. Tschorsch, editors: *IEEE 47th LCN*, IEEE, pp. 189–197, doi:10.1109/LCN53696.2022.9843690.

[15] Formal Systems (Europe) Ltd (2012): *Failures-Divergence Refinement: FDR2 User Manual*. Available at `https://www.cs.ox.ac.uk/projects/concurrency-tools/download/fdr2manual-2.94.pdf`.

[16] L. E. M. Morales (2014): *Specifying BPMN Diagrams with Timed Automata: Proposal of Some Mapping Rules*. In: *9th CISTI*, pp. 1–6, doi:10.1109/CISTI.2014.6876897.

[17] Predictable Network Solutions Ltd (PNSol) (2022): Available at `http://www.pnsol.com`.

[18] K. J. Sherry (2012): *Business Process Modelling with BPMN: Modelling and Designing Business Processes Course Book using The Business Process Model and Notation Specification Version 2.0*. CreateSpace Independent Publishing Platform.

[19] P. Thompson (2022): *TR-452.2 Quality Attenuation Measurements using Active Test Protocols*. Technical Report, The Broadband Forum.

[20] P. Thompson & R. Hernadaz (2020): *Quality Attenuation Measurement Architecture and Requirements*. Technical Report TR-452.1, Broadband Forum. Available at `https://www.broadband-forum.org/download/TR-452.1.pdf`.

[21] K. S. Trivedi (2002): *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*, 2 edition. Wiley, New York, NY, USA.

[22] P. Van Roy, N. Davies, P. Thompson & S. H. Haeri (2023): *ΔQSD: Designing Systems with Predictable Latency at High Load*. Tutorial, HiPEAC 2023 (Conf. High Perf. Emb. Arch. & Compil.). Available at `shorturl.at/dmKSW`.

[23] P. Y. H. Wong & J. Gibbons (2011): *Formalisations and Applications of BPMN*. *SCP* 76(8), pp. 633–650, doi:10.1016/j.scico.2009.09.010. Available at `https://www.sciencedirect.com/science/article/pii/S0167642309001282`.

[24] P. Y. H. Wong & J. Gibbons (2011): *Property Specifications for Workflow Modelling*. *SCP* 76(10), pp. 942–967, doi:10.1016/j.scico.2010.09.007. Available at `https://www.sciencedirect.com/science/article/pii/S0167642310001735`.