Granularity-based Interfacing between RTC and Timed Automata Performance Models

Yanhong Liu, Karine Altisen, Matthieu Moy

Abstract: To analyze the complex and heterogenous real-time embedded systems, recent works have proposed interface techniques between *real-time calculus* (RTC) and *timed automata* (TA), in order to take advantage of the strengths of each technique for analyzing various components. But the time to analyze a state-based component modeled by TA may be prohibitively high, due to the *state space explosion* problem. In this paper, we propose a framework of granularity-based interfacing to speed up the analysis of a TA modeled component. First, we abstract the fine models to work with event streams at coarse granularities. Then based on the RTC theory, we develop a formal mathematical algorithm to derive lower and upper bounds on the arrival patterns of the fine output streams, by analyzing the component for multiple runs at coarse granularities. These bounds may be tighter than those simply implied by the results analyzed from the abstracted models. Our framework can help to achieve the tradeoffs between the precision and the analysis time.

1 Introduction

Modern real-time embedded systems are increasingly complex and heterogenous. They may be composed of various subsystems such as hard-disk drives, displays, radio-frequency communication devices and computational elements like general-purpose configurable processors, DSPs or FPGAs etc. It is a general practice that some of the subsystems may be power-managed [1, 2], in order to reduce energy consumption and to extend the system life time. Such a subsystem may have multiple running modes, while a mode with lower power consumption also implies lower performance levels. Due to the real-time requirements, it is critical to analyze the timing performance of such systems. However, it is challenging to manage the complexity of such analysis, especially when it is scaled to large and heterogenous systems.

1.1 Related Work

Compositional analysis techniques have been presented as a way of tackling the complexity of accurate performance evaluation of large real-time embedded systems. The examples include the SymTA/S (Symbolic Timing Analysis for Systems) [3] and modular performance analysis with real-time calculus (RTC) [4]. Various models and formalisms have also been proposed to specify and analyze heterogenous components [5]. Different analysis techniques have their own particular strengths and weaknesses. For example, SymTA/S or RTC based analysis can provide hard lower/upper bounds for the best-/worst-case performance of a system, and has the advantage of short analysis time. But typically they are not able to model complex interactions and state-dependent behavior and can only give very pessimistic results for such a system. On the other hand, state-based techniques, e.g. timed automata (TA) [6], construct a model that is more accurate, and can determine the exact best-/worst-case results. But they face the *state space explosion* problem and can possess prohibitively high analysis time even for a system with reasonable size.

Efforts have been paid to couple heterogenous approaches [7, 8, 9], e.g. to combine the functional RTC-based analysis with state-based techniques. A model, called *multi-mode RTC* [10], extends the RTC framework to include state information. In this new framework, system properties within a single mode is analyzed using the RTC-based technique and state-space exploration is used to piece together the results for the individual modes. In [11], an interfacing technique is proposed to compose RTC-based techniques with state-based analysis methods, by transforming between RTC models and a state-based model called *event count automata* [12]. A tool called CATS [13] is just at the beginning of its development, which abstracts TA as a transducer of abstract streams described by RTC.

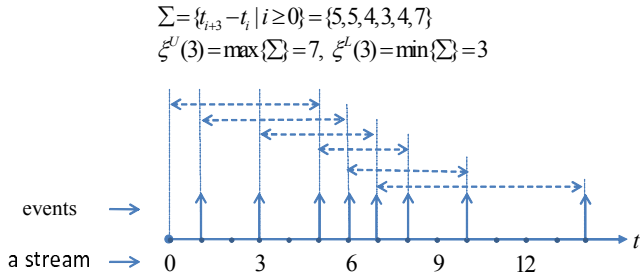

Figure 1: Illustration of the definition of an arrival curve ξ.

1.2 Performance Models: RTC and TA

In this paper, we follow the line of recent development in interfacing between RTC and state-based models, and focus on speeding up the analysis for a power-managed component (PMC) [14] modeled by TA. In the following, we first describe the basic concept of RTC performance model and a state-based one TA.

RTC: Suppose that an event stream is processed by a component. The arrival patterns for all possible streams that can be input to the component are described by an *arrival curve* $\xi = (\xi^L(k), \xi^U(k))$ for $k \geq 0$. Defined for a class of streams, $\xi^L(k)$ and $\xi^U(k)$ respectively provide for any potential stream the lower and upper bounds on the length of the time interval during which any k consecutive events can arrive. Let t_i denote the arrival time of the i-th event, then we have $\xi^L(k) \leq t_{i+k} - t_i \leq \xi^U(k)$ for all $i \geq 0$ and $k \geq 0$. Figure 1 shows an example stream and the definition of $\xi^L(3)$ and $\xi^U(3)$. Σ records the set of inter-arrival times between any 3 consecutive events. The minimum and maximum of all elements in Σ are assigned to $\xi^L(3$ and $\xi^U(3)$ respectively. Similarly, the processing capacity of a component can be specified by a *service curve* $\psi(k) = (\psi^L(k), \psi^U(k))$. The length of the time to process any k consecutive events for any potential stream is at least $\psi^L(k)$ and at most $\psi^U(k)$. The process of events can be at the unit of processing cycle (computation) or bit (communication), which can be transformed into the same unit as the arrival curve.

In the RTC theory, an arrival curve $\alpha = (\alpha^L(\Delta), \alpha^U(\Delta))$ (respectively, service curve β) is usually expressed in terms of *number of events*, which provides lower and upper bounds on the number of events that can arrive (respectively, be processed) within any time interval of length Δ. In this paper, we express the arrival curves (ξ) and service curves (ψ) in terms of *length of time interval*, in order to better explain our work. Actually, ξ is a *pseudo-inverse* of α, satisfying $\xi^U(k) = \min_{\Delta \geq 0}\{\Delta | \alpha^L(\Delta) \geq k\}$ and $\xi^L(k) = \max_{\Delta \geq 0}\{\Delta | \alpha^U(\Delta) \leq k\}$. Throughout the paper, any curve $F(x)$ is assumed to be wide-sense increasing, meaning that $F(x_1) \leq F(x_2)$ for $x_1 \leq x_2$ and $F(x) = 0$ for $x \leq 0$.

