Increasing Accuracy of Timing Models: From CPA to CPA+

Leonie Köhler, Borislav Nikolic, Rolf Ernst
Institute of Computer and Network Engineering, TU Braunschweig
Braunschweig, Germany
koehler—nikolic—ernst@tu-braunschweig.de

Marc Boyer
ONERA, The French Aerospace Lab
Toulouse, France
marc.boyer@onera.fr

***Abstract*—Formal analysis methods of embedded systems provide safe, but unfortunately often pessimistic bounds on response times. An important source of pessimism is the common approach to characterize service request either by the amount of data or the number of events to be processed. Several works, e.g. [1]–[4], have demonstrated that a dual model – which includes information on both data and events – is more accurate, especially for more complex scheduling problems. In this paper, we enrich Compositional Performance Analysis (CPA) by a new component interface which, as we show, is consistent with the generic dual model proposed in [3]. Furthermore, we discuss how composition of components should be realized and how the new information should be integrated into the analysis technique. The improved CPA is called CPA+, and we identify different types of scenarios where CPA+ is particularly beneficial.**

***Index Terms*—Timing analysis, compositional performance analysis, real-time system models**

I. Introduction

The objective of formal timing verification is to prove real-time properties of embedded software systems, buses and, networks. Of particular interest are formally verified lower and upper bounds on latencies, e.g., worst case response times (WCRTs) of software tasks or packet traversal times. A major issue with formal timing verification in practice is that the upper bounds on latencies are to such an extent pessimistic that they are often deemed as not useful by industry. One approach to tackle this problem is to reconsider whether the underlying system model of the timing verification mechanisms is accurate enough. An interesting starting point is to compare the system models of two major timing verification frameworks namely Network Calculus (NC) [5] and CPA [6]. Both NC and CPA have their useful domains of applications, and as being different they dominate each other in specific scenarios [7]. Why is that? One reason relates to how they model service request. In NC, a data flow $A(t)$ models service request in units of data to be processed up to time t. In CPA, an event flow $E(t)$ counts the number of events to be processed up to time t. Of course, conversion is possible but it is pessimistic: The standard approach is that NC obtains information on event count by $E(t) = \frac{A(t)}{p}$, where p is a known static packet size. CPA recovers information on the amount of data by weighting the event flow with the static packet size $A(t) = p \cdot E(t)$. It is obvious that these relations are not accurate if packet sizes may vary, which is the realistic case. Variations in packet sizes result from different packet types in a stream (e.g. I, P or B frames in MPEG video streams), a varying amount of payload to be transmitted, re-packaging (e.g. protocol translation), etc. We conclude that NC has an advantage in precisely modeling workload, while CPA is favorable when the number of events is decisive for modeling accuracy. Modern, sophisticated modeling problems have yet been a game changer: When scheduling decisions depend on *both* the number of occurred events and the amount of received workload, then it is desirable to have precise information on both data and events. Examples are (de)packetizers, gateways, and traffic shapers. In this paper, we make the case for a true dual system model in CPA, including both the workload-based and the event-based information. We refer to the work of Boyer et al. [3] which shows how the data flow $A(t)$ and the event flow $E(t)$ can be transformed into each other: $E = P \circ A$, where P is the so called packet function. The dual model, that we propose, comprises the triple A, E, P as well as their best case and worst case bounds $(\underline{\alpha}, \overline{\alpha}), (\underline{\eta}, \overline{\eta}), (\underline{\pi}, \overline{\pi})$. As an addition, we show that the event distance functions $(\underline{\delta}, \overline{\delta})$, which are often used in CPA, can be represented in this model. We discuss the question whether the triple A, E, P entails redundancy and how propagation of data flow resp. event flow between components should be realized, as well as how to integrate the new information properly in the analysis technique. Also, we illustrate in which types of scenarios the dual model is particularly beneficial. An interesting side effect of the enriched version of CPA, called CPA+, is that the input models are then interchangeable with NC.

II. Related Work

Both event and data count are important parameters to accurately quantify service request, and therefore the existing system models which consider only one of the two domains are ultimately unsatisfactory [4]. Examples for analysis frameworks with one-domain system models are Real-Time Calculus (RTC) [8], NC [5] (data/workload domain) and CPA [6] (event domain). Transformations between the data and event domain are indeed possible, yet the prevalent transformation techniques are pessimistic. Works from the RTC [1] resp. NC community [2] addressed this problem and showed that a packet function together with the data flow is necessary to obtain information on the associated event flow. The work of Boyer et al. [3] goes one step further and formulates how the input models of CPA and NC relate, discussing the subtleties

978-3-9819263-2-3/DATE19/©2019 EDAA

of transformations. The authors of [1] were the first to use a dual system model in the context of a compositional analysis framework namely RTC. They developed a new component interface as well as algorithms how to compute the output data/event flows. The works [2] [3] discuss the benefits of dual models only for single-component systems, neglecting the composition aspect. In this paper, our goal is to introduce a dual system model for CPA, and to provide the necessary new component interface, composition rules, as well as an algorithm how to compute an output data flow.