TA: A timed automaton is a finite-state machine extended with *clock* variables. A clock measures the time elapsed since its last reset. Figure 4 shows an example of a timed automaton. A node represents a *state*, and an edge represents a transition from one state to the other. An *invariant* (*guard*) may be put on a state (transition), which is a conjunction of upper (lower and upper) bounds on clocks, differences between clocks or integer expressions. The automaton can only stay in a state when the invariant is evaluated to be true. The transition can only happen when the guard is evaluated to be true. The invariant and guard together determine the bounds on when a transition can be taken. As shown on the transition from "Lower" to "Upper", a timed automaton synchronizes with another via sending a signal **m**! or receiving **m**? on a transition (e.g. **signal**!). After a transition is taken, the values of clocks or variables can be updated (e.g. $v \leftarrow 0$). The automaton immediately passes a *committed* state (e.g. "Lower"), with no time elapsed.

1.3 Our Contribution

A PMC interacts with other components or environments via data streams of events. In this paper, we study the case that the PMC processes a single input stream. It is straightforward to apply our results to the case of multiple streams, which can be merged into one [15] or treated separately, depending on the semantics of the PMC. The processing semantics of the PMC is modeled by a *processing element* (PE), as shown in Figure 2. We can cluster all the states of the PE modeled by TA into multiple operation modes, based on the

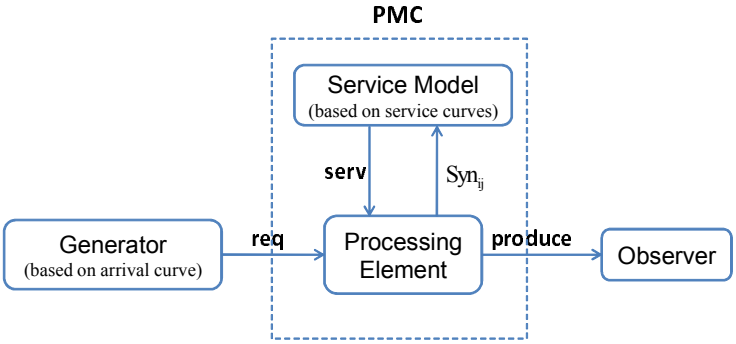

Figure 2: Abstracted models for a PMC and its interfaces.

processing capability (and/or power) of each mode. Associated with each mode, a service curve specifies the lower and upper bound on the time patterns following which the stream of events are processed in that mode. At an abstract level, the PE selects its running mode, based on the the number of buffered events waiting for process. In the PMC, we also include a *service model*, which generates a sequence of signals 'serv!' to indicate when the process of each event is completed, based on the specification of service curves and the running mode. The arrival patterns of all the potential streams at an input to a component are characterized by an arrival curve. The *generator*, generates a sequence of signals 'req!' to indicate when each input event arrives. At the output of the PMC, the signals 'produce?' are recorded by the *observer*, which can be analyzed to obtain an output arrival curve in order to couple with other components.

The generator (or service model) uses the clock variables to encode the arrival curve (or service curve). The number of clocks is equal to the length of the curve. Hence, to analyze the TA-modeled PMC, the size of the state space is dependent on the length of the curves and also the number of events to be explored. We can abstract the granularity of the event stream into a coarse level, i.e. multiple fine events are regarded as one coarse event. Intuitively, such abstraction would reduce the length of the curves and the number of events to be explored, which then reduce the size of the state space and the analysis time for the PMC. In this paper, we propose a formal framework of granularity-based interfacing between RTC and TA performance models. The generator, PMC and observer deals with coarse events, which speeds up the analysis of the PMC.

In the framework, the TA models of the PMC have to be abstracted to deal with the coarse events. However, it is not straightforward to guarantee that the abstraction is accurate, in the sense that the lower and upper coarse output curves (analyzed from the abstracted coarse PMC model) always provide lower and upper bounds on the lower and upper fine curves (analyzed from the original fine PMC model) respectively. An analyzed coarse output curve already implicitly provides lower and upper bounds on the arrival patterns of the output stream at the fine granularity. But we find that it is possible to derive tighter fine output curve, using the resulted coarse output curves from multiple runs of analyzing the component at different coarse granularity levels, based on the properties of RTC curves. In the proposed interfacing framework, as shown in Figure 3, we first vary the TA models for a PMC such that the models at a coarse granularity are guaranteed to be accurate abstraction of those at the fine granularity. Then we conduct the analysis of the TA models for multiple runs with different granularity levels ($g_1 ... g_m$). Finally, we apply a formal mathematical algorithm to refine the multiple coarse output arrival curves ($\check{\xi}_{g_1} ... \check{\xi}_{g_m}$) to tighter output arrival curve ($\tilde{\xi}$) at the fine granularity. Our formal algorithm guarantees that the derived fine output curve still provide valid lower and upper bounds on the output patterns of events processed from the PMC. To the best of our knowledge, no existing work has worked on a granularity-based scheme with formal validation on the abstraction. The proposed framework complements the recent works on interfacing between RTC and TA models.

Organization of the paper: In the next section, we detail the implementation of the existing interfacing techniques between RTC and TA, and propose our framework of granularity-based interfacing. Then in section 3, we discuss our targeted PMC, and show its fine and coarse TA models. In section 4 we illustrate with an example PMC, which is followed by experimental validation in section 5. Finally, we make a

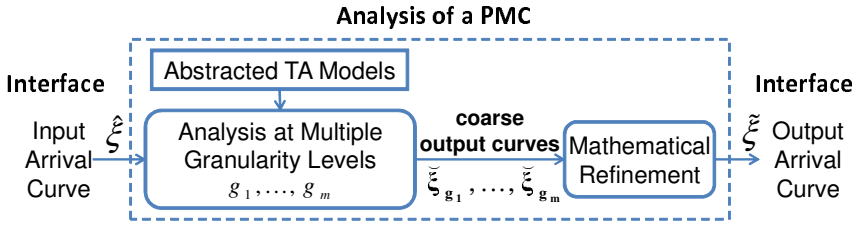

Figure 3: The framework of granularity-based interfacing.

conclusion and present the future work in section 6.