III. Elements of a Dual System Model

We present now the elements of a system model, which explicitly models the service request in terms of data to be processed *and* in terms of events to be processed. First, let us revisit several definitions from the unified model in [3].

A. Background and Preliminaries

Definition 1 (Data flow [3] [9])**.** *A data flow $A : \mathbb{R}^+ \to \mathbb{R}^+$ is a request function indicating the amount of data that has been sent up to time t.*

Definition 2 (Event flow [3])**.** *An event flow $E : \mathbb{R}^+ \to \mathbb{N}$ is a request function indicating the number of activation events that have occurred up to time t.*

A service request of a task is thus modeled by cumulative functions. The packet function P establishes a connection between the data flow and the event flow in the following way: $E = P(A(t)) = P \circ A$.

Definition 3 (Packet function [3])**.** *A packet function $P : \mathbb{R}^+ \to \mathbb{N}$ is a function indicating the number of packets in the first m units of the data flow.*

Both CPA and NC, which concentrate on best case and worst case scenarios, employ interval bounding pairs (IBPs) on data flow $\underline{\alpha}, \overline{\alpha}$, event flow $\underline{\eta}, \overline{\eta}$ and the packet function $\underline{\pi}, \overline{\pi}$. The concept of IBPs is introduced with Definition 4.

Definition 4 (Interval bounding pair [3])**.** *Let $f : \mathbb{R}^+ \to \mathbb{I}$ (with $\mathbb{I}$ being $\mathbb{R}^+$ or $\mathbb{N}$) and $\underline{\phi}, \overline{\phi} : \mathbb{R}^+ \to \mathbb{R}^+$ be functions, $\underline{\phi}$ and $\overline{\phi}$ non-decreasing. Then $\underline{\phi}$ and $\overline{\phi}$ are respectively a lower and upper interval bounding function of f if*

$$\forall t, d \in \mathbb{R}^+, \underline{\phi}(d) \leq f(t+d) - f(t) \leq \overline{\phi}(d).$$

Definition 5 (Pseudo-inverses)**.** *Let $f : \mathbb{D} \to \mathbb{I}$ (with $\mathbb{D}$ and $\mathbb{I}$ being $\mathbb{R}^+$ or $\mathbb{N}$)*

$$f^{\underline{-1}}(y) = \inf\{x \in \mathbb{D} \mid f(x) \geq y\}$$
$$f^{\overline{-1}}(y) = \sup\{x \in \mathbb{D} \mid f(x) \leq y\}$$

Note that Definition 5 is a generalization of the definition in [3] where $\mathbb{D}$ was $\mathbb{R}^+$.

B. Event Occurrence Function and Event Distance IBP

The aforementioned three functions, as well as their respective IBPs, are connected in a sense that one of them can be derived from the other two with a certain degree of pessimism (more details available in the work of Boyer et al. [3]). However, the CPA framework consists of yet another pair $(\underline{\delta}, \overline{\delta})$, which holds the information about the distance between events, and constitutes an essential part in the event propagation process for multi-component problems in CPA as detailed in Section IV. Specifically, $\underline{\delta}(n)$ and $\overline{\delta}(n)$ represent the minimum and the maximum distance between n consecutive events, respectively. Therefore, in order to establish a dual system model for CPA+, the pair $(\underline{\delta}, \overline{\delta})$ has to be expressed in the context of A, E, P and $(\underline{\alpha}, \overline{\alpha}), (\underline{\eta}, \overline{\eta}), (\underline{\pi}, \overline{\pi})$. In order to do that, we first define the event occurrence function which is also illustrated in Figure 1.

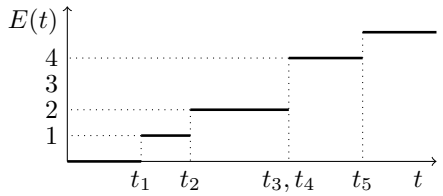


Fig. 1: Event flow E and arrival instants $t_1, t_2, \ldots$

Definition 6 (Event occurrence function)**.** *Let E be an event flow. The function $T : \mathbb{N} \to \mathbb{R}^+$, $T(i) = \min\{t \mid E(t) \geq i\}$, is a function indicating the time instants of event arrivals.*

The arrival time of the i^{th} event is the smallest time instant t_i which satisfies the following condition: $E(t_i) \geq i$, or alternatively: $t_i = min\{t \mid E(t) \geq i\}$. To unify the notation, we refer to that time instant as $T(i)$, i.e. $T(i) = t_i$. Similarly, it is possible to extract the event flow function from the event occurrence function as follows: $E(t) = \max\{i \in \mathbb{N} \mid T(i) \leq t\}$. The fact that it is possible to extract E from T and vice versa implies that these functions have a pseudo-inverse relation, i.e. $T(i) = E^{\underline{-1}}(i)$ and $E(t) = T^{\overline{-1}}(t)$ (see Definition 5).