2 Granularity-based Interfacing

2.1 Interfacing between RTC and TA Models

Figure 2 shows the abstracted models for a PMC and its interfaces with RTC curves. As discussed above, an arrival curve $\hat{\xi}$ is specified to bound the arrival patterns of the input stream into a PMC. A PE (processing element), modeled by TA, gives the processing semantics of the PMC. A service curve is associated with each running mode of the PE, to specify the service time patterns for the stream at that mode. The arrival patterns of the output event stream are captured by an output arrival curve $\check{\xi}$, in order to couple with other components. Techniques have been proposed to interface between RTC and TA models. In Figure 2, the *generator* emits a sequence of signals 'req' based on the specification of arrival curve $\hat{\xi}$, indicating the arrival of an event. The *service model* has the same number of modes as the PE and keeps staying in the same mode as the PE. Whenever the PE switches from mode M_i to M_j, the service model synchronizes its mode with the PE by the signal 'Syn_{ij}'. The service model determines when the process of an event is completed, based on the service curve specified, and notifies the PE with signal 'serv'. Whenever the PE receives a signal 'serv', it emits a signal 'produce', implying that a new event is added to the output stream. The *observer* captures the signals 'produce' and records the output events. The output arrival curve can be obtained by analyzing the TA models of PMC, generator and observer. In the following, we show the techniques from CATS[13] for modeling the generator, service and observer.

2.1.1 From RTC to TA - Converter

Figure 4 shows a TA model of a *Converter(∂^L,∂^U,signal)*, which generates a sequence of signals, the time patterns of which are lower and upper bounded by a pair of RTC curves $(\partial^L(i), \partial^U(i))$ for $1 \leq i \leq N$. A circular clock array $y[0:N-1]$ is defined to record the time patterns between consecutive signals. The function $getIdx(i)$ returns the index of $(\lambda - i + N)\%N$. When a signal is emitted, a clock element $y[\lambda]$ associated with it will be reset to zero. The value of λ is updated by $(\lambda+1)\%N$. Suppose that ε_k $(k > N)$ is the next signal to be generated, $y[getIdx(i)]$ is the clock associated with ε_{k-i} for $1 \leq i \leq N$, which records the time that has elapsed since ε_{k-i} was generated. Based on the definition of a RTC curve, we have $\partial^L(i) \leq y[getIdx(i)] \leq \partial^U(i)$ for $1 \leq i \leq N$.

In the model, time can only progress in "Upper" state, where an invariant specifies that all the upper bounds $\partial^U(i)$ are satisfied. It can randomly transit out from "Upper". In "Lower" and "Temp" states, it checks if all the lower bounds are satisfied by checking how much time has elapsed since the last $(v+1)$-th signal was emitted, where $0 \leq v \leq \min\{N-1, \theta\}$. If less than N signals have been emitted, we do not need to check for all elements of the clock array. θ is used for that purpose. If $y[getIdx(v+1)]$ is less than $\partial^L(v+1)$ for some v, it then returns to "Upper" to wait longer. When all the lower bounds are satisfied, it emits a new signal, resets clock $y[\lambda]$ and continues from state "Upper".

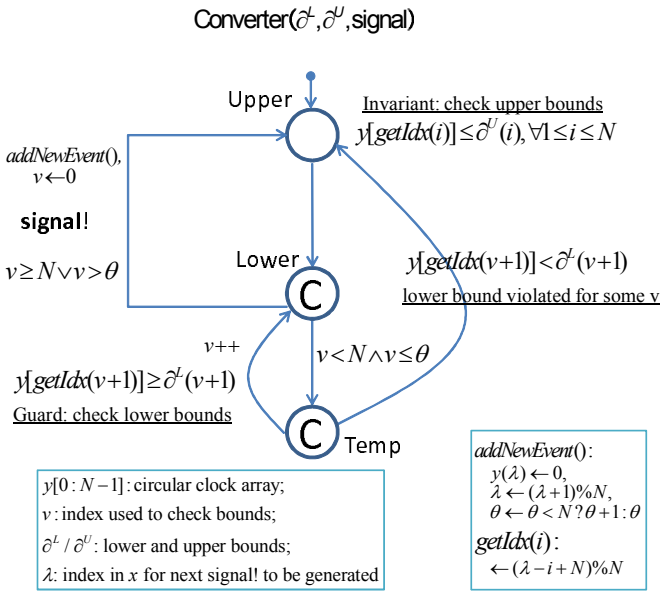

Figure 4: Model of a converter from RTC to TA.

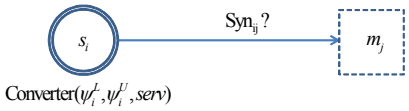

Figure 5: Service model.

2.1.2 Generator

The model of the generator is obtained by instantiating the model shown in Figure 4 with *Converter*($\hat{\xi}^L$,$\hat{\xi}^U$,*req*). The arrival curve $\xi = (\hat{\xi}^L, \hat{\xi}^U)$ specifies the lower and upper bounds on the arrival patterns of the input stream to the PMC.

2.1.3 Service Model

The service model determines when the process of an event is completed. Since a service curve $\psi_i = (\psi_i^L, \psi_i^U)$ is specified for each mode of the PE, the service model has the same number of modes and keeps synchronizing its running mode with that of the PE by a signal Syn_{ij}. As shown in Figure 5, each mode m_i of the service model is instantiated with *Converter*(ψ_i^L,ψ_i^U,*serv*).