Now, the event occurence function T allows us to reason about the distance between consecutive events. For example, the distance between two consecutive events i^{th} and $i+1^{th}$ is $T(i+1) - T(i)$. Similarly, the distance between n consecutive events i^{th} and $i+n-1^{th}$ is: $T(i+n-1) - T(i)$. Given this, the pair $\underline{\delta}, \overline{\delta}$ can be formally introduced.

Definition 7 (Event distance)**.** *Let T be an event occurence function. An* event distance pair *is a pair of functions $\underline{\delta}, \overline{\delta} : \mathbb{N} \to \mathbb{R}^+$ such that*

$$\forall i, \forall n > 0 : \underline{\delta}(n) \leq T(i+n-1) - T(i) \leq \overline{\delta}(n)$$

However, $(\underline{\delta}, \overline{\delta})$ cannot be directly identified as an IBP since Definition 7 contains the term $T(i+n-1)$, whereas Definition 4 of an IBP requires $T(i+n)$. This can yet be solved by a substitution $\overline{\delta}_{\text{IBP}}(n) = \overline{\delta}(n-1)$, $\underline{\delta}_{\text{IBP}}(n) = \underline{\delta}(n-1)$. Now we demonstrate that the IBPs on event flow $(\underline{\eta}, \overline{\eta})$ and event occurrence $(\underline{\delta}, \overline{\delta})$ can be extracted from each other.

Theorem 1. *Let E be an event flow and T its event occurrence function. Then, if $(\underline{\eta}, \overline{\eta})$ is an IBP for E, then $(\overline{\eta}^{\overline{-1}}, \underline{\eta}^{\underline{-1}})$ is an IBP for T, and $(\overline{\eta}^{\overline{-1}}(n+1), \underline{\eta}^{\underline{-1}}(n+1))$ is an event*

distance pair. Conversely, if $(\underline{\delta}, \overline{\delta})$ is an event distance pair, then $((\overline{\delta}_{IBP})^{\overline{-1}}, (\underline{\delta}_{IBP})^{\underline{-1}})$ is an IBP for E.

Proof. Going from $(\underline{\eta}, \overline{\eta})$ to $(\overline{\eta}^{\overline{-1}}, \underline{\eta}^{\underline{-1}})$ is a direct application of Proposition 5 in [3]. We cannot use it for the other way since the domain of event distance is the set of natural numbers whereas the proof of Boyer et al. [3] assumes that the domain is the set of real numbers. Nevertheless, we just have to conduct the same proof with natural numbers.

Let $t, d \in \mathbb{R}^+$. Notice that $E(t) = \sup\{i \mid T(i) \leq t\} = \max\{i \mid T(i) \leq t\} = \min\{i \mid T(i+1) > t\}$. Let $I = \{i \mid T(i+1) > t\}$, $E(t) = \min I$. The same way, with $J = \{j \mid T(j) \leq t + d\}$, $E(t+d) = \max J$.

Let be any $(i, j) \in I \times J$: $T(j) - T(i+1) < (t+d) - t = d$. And by definition of $\underline{\delta}$, $\underline{\delta}_{\text{IBP}}(j-i-1) \leq T(j) - T(i+1)$, leading to $\underline{\delta}_{\text{IBP}}(j-i-1) < d$, which yields $j - i - 1 < \underline{\delta}_{\text{IBP}}^{\underline{-1}}(d)$ and since $(E(t+d), E(t)) \in I \times J$, $E(t+d) - E(t) - 1 < \underline{\delta}_{\text{IBP}}^{\underline{-1}}(d)$, so $E(t+d) - E(t) \leq \underline{\delta}_{\text{IBP}}^{\underline{-1}}(d)$. The relation $\overline{\delta}_{\text{IBP}}^{\overline{-1}} \leq E(t+d) - E(t)$ is done the same way. □

IV. Principles of CPA

CPA [10] is a compositional analysis framework, that analyzes the timing behavior of each component in the system and encodes behavioral dependencies between components by the propagation of event flows. A component can represent (1) a task or (2) a join or fork of event flows where the join/fork operation adheres to system-specific rules. In the following, we present how CPA computes the input-output relations for a task. A task τ_i is an entity, which takes a flow of events as input $\underline{\eta}_i, \overline{\eta}_i$ and produces a flow of output events $\underline{\eta}'_i, \overline{\eta}'_i$ as illustrated in Figure 2. Further task parameters comprise the max. amount of data to be processed per job p_i^+ and the priority. CPA computes the following relations for each task

$$\underline{\eta}' = f_1(\underline{\eta}, \overline{\eta}) \qquad \overline{\eta}' = f_2(\underline{\eta}, \overline{\eta}) \tag{1}$$

According to the CPA algorithm at the component level [10], the IBP $\underline{\eta}', \overline{\eta}'$ is derived in a 3-step procedure which we briefly reproduce:

Step 1 The best case response time (BCRT) R^- and the WCRT R^+ of the task are obtained based on the busy window approach. The busy window approach was first introduced in [11]. It was later generalized to arbitrary activation event flows and the computation of an output event flow was added for the composition of components [6] [12].

Definition 8 (Level-i busy window [11])**.** *A level-i busy window is a time interval $[t, t + \Delta t]$ within which jobs of priority i or higher are processed but no jobs of level i or higher are processed in $(t - \epsilon, t)$ or $(t + \Delta t, t + \Delta t + \epsilon)$ for sufficiently small ϵ. The longest level-i busy window comprises q_i^+ jobs of task τ_i.*

The processing duration of q consecutive jobs inside a level-i busy window can be bounded by the maximum and minimum multiple activation processing time.

Definition 9 (Multiple activation processing time [12])**.** *The maximum q-activation processing times $B_i^+(q)$ returns an upper bound on the time interval between the arrival of the first activation and the termination of the q-th activation in any level-i busy window, where $1 \leq q \leq q_i^+$.*

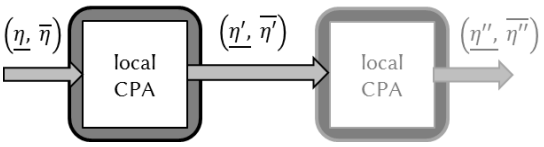


Fig. 2: CPA component interface

The maximum processing times $B_i^+(q)$ are computed by the fixed-point equation

$$\underline{\beta}(B_i^+(q)) = \overline{\alpha}_{iBW}(B_i^+(q)). \tag{2}$$

The fixed point problem converges if the minimum service available in $B_i^+(q)$ ($\underline{\beta}(B_i^+(q))$) corresponds to the maximum workload to be processed until the qth job of task τ_i terminates ($\overline{\alpha}_{iBW}(B_i^+(q))$). The workload $\overline{\alpha}_{iBW}(B_i^+(q))$ depends on the maximum workload of q jobs of task τ_i, and the maximum workload of each task τ_j that may delay the execution or preempt task τ_i in $B_i^+(q)$. We denote the set of tasks which may delay/preempt task τ_i as Γ_i.

$$\overline{\alpha}_{iBW}(B_i^+(q)) = q \cdot p_i^+ + f(\sum_{\tau_j \in \Gamma_i} p_j^+ \cdot \overline{\eta}_j(B_i^+(q))) \tag{3}$$

It is known that in the longest level-i busy window, which corresponds to $B_i^+(q_i^+)$, the WCRT of task τ_i, denoted as R_i^+, is contained. To derive R_i^+, the maximum response time of every job q of task τ_i, denoted as $RT_i^+(q)$, in longest level-i busy period is computed.

$$R_i^+ = \max_{1 \leq q \leq q_i^+} \left\{RT_i^+(q)\right\} = \max_{1 \leq q \leq q_i^+} \left\{B_i^+(q) - \delta_i^-(q)\right\} \tag{4}$$

For the BCRT, the lower bound $R_i^- = C_i^-$ can be applied.

Step 2 It is known that the maximum response time jitter

$$J^+ = R^+ - R^- \tag{5}$$

determines the transformation of the input event flow [6]: In the best case, the distance between events grows at most by the jitter J^+. In the worst case, the density of events increases at most by the maximum response time jitter of the task.

$$\overline{\delta}'_e(n) = \overline{\delta}_e(n) + J^+ \qquad \underline{\delta}'_e(n) = \underline{\delta}_e(n) - J^+ \tag{6}$$

Step 3 Finally, the IBP $\overline{\delta}', \underline{\delta}'$ is converted back to $\underline{\eta}, \overline{\eta}$ as defined in [13].

V. Motivating The Dual Model in CPA

As already discussed, CPA models service request by lower and upper bounds on event arrival encoded by the IBP $\underline{\eta}, \overline{\eta}$. Conversion rules from the event domain to the workload domain are known from the unified model [14]

$$\underline{\alpha}(\Delta t) = (\underline{\pi}^{\overline{-1}} \circ \underline{\eta})(\Delta t) \qquad \overline{\alpha}(\Delta t) = (\overline{\pi}^{\underline{-1}} \circ \overline{\eta})(\Delta t) \tag{7}$$

A step beyond conversion is to integrate both dimensions – event and workload – into the analysis framework, creating