2.1.4 Observer

To capture the arrival patterns of the output stream produced from the PMC, a model of the observer is used, as shown in Figure 6 (a). The observer non-deterministically transits from the initial state "Idle" to "Busy". The variable η records the number of events that have been output from the PE since it enters "Busy" state. To compute the output arrival curve, we use a model checking tool with cost optimal reachability analysis. The cost spent in a state is computed to be $rate$ multiplied by the total time spent in that state, where $rate$ is the cost rate specified in the invariant. The cost spent on a transition is assigned with a pre-specified constant; by default, it is equal to zero. We set the $rate$ to 1 for "Busy".

The cost, when reaching a status of "Observer is in "Busy" and $\eta > k$ and $\eta <= k+1$" for $1 \leq k \leq K_{max}$, is equal to the time for the output of any $(k+1)$ consecutive events since the observer enters "Busy" state. By verification, we can compute the minimum (and maximum) cost (i.e. time) for the output of any $(k+1)$ events. Since the $(k+1)$ consecutive events are chosen non-deterministically (randomly entering

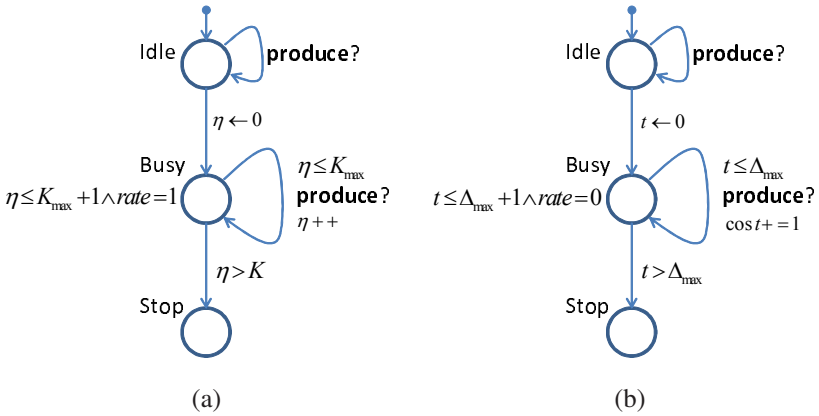

Figure 6: (a) Model of the observer and (b) its varied model to compute $\breve{\alpha}(\Delta)$.

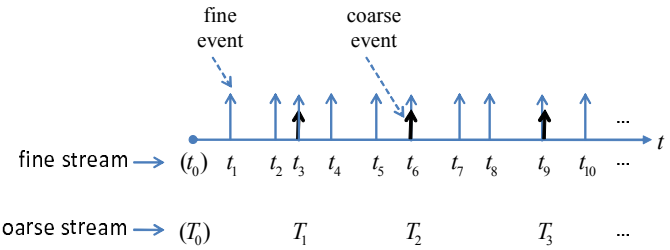

Figure 7: The illustration of a fine stream and its abstraction with $g = 3$.

"Busy" state), the minimum (and maximum) cost computed provides value to $\breve{\xi}^L(k)$ (and $\breve{\xi}^U(k+1)$) for $1 \leq k \leq K_{max}$. Here we also show a varied model of the observer in Figure 6 (b) where t is a clock, which can be used to compute $\breve{\alpha}(\Delta)$ for $0 < \Delta \leq \Delta_{max}$. $\breve{\xi}$ can also be computed to be a pseudo-inverse of $\breve{\alpha}$.

2.2 Framework

2.2.1 Definitions and Notations

In this paper, we denote a specific stream at the fine granularity to be $\tau = (t_0)t_1t_2...$ and a specific coarse stream to be $\top = (T_0)T_1T_2....$ t_i (T_i) denotes the arrival time of the i-th fine (coarse) event ε_i ($\mathcal{E}_i$) for $i \geq 1$. t_0 (T_0) does not indicate a real event, which just records the time when the first fine (coarse) event starts to be generated. We also use $\hat{\tau}$ ($\hat{\top}$) and $\breve{\tau}$ ($\breve{\top}$) to differentiate between an input stream and an output one.

As illustrated in Figure 7, we can abstract a fine stream to a coarse one at some granularity of g, by regarding g consecutive fine events as a coarse one. In this paper, we *abstract* a fine stream τ to a coarse one $\top$ in such a way that T_i is equal to t_{gi} for $i \geq 0$, i.e. sampling the fine stream every g events. We can also *refine* a coarse stream $\top$ at granularity of g to a fine one. We assume that the $(g \times i)$-th fine event arrives at the same time as the i-th coarse event, i.e. $t_{gi} = T_i$ for $i \geq 0$. The arrival time t_j for any other fine event ε_j with $g \times (i-1) < j < g \times i$, can be assigned to any value within $[T_{i-1}, T_i]$. It is easy to see that a coarse stream $\top$ can be refined to a fine stream τ if and only if τ can be abstracted to $\top$.

2.2.2 Validation of Abstraction

Figure 8 shows the analysis models for a PMC, at both fine and coarse granularities. At the input to the PMC, it is specified that the inter-arrival time patterns between fine events are lower and upper bounded by $\hat{\xi}(k)$ for $k \geq 0$. Let us see the fine models first. The generator (see [13, 11]) generates an input stream of events, based on the specification of $\hat{\xi}$. The PMC $\mathcal{P}_0$ processes the input stream of events. Whenever an

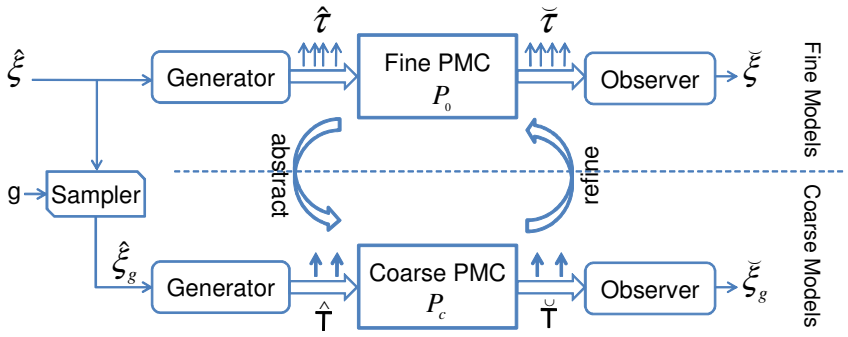

Figure 8: Analysis models of a PMC at both fine and coarse granularities.

event is output from $\mathcal{P}_0$, it is recorded by the observer. Using the model checking tools, the output arrival curve of the outgoing stream can be computed, denoted by $\breve{\xi}$.