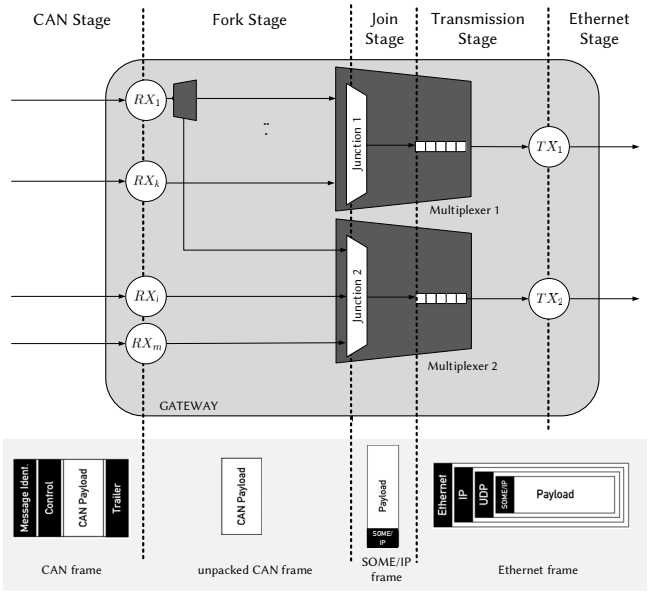


Fig. 3: CAN-to-Ethernet Gateway

a dual model. For now, we propose to complement the input model of CPA with the IBP $\underline{\pi}^{-1}, \overline{\pi}^{-1}$ as illustrated in Figure 4a. We call this enhanced version *CPA+*; and it will be discussed in more detail in Section VI. In the following, we present a practically relevant use case which shows the modeling limitations of the traditional versions of CPA.

A. Use Case

1) CAN-to-Ethernet Gateway: For over 30 years, the message-based Controller Area Network (CAN) protocol has been the cornerstone of automotive communication networks. CAN is still dominant w.r.t. many vehicle functions, but in recent years Ethernet has been adapted as a new automotive network protocol given its high bandwidth, flexibility and competitive cost. The exchange of data between different network technologies is realized by gateways, we focus here on CAN-to-Ethernet gateways as modeled in [15]. CAN frames have small payload sizes in comparison to Ethernet frames, while the protocol overhead of Ethernet frames is much higher than that of CAN frames. It is therefore reasonable to buffer a number of incoming CAN frames, and aggregate their payload in a common Ethernet frame. Buffering is assumed as lossless, i.e., CAN frames are queued until their collective dispatch in a common Ethernet frame. Apart from a good payload/overhead ratio of packaging, the timeliness of data transmission has to be guaranteed for CAN-to-Ethernet traffic by finding triggering conditions for the dispatch of an Ethernet frame. Different triggering conditions have been proposed in the literature, and a selection of those has been standardized in AUTOSAR. For simplicity, we concentrate here on the dispatch of an Ethernet frame on a buffer-full event, i.e., an Ethernet frame is transmitted once the buffer is full. Let us explain the gateway functionality, which can be divided in several stages, with the help of Figure 3.

CAN stage. A set of CAN message streams represents the input to the gateway. Messages of a stream arrive periodically with jitter. A CAN message stream s_i has a given priority i and the payload of a message varies between 0 to 64 bits. A task RX_i models the service that is required to unpack and process a CAN message of stream s_i.

Fork stage. In the fork stage, the unpacked CAN payload is forwarded to one or multiple receivers. A receiver may request the entire payload or only a fragment.

Join stage. Each unpacked CAN frame which arrives at a multiplexer is equipped with a SOME/IP header. The junction then combines the CAN message streams of the fork stage with added SOME/IP headers to a single stream.

Transmission stage. The frames of a stream of the join stage are queued in a buffer according to the FIFO principle. The content of the buffer is dispatched at the output if a a buffer-full event occurs.

Ethernet stage. A task TX_i models the execution time of the Ethernet stack and the time that is required to transmit an Ethernet frame.

2) Benefits of the Dual Model for the CAN-to-Ethernet Gateway: In this section, we discuss the benefits gained in timing verification by applying the dual model to the CAN-to-Ethernet gateway.

CAN stage and Fork stage. For the event-based CPA, complementing the input model with the IBPs $\underline{\pi}, \overline{\pi}$ allows to model the variation of CAN payloads in a stream as well as the effects of subsequent splitting of CAN payload in the fork stage. CPA+ brings here real benefit compared to a reasoning based on minimum and maximum CAN payload sizes.

Transmission stage. A major benefit of the dual model is related to the transmission stage. One important problem in transmission stage analysis is to calculate how many buffer-full events may occur at most in any given time interval of size Δt. Let us assume, that the organization of the buffer is such that frames are placed in a contiguous memory. In CPA, the amount of required bit storage in Δt is then pessimistically given by *max. payload* $\cdot\ \overline{\eta}(\Delta t)$. CPA+ improves considerably by deriving the tighter bound $\overline{\alpha}(\Delta t) = (\overline{\pi}^{-1} \circ \overline{\eta})(\Delta t)$, provided that packet sizes are variable. The maximum number of buffer-full events in Δt is then given by the number of required bit storage divided by the buffer size.

Ethernet stage. If the transmission stage is accurately modeled by CPA+, then the computed upper bound on the service demand w.r.t. Ethernet network is tighter compared to the results provided by NC and CPA. This is because fewer buffer-full events are predicted.

B. Scenarios in Which the Dual Model is Useful

In general, the dual model is useful in two types of scenarios: Firstly, if the system to be analyzed has heterogeneous components and the behavior of some is event-dependent

while others have a more workload-dependent behavior. Secondly, if both the number of occurred events and the amount of data to be processed are important parameters for scheduling decisions or buffer allocation. Apart from the gateway scenario, we would like to give some further examples:

(1) Round-robin scheduling relies on per-packet-decisions for arbitration (event), while the size of the packets (workload) determines for how long the processor/network is occupied.

(2) Packetizers can have complicated rules, when to empty the buffer and dispatch a packet. Often, such rules are a trade-off between having a good protocol/payload ratio and high responsiveness. Fill level thresholds of a buffer (workload) are then combined with trigger events (event) or timeouts (event).

(3) Buffer analysis depends much on how the storage is organized. The assignment of memory slots per-frame (event) and per-byte (workload) or combinations thereof are possible decisions.

VI. Introducing the Dual Model and Analysis: CPA+

In the previous section, we had a preliminary concept of CPA+. We proposed to complement the input model of CPA with the IBP $\underline{\pi}^{-1}, \overline{\pi}^{-1}$. We now discuss and improve over this choice, and touch upon more subtle details of the dual model.

A. Can the full dual information in CPA be fully recovered by the IBP of the packet function?

While it is possible to construct the full information on the event flow IBP $(\underline{\eta}, \overline{\eta})$ on the basis of the data flow IBP $(\underline{\alpha}, \overline{\alpha})$ and the packet function IBP $(\underline{\pi}, \overline{\pi})$ [14]

$$\underline{\eta}(\Delta t) = (\overline{\pi} \circ \underline{\alpha})(\Delta t) \qquad \overline{\eta}(\Delta t) = (\underline{\pi} \circ \overline{\alpha})(\Delta t), \quad (8)$$

this is not true for the reverse direction of transformation as needed in CPA. The reason is that neither the event flow IBP $(\underline{\eta}, \overline{\eta})$ nor the packet function IBP $(\underline{\pi}, \overline{\pi})$ contains information on the bit arrival rate of packets. As a consequence, the extremes of bit arrival rates (instantaneous arrival and maximal delay) are considered when computing the best case and worst case of workload in form of the IBP $\underline{\alpha}, \overline{\alpha}$ [14]

$$\underline{\alpha}(\Delta t) = (\overline{\underline{\pi}^{-1}} \circ \underline{\eta})(\Delta t) \quad \overline{\alpha}(\Delta t) = (\overline{\pi}^{-1} \circ \overline{\eta})(\Delta t). \quad (9)$$

Thus the IBP $\underline{\alpha}, \overline{\alpha}$ is safe but potentially pessimistic.

B. Is a triple interface of a data flow IBP, event flow IBP and packet function IBP adequate?

As a consequence of the discussion in the previous section, it seems insufficient to complement the input model of CPA with the IBP $\underline{\pi}^{-1}, \overline{\pi}^{-1}$, if the full information on bit arrival rate should be preserved. A solution is to define the triple IBPs $(\underline{\eta}, \overline{\eta})$, $(\underline{\alpha}, \overline{\alpha})$, $(\overline{\underline{\pi}^{-1}}, \overline{\pi}^{-1})$ as the extended input interface for CPA+. This goes beyond the CPA+ interface proposed in Section V which only adds the IBP $(\overline{\underline{\pi}^{-1}}, \overline{\pi}^{-1})$.

Another important argument for this interface of three IBPs is the the issue of repeated transformations. A transformation from the event domain to the workload domain and back entails information loss, this has been observed in [16]. The effect can be illustrated by the following equation

$$\overline{\eta}(\Delta t) = (\underline{\pi} \circ \overline{\pi}^{-1} \circ \overline{\eta})(\Delta t). \quad (10)$$

The key problem here is as follows. We begin with the transformation from the event domain to the workload domain. The maximum data flow is obtained in the scenario, in which maximum packet size is combined with maximum event arrival as expressed by $\overline{\alpha}(\Delta t) = (\overline{\pi}^{-1} \circ \overline{\eta})(\Delta t)$. The transformation back to the event domain, namely $\overline{\eta}(\Delta t) = (\underline{\pi} \circ \overline{\alpha})(\Delta t)$ relies on the scenario that minimum packet sizes coincide with maximum data flow, creating a maximum of event arrivals. The different assumptions on packet sizes leads to extreme pessimism in the composition of the two rules as in Eq. 10 [16]. The problem of repeated transformation is thus a strong argument to use the triple IBPs interface to avoid multiple chained transformations.