The coarse models work with a stream at coarse granularity of g. It is easy to see that the arrival patterns of the input stream $\hat{\top}$ for the coarse PMC $\mathcal{P}_c$ are lower and upper bounded by $\hat{\xi}_g(k) = \hat{\xi}(gk)$ for $k \geq 0$, which is obtained by sampling the arrival curve $\hat{\xi}$. Hence, we add a *sampler* that samples $\hat{\xi}$ with granularity of g and specifies a new arrival curve $\hat{\xi}_g$ for the generator in the coarse models, as shown in Figure 8.

After we abstract the fine input *streams* $\hat{\tau}$ to coarse ones $\hat{\top}$ at granularity of g, it only loses information about the fine events other than ε_{gi}. But to work on the coarse input streams, we also have to abstract the fine *model* $\mathcal{P}_0$ itself, since the exact number of buffered fine events is not known when mode is switched. which introduces more non-determinism in it.

To validate the proposed framework, we have to guarantee that the coarse models provide accurate abstraction of the fine models. Firstly, $\mathcal{P}_c$ should exhibit at least all the behaviors that the fine model $\mathcal{P}_0$ can produce, in the sense of the set of event streams that can be input into and output from the PMC. Formally, $\mathcal{P}_c$ should satisfy,

Proof Obligation 1 *Let $\hat{\tau} = (\hat{t}_0)\hat{t}_1\hat{t}_2...$ ($\breve{\tau} = (\breve{t}_0)\breve{t}_1\breve{t}_2...$) be any fine event stream that is an input to (the corresponding output stream produced from) $\mathcal{P}_0$. We can prove that there always exists some coarse event stream $\hat{\top}$ ($\breve{\top}$) that can be an input to (the corresponding output stream produced from) $\mathcal{P}_c$, such that $\hat{\top}$ ($\breve{\top}$) can be refined to $\hat{\tau}$ ($\breve{\tau}$).*

It is then followed that we can derive valid coarse output curve $\breve{\xi}_g$, by analyzing $\mathcal{P}_c$.

Lemma 1 *If Proof Obligation 1 is satisfied, it is guaranteed that the analyzed $\breve{\xi}_g^L(k)$ and $\breve{\xi}_g^U(k)$ provide lower and upper bounds on $\breve{\xi}^L(gk)$ and $\breve{\xi}^U(gk)$ respectively for $k \geq 0$.*

Proof: It is followed from Proof Obligation 1 that, for any fine output stream $\breve{\tau}$ from $\mathcal{P}_0$, there always exists a coarse output stream $\breve{\top}$ such that $\breve{\top}$ can be refined to $\breve{\tau}$. It follows that the production time of the $(g \times k)$-th fine event in $\breve{\tau}$ is equal to that of the g-th coarse event in $\breve{\top}$. It is then easy to have $\breve{\xi}_g^L(k) \leq \breve{\xi}^L(gK)$ and $\breve{\xi}_g^U(k) \geq \breve{\xi}^U(gk)$. □

2.2.3 Mathematical Refinement Algorithm

After conducting multiple runs of analyzing the coarse PMC at different granularities of $g_1, ..., g_m$, we obtain output arrival curves $\breve{\xi}_{g_i}$ $(i = 1, ..., m)$. From Lemma 1, we have $\breve{\xi}_{g_i}^U(k) \geq \breve{\xi}^U(kg_i)$ and $\breve{\xi}_{g_i}^L(k) \leq \breve{\xi}^L(kg_i)$ for $k \geq 0$. Since $\breve{\xi}^U(k_1) \leq \breve{\xi}^U(k_2)$ with $k_1 \leq k_2$, $\breve{\xi}_{g_i}^U(k)$ provides an upper bound on $\breve{\xi}^U(j)$ for any $j \leq kg_i$. Similarly, $\breve{\xi}_{g_i}^L(k)$ also provides a lower bound on $\breve{\xi}^L(j)$ for any $j \geq kg_i$. Hence, we can compute the fine output curve $\tilde{\xi}$ to be $\tilde{\xi}^U(j) = \min_{k\geq 0}\{\breve{\xi}_{g_i}^U(k) | kg_i \geq j\}$ and $\tilde{\xi}^L(j) = \max_{k\geq 0}\{\breve{\xi}_{g_i}^L(k) | kg_i \leq j\}$. In this section, we propose a formal mathematical algorithm to derive tighter $\tilde{\xi}$ from coarse curves $\breve{\xi}_{g_i}$, but still satisfying $\tilde{\xi}^U(k) \geq \breve{\xi}^U(k)$ and $\tilde{\xi}^L(k) \leq \breve{\xi}^L(k)$. In the following, we show an example to illustrate the basic rationale behind the algorithm.

Suppose that we obtain output curves $\breve{\xi}_2$ and $\breve{\xi}_3$ after analyzing $\mathcal{P}_c$ with $g = 2, 3$. Simply we can get the minimum of $\breve{\xi}_2^U(1)$ and $\breve{\xi}_3^U(1)$ as an upper bound on $\breve{\xi}^U(1)$. Similarly, we can only get $\breve{\xi}_2^U(0) = \breve{\xi}_3^U(0) = 0$ as a lower bound on $\breve{\xi}^L(1)$. Now let us see if we can get some tighter bounds on $\breve{\xi}(1)$.

Let $\tau = t_0 t_1 t_2 ...$ denote the trace of the output stream at the fine granularity, where t_i denotes the production time of the i-th fine event (ε_i) for $i \geq 1$. We assume that the stream starts to be generated from $t_0 = 0$. Recall that $\breve{\xi}^L(k)$ and $\breve{\xi}^U(k)$ are defined to be lower and upper bounds on the length of the time interval during which any k consecutive events are output from the PE. For any $i \geq 0$, we have $\breve{\xi}^L(2) \leq t_{i+2} - t_i \leq \breve{\xi}^U(2)$ and $\breve{\xi}^L(3) \leq t_{i+3} - t_i \leq \breve{\xi}^U(3)$, from which we can derive,

$$\breve{\xi}^L(3) - \breve{\xi}^U(2) \leq t_{i+3} - t_{i+2} \leq \breve{\xi}^U(3) - \breve{\xi}^L(2) \tag{1}$$

We also have similar constraints of

$$\breve{\xi}^L(2) \leq t_3 - t_1 \leq \breve{\xi}^U(2),\ \breve{\xi}^L(3) \leq t_3 - t_0 \leq \breve{\xi}^U(3)$$

$$\breve{\xi}^L(2) \leq t_4 - t_2 \leq \breve{\xi}^U(2),\ \breve{\xi}^L(3) \leq t_4 - t_1 \leq \breve{\xi}^U(3)$$

from which we can derive,

$$\begin{array}{l} \breve{\xi}^L(3) - \breve{\xi}^U(2) \leq t_1 - t_0 \leq \breve{\xi}^U(3) - \breve{\xi}^L(2) \\ \breve{\xi}^L(3) - \breve{\xi}^U(2) \leq t_2 - t_1 \leq \breve{\xi}^U(3) - \breve{\xi}^L(2) \end{array} \tag{2}$$

Combining Ineqs. (1) and (2), we know that $[\breve{\xi}^U(3) - \breve{\xi}^L(2)]$ and $[\breve{\xi}^L(3) - \breve{\xi}^U(2)]$ respectively provide upper and lower bounds on the length of the time interval $[t_{i+1} - t_i]$ (for all $i \geq 0$) during which any single event is output. From the definition of arrival curve, we have $\breve{\xi}^U(3) - \breve{\xi}^L(2) \geq \breve{\xi}^U(1)$ and $\breve{\xi}^L(3) - \breve{\xi}^U(2) \leq \breve{\xi}^L(1)$. It follows from Lemma 1 that $\breve{\xi}^U(1)$ is upper bounded by $[\breve{\xi}_3^L(1) - \breve{\xi}_2^U(1)]$ and $\breve{\xi}^L(1)$ is lower bounded by $[\breve{\xi}_3^L(1) - \breve{\xi}_2^U(1)]$. Similarly, we can also obtain lower and upper bounds on $\breve{\xi}(1)$ from other pairs of $\breve{\xi}_3(k_1)$ and $\breve{\xi}_2(k_2)$, e.g. $\breve{\xi}_2(2)$ and $\breve{\xi}_3(1)$, $\breve{\xi}_3(3)$ and $\breve{\xi}_2(4)$, etc. Finally, $\tilde{\xi}^L(1)$ can be assigned with the maximum of all computed lower bounds on $\breve{\xi}^L(1)$ and $\tilde{\xi}^U(1)$ be with the minimum of all computed upper bounds on $\breve{\xi}^U(1)$.

Generally, our mathematical refinement algorithm works as follows. Given any k, we compute values of (ν_k^U, ν_k^L) for every pair of coarse granularities (g_i, g_j),

$$\nu_k^U = \min\{\breve{\xi}_{g_i}^U(a_1) - \breve{\xi}_{g_j}^L(b) | b \geq 0\}$$
$$\nu_k^L = \max\{\breve{\xi}_{g_i}^L(a_2) - \breve{\xi}_{g_j}^U(b) | b \geq 0\}$$

where $a_1 = \min\{k_1 | k_1 \times g_i - b \times g_j \geq k\}$ and $a_2 = \max\{k_1 | k_1 \times g_i - b \times g_j \leq k\}$. $\tilde{\xi}^U(k)$ and $\tilde{\xi}^L(k)$ can then be updated to $\min\{\nu_k^U, \tilde{\xi}^U(k)\}$ and $\max\{\nu_k^L, \tilde{\xi}^L(k)\}$ respectively. Clearly, if we analyze the component with more granularity levels, we may obtain tighter output curves $\tilde{\xi}$. Note that the refinement algorithm can be easily extended to the case that the bounds are provided on arbitrary $\breve{\xi}(k)$ where k may not be divisible by any g_i.

3 Models of PMC

3.1 Target Model

Given a TA model of the PE (processing element) in a PMC (power-managed component), we can cluster all its states into a finite set of operational modes $\mathcal{M}$. Each mode $M_i \in \mathcal{M}$ is characterized with distinct processing capability (and/or power consumption), which can be captured by a service curve $\psi_i(k) = (\psi_i^L(k), \psi_i^U(k))$ for $k \geq 0$. All the transitions whose source and destination states are clustered into the same mode are omitted. Our framework works on such models of a PE. We consider the following cases when a mode switch happens. We believe that our work can be easily extended to other cases of mode switch. A mode M_i may be specified with a time constraint in the form of a time interval $[L_i, U_i]$. It

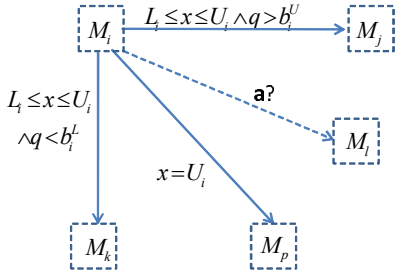

Figure 9: Descriptive model of a PE.