C. Improved CPA+ interface and its composition

Figure 4b shows the improved interface of a task in CPA+ and how it can be connected to another task, if their behaviors are correlated. We assume that the external input provided to the first task is specified by $(\underline{\eta}, \overline{\eta})$ and $(\underline{\alpha}, \overline{\alpha})$. Since there is no a priori information which of the two inputs is the more accurate one, a pointwise minimum w.r.t. the upper bounds $\overline{\pi}^{-1} \circ \overline{\eta}$, $\overline{\alpha}$ and a pointwise maximum w.r.t. the lower bounds $\overline{\underline{\pi}^{-1}} \circ \underline{\eta}$, $\underline{\alpha}$ is performed in the workload domain. The CPA+ interface then provides also a dual output namely $(\underline{\eta}', \overline{\eta}')$ and $(\underline{\alpha}', \overline{\alpha}')$. If the outputs are independent of each other, so that $(\underline{\alpha}', \overline{\alpha}')$ is not the result of a transformation of $(\underline{\eta}', \overline{\eta}')$, it is not known which of the bounds is tighter and a pointwise minimum/maximum is again applied for comparison. To achieve some independence between outputs, we propose how to compute the IBP of an output data flow – this has not been part of CPA before. An important point of the proposed CPA+ interface is that it excludes multiple chained transformations of a flow into the event/data domain and back can occur. This is achieved by performing the comparison of bounds only in the workload domain.

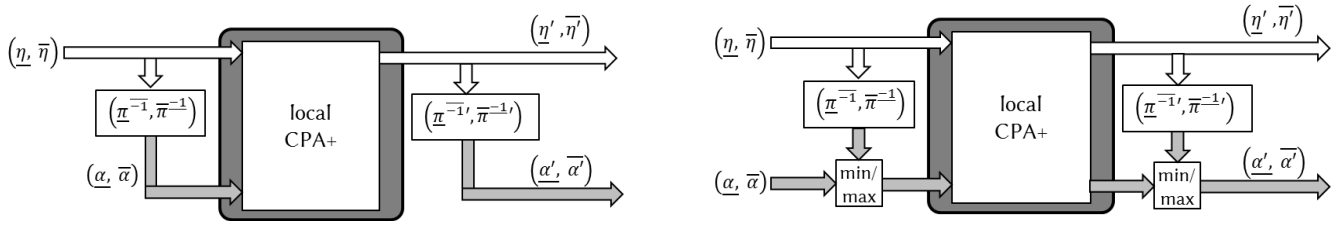
D. Impact of the CPA+ interface on the analysis

CPA+ has thus a dual input to the component covering both the event and workload domain, and in the following we discuss how the analysis can profit from the enriched input.

The above gateway example has illustrated that CPA+ is helpful to accurately model complex fork, join as well as buffering operations. With respect to tasks, we want now to show formally how and why CPA+ is beneficial. First of all, CPA+ derives a tighter upper bound for $B_i^+(q)$ than CPA.

Theorem 2. *The CPA+ interface modifies the busy window equation* $\underline{\beta}(B_i^+(q)) = \overline{\alpha}_{iBW}(B_i^+(q))$ *such that* $\overline{\alpha}_{iBW}(B_i^+(q))$ *is given by*

$$\overline{\alpha}_{iBW}(B_i^+(q)) = \overline{\pi}_i^{-1}(q) + f\Big(\sum_{\tau_j \in \Gamma_i} \min\{\overline{\alpha}_j(B_i^+(q)),\ (\overline{\pi}_j^{-1} \circ \overline{\eta}_j)(B_i^+(q))\}\Big).$$



(a) Preliminary CPA+ interface

(b) Improved CPA+ interface

Fig. 4: Versions of CPA+ interfaces

Proof. In Eq. 3, we replace the weighting approach by the application of the packet function, so that

$$q \cdot p_i^+ \rightarrow \overline{\pi}_j^{-1}(q)$$
$$p_j^+ \cdot \overline{\eta}_j(B_i^+(q)) \rightarrow (\overline{\pi}_j^{-1} \circ \overline{\eta}_j)(B_i^+(q)).$$

Moreover, we take into account by the minimum function that the second input $\overline{\alpha}_j(B_i^+(q))$ to the CPA component might be a tighter bound than $(\overline{\pi}_j^{-1} \circ \overline{\eta}_j)(B_i^+(q))$. □

The fixed point equation of Theorem 2 delivers a better bound $B_i^+(q)$ than the standard CPA version in Eq.s 2-3. This tighter bound on $B_i^+(q)$ results in a smaller WCRT and consequently in a smaller maximum response time jitter J^+ (see Eq.s 4, 5). A second positive result is that the IBP of output event distance functions is tighter (see Eq. 6). Also, the IBP of the output data flow, which can be computed according to Theorem 3, benefits from a smaller J^+. Achieving tighter bounds on event and data flows at each output stage is extremely important for multi-component scheduling problems, where pessimism accumulates across components and may dominate results.

Theorem 3. *The IBP of the output data flow is given by*

$$\underline{\alpha}'(\Delta t) = \underline{\alpha}(\max\{\Delta t - J^+, 0\}) \quad (11)$$
$$\overline{\alpha}'(\Delta t) = \overline{\alpha}(\Delta t + J^+). \quad (12)$$

Proof. Let $d^{\min}$ be the minimal delay, and $d^{\min} + J^+$ the maximal delay. Then $A(t+d^{\min}) \leq D(t) \leq A(t+d^{\min}+J^+)$, then $D(t+\Delta t)-D(t) \leq A(t+\Delta t+d^{\min}+J^+)-A(t+d^{\min}) \leq \overline{\alpha}(\Delta t+J^+)$. Conversely, $D(t+\Delta t)-D(t) \geq \underline{\alpha}(\Delta t-J^+)$. □

VII. Conclusion

In this paper, we introduced an extended version of CPA called CPA+. The new framework features a true dual component model, where the service request at the input/output is represented by lower and upper bounds on (i) the data flow, (ii) the event flow, and (iii) the relating packet function. However, CPA+ does not only provide a new component interface, but it also defines algorithms how to compute the new outputs. Composition rules of CPA+ are such that the dual information is propagated between components and transformation losses are prevented. Moreover, we identified a set of practical scenarios where a dual model is beneficial, and elaborated on a CAN-to-Ethernet gateway example in more detail. As future work, we plan to conduct an experimental evaluation, so as to quantitatively assess the benefits of CPA+. Moreover, in terms of analysis frameworks, one step beyond is to realize a collaboration between CPA+ and NC, given that both frameworks rely on the common input model [3].

Acknowledgment

This work has been partially funded by the ANR-DFG project RT-proofs (ANR-17-CE25-0016) and by the DFG project TypicalCPA (168/30-2).

References

[1] E. Wandeler and L. Thiele, "Characterizing workload correlations in multi processor hard real-time systems." in *IEEE Real-Time and Embedded Technology and Applications Symposium*, 2005, pp. 46–55.
[2] A. Bouillard, N. Farhi, and B. Gaujal, "Packetization and aggregate scheduling," Ph.D. dissertation, INRIA, 2011.
[3] M. Boyer and P. Roux, "Embedding network calculus and event stream theory in a common model," in *Proc. of the 21st IEEE Int. Conf. on Emerging Technologies and Factory Automation*, 2016.
[4] L. Ahrendts, S. Quinton, and R. Ernst, "Exploiting execution dynamics in timing analysis using job sequences," *IEEE Design & Test*, 2017.
[5] J.-Y. Le Boudec and P. Thiran, *Network calculus: a theory of deterministic queuing systems for the internet*. Springer, 2001, vol. 2050.
[6] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, "System level performance analysis–the SymTA/S approach," *IEE Proc. - Computers and Digital Techniques*, vol. 152, no. 2, pp. 148–166, 2005.
[7] S. Perathoner, E. Wandeler, L. Thiele, A. Hamann, S. Schliecker, R. Henia, R. Racu, R. Ernst, and M. G. Harbour, "Influence of different abstractions on the performance analysis of distributed hard real-time systems," *Design Automation for Embedded Systems*, vol. 13, no. 1-2, pp. 27–49, 2009.
[8] L. Thiele, S. Chakraborty, and M. Naedele, "Real-time calculus for scheduling hard real-time systems," in *ISCAS 2000*, vol. 4. IEEE, 2000, pp. 101–104.
[9] A. Bouillard, M. Boyer, and E. Le Corronc, *Deterministic Network Calculus: From Theory to Practical Implementation*. Wiley, 2018.
[10] R. Hofmann, L. Ahrendts, and R. Ernst, "Cpa: Compositional performance analysis," *Handbook of Hardware/Software Codesign*, pp. 721–751, 2017.
[11] J. P. Lehoczky, "Fixed priority scheduling of periodic task sets with arbitrary deadlines," in *Real-Time Systems Symposium, 1990. Proceedings., 11th*. IEEE, 1990, pp. 201–209.
[12] S. Schliecker, J. Rox, M. Ivers, and R. Ernst, "Providing accurate event models for the analysis of heterogeneous multiprocessor systems," in *Proc. of the 6th IEEE/ACM/IFIP Int. Conf. on Hardware/Software Codesign and System Synthesis*. ACM, 2008, pp. 185–190.
[13] J. Diemer, *Predictable Architecture and Performance Analysis for General-Purpose Networks-on-Chip*. Verlag Dr. Hut, 2016.
[14] M. Boyer and P. Roux, "A common framework embedding network calculus and event stream theory," 2016.
[15] D. Thiele, J. Schlatow, P. Axer, and R. Ernst, "Formal timing analysis of can-to-ethernet gateway strategies in automotive networks," *Real-time systems*, vol. 52, no. 1, pp. 88–112, 2016.
[16] E. Wandeler, *Modular performance analysis and interface-based design for embedded real-time systems*. ETH Zurich, 2006.