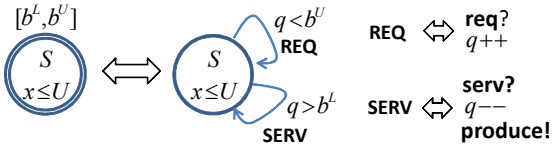

Figure 10: Simplified notation of a state of the PE.

means that the PE can only stay in M_i for at least L_i and at most U_i time units. The PE also switches its mode based on the buffer fill level (i.e. the number of events stored in the buffer), under the timing constraints of source mode. It is expected to run in a mode with higher processing capability when there are more buffered events. The power management policy may also force an immediate switch of the mode by emitting a signal, without caring the timing constraints.

Figure 9 shows a descriptive model of the PE considering the above possible cases. x denotes the time that the PE has stayed in mode M_i. q denotes the buffer fill level. Whenever the timing constraints $L_i \le x \le U_i$ is satisfied, if the buffer fill level q exceeds b_i^U, the PE switches to mode M_j; if q is less than b_i^L, it switches to mode M_k. It may also be forced to switch to model M_l whenever it receives a signal 'a?'. When it has stayed in M_i for up to U_i time units, it should transit to another mode M_p. In the following sections, we describe the details of fine TA models of a PMC (PE plus service model) based on this descriptive PE model, and then show how to abstract into coarse models.

3.2 Fine Models

When the PE stays in a mode, it receives signals 'req?' from the generator indicating an event has arrived and 'serv?' from the service model indicating an event has been processed. Firstly we introduce a simplified notation of a state, as shown in Figure 10. A state S specified with parameters $[b^L, b^U]$ can stay in this state only when the buffer fill level q is within this range. An invariant of $x \le U$ specifies an upper bound U on how long it can stay in this state, where x is a clock. In the state, it updates the buffer fill level whenever receiving 'req?' or 'serv?'. It also emits a 'produce!' to the observer at the same time as a 'serv?' is received.

Figure 11 shows the possible transitions for leaving a mode in PE. A mode M_i consists of two states S_i and S_{i1}. S_i is added to guarantee the lower timing constraint L_i for this mode. When the PE enters a mode M_i, it first stays in state S_i with the invariant of $x \le L_i$. When $x = L_i$, it is time to leave S_i. It can switch to a new mode if some threshold on q is already satisfied. Referring to the descriptive model shown in Figure 9, it can go to mode M_j if $q > b_i^U$ or to M_k if $q < b_i^L$, or else to state S_{i1}. It can stay in S_{i1} when q is within the range of $[b_i^L, b_i^U]$. Whenever q exceeds b_i^U or falls below b_i^L, it switches to new mode. When it has stayed up to U_i time units in mode M_i, it has to transit out to a new mode M_p. Whenever it receives a signal 'a?', it must immediately transit out, no matter whether it is staying in S_i or S_{i1} at that time.

As can be seen in Figure 11, whenever the PE transits from one mode M_i to a new one M_j, it sends a signal 'Syn$_{ij}$!' to synchronize with the service model.

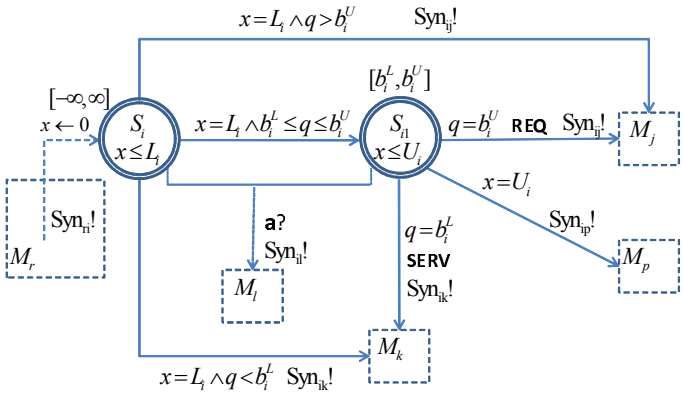

Figure 11: Fine TA model of the PE.

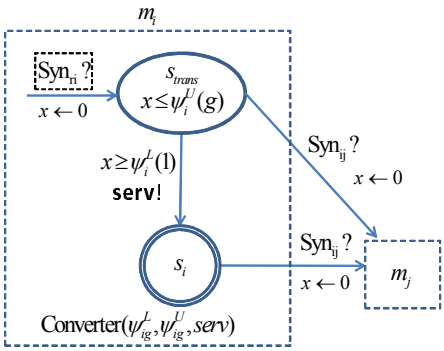

Figure 12: Simple coarse service model.

3.3 Simple Coarse Models

In the coarse models, the fine event stream is abstracted at some granularity of g for $g > 1$. The generator, defined by *Converter*($\hat{\xi}_g^L$, $\hat{\xi}_g^U$, *req*) shown in Figure 4, is now generating coarse events, where the fine arrival curve $\hat{\xi}$ is replaced with coarse one $\hat{\xi}_g$. The service model emits 'serv!' signal in a mode M_i, indicating a coarse event has been processed, based on the coarse service curve ψ_{ig}. The fine PE model is also abstracted to a coarse one, where the coarse buffer fill level is updated based on the number of coarse events received and processed. When the PE switches to a new mode, we do not know how many number of fine events have been processed since last 'serv?' in previous mode, which has to be within the range of $[0, g)$. In the new mode, it may need to process another $i \in (0, g]$ more fine events to receive the next 'serv?'. In the service model, as shown in Figure 12, we add a state S_{trans} to capture the service time for the rest of fine events until the next 'serv?' is emitted, which is lower and upper bounded by the fine service curves $[\psi_i^L(1), \psi_i^U(g)]$.

In the following, we show the rationale behind the simple coarse PE model. Suppose that N_r (N_s) denotes the total number of signals 'req?' ('serv?') received since the system starts, i.e. N_r (N_s) coarse events have arrived (been processed). n_r (or n_s) denotes the corresponding total number of fine events that have arrived (or been processed). Q (q) denotes the coarse (fine) buffer fill level. At any time, we have the following constraints,

$$\begin{array}{lll} gN_r & \le n_r < & g(N_r+1) \\ gN_s & \le n_s < & g(N_s+1) \end{array} \tag{3}$$

Since the fine and coarse buffer fill levels q and Q satisfy $q = n_r - n_s$ and $Q = N_r - N_s$ respectively, we have

$$g(Q-1) < q < g(Q+1) \tag{4}$$

from which we have $q/g - 1 < Q < q/g + 1$, which is equivalent to

$$\lfloor q/g \rfloor \leq Q \leq \lceil q/g \rceil \tag{5}$$

When q is replaced with $b_i^U + 1$ in Ineq. (5), we can obtain lower and upper bounds $[Y^L, Y^U]$ on the value of Q when it is possible to transit from M_i to M_j. Similarly, we can obtain bounds $[H^L, H^U]$ on the value of Q when it is possible to transit from M_i to M_k, by replacing q with $b_i^L - 1$. Y^L, Y^U, H^L, H^U is defined in Figure 13.

Figure 13 shows the simple coarse model of the PE. When it enters a mode M_i, it stays in S_i first until $x = L_i$. When it is time to leave S_i, it chooses where to go based on the coarse buffer fill level Q. If $Q > Y^U$ (or $Q < H^L$), it can transit to M_j (or M_k) directly; if Q is within $[Y^L, Y^U]$, it is possible that q is greater than b_i^U and hence moves to S_{inc}; in state S_{inc}, it can transit to M_j non-deterministically any time before Q reaches $Y^U + 1$; if Q is within $[H^L, H^U]$, it is possible that q is less than b_i^L and hence moves to S_{dec}. In state S_{dec}, it can transit to M_k non-deterministically any time before Q falls to $H^L - 1$; if $H^U < Q < Y^L$, q must be within $[b_i^L, b_i^U]$ and it moves to S_{i1} to wait. It can switch between S_{inc}, S_{i1}, and S_{dec} based on the update of Q. Figure 13 (b) shows that it should transit out immediately whenever a signal 'a?' is received or it has stayed in mode M_i for up to U_i time units.

Validation: Now we validate that the proposed simple coarse models provide correct abstraction for the fine models. In other words, the coarse models need to satisfy the proof obligation 1.
Proof: Let $\hat{\tau} = (\hat{t}_0)\hat{t}_1\hat{t}_2...$ be any fine input stream to the fine PMC $\mathcal{P}_0$, and $\breve{\tau} = (\breve{t}_0)\breve{t}_1...$ is the corresponding output stream. Firstly, we construct a coarse input stream $\hat{\top} = (\hat{T}_0)\hat{T}_1\hat{T}_2...$ with $\hat{T}_j = \hat{t}_{gj}$ for $j \geq 0$. It is clear that $\hat{T}$ can be refined to $\hat{\tau}$, while $\hat{T}$ can be an input to the coarse PMC $\mathcal{P}_c$ due to,

$$\hat{\xi}_g^L(j) = \hat{\xi}^L(gj) \leq \hat{t}_{gj} - \hat{t}_0 \leq \hat{\xi}^U(gj) = \hat{\xi}_g^U(j), \quad \forall j \geq 0 \tag{6}$$

Note that we can always get this coarse input stream $\hat{\top}$ independent of the abstracted PMC model $\mathcal{P}_c$, since the generator is same for any abstracted $\mathcal{P}_c$.

Then we show how to construct a coarse output stream $\breve{T}$ from $\mathcal{P}_c$ corresponding to the input stream $\hat{T}$ by comparing the behaviors of $\mathcal{P}_0$ and $\mathcal{P}_c$, where $\breve{T}$ is an abstract of $\breve{\tau}$. We assume the following lemma is true, which will be proven by induction.

Lemma 2 *Given the behavior of $\breve{\tau}$ for the fine PE, we can build a behavior of the coarse PE where it switches mode at the same time as the fine PE. Also, the coarse output stream $\breve{T}$ is an abstract of the fine output $\breve{\tau}$.*

Firstly, when the system starts, both the PEs can enter the starting mode at the same time.

Suppose that the coarse and fine PE transits to M_i from M_r at the same time. Assume that the coarse output stream $\breve{T}$ has been constructed to be $\breve{T} = (\breve{T}_0)\breve{T}_1...\breve{T}_{A-1}$ with $\breve{T}_j = \breve{t}_{gj}$ for some $A \geq 1$ and all $0 \leq j \leq A - 1$. Suppose that the fine PE leaves M_i when the $(gB + n)$-th fine event is processed with $n < g$. It can be classified into two cases.
Case 1: $B = A - 1$. From the service model shown in Figure 12, the time to generate next 'serv?' for the coarse PE is lower and upper bounded by $\psi_i^L(1)$ and $\psi_i^U(g)$ respectively. It covers all the possibility of what can happen in the fine service model. Hence, it is possible that the coarse PE leaves M_i at the same time as the fine PE.
Case 2: $B > A - 1$. In mode M_i, we continue to construct $\breve{T}$ to be $...\breve{T}_{A-1}\breve{T}_A...\breve{T}_B$, with $\breve{T}_j = \breve{t}_{gj}$ for $A \leq j \leq B$. Similar to the explanation in *Case 1* above, we can show that $\breve{T}_A$ can be an output from $\mathcal{P}_c$. We can also show that $\breve{T}_{A+1}...\breve{T}_B$ can be an output from $\mathcal{P}_c$, due to

$$\psi_i^L(g(j - A)) \leq \breve{t}_{gj} - \breve{t}_{gA} \leq \psi_i^U(g(j - A))$$

and

$$\psi_{gi}^L(j - A) = \psi_i^L(g(j - A)), \psi_i^U(g(j - A)) = \psi_{gi}^U(j - A)$$

We can also show that the coarse PE may leave M_i at the same time as the fine PE in *Case 2*. As shown in Figure 13, $\mathcal{P}_c$ may transit out from S_{inc} (or S_{dec}) anytime before Q increases to $Y^U + 1$ (or falls below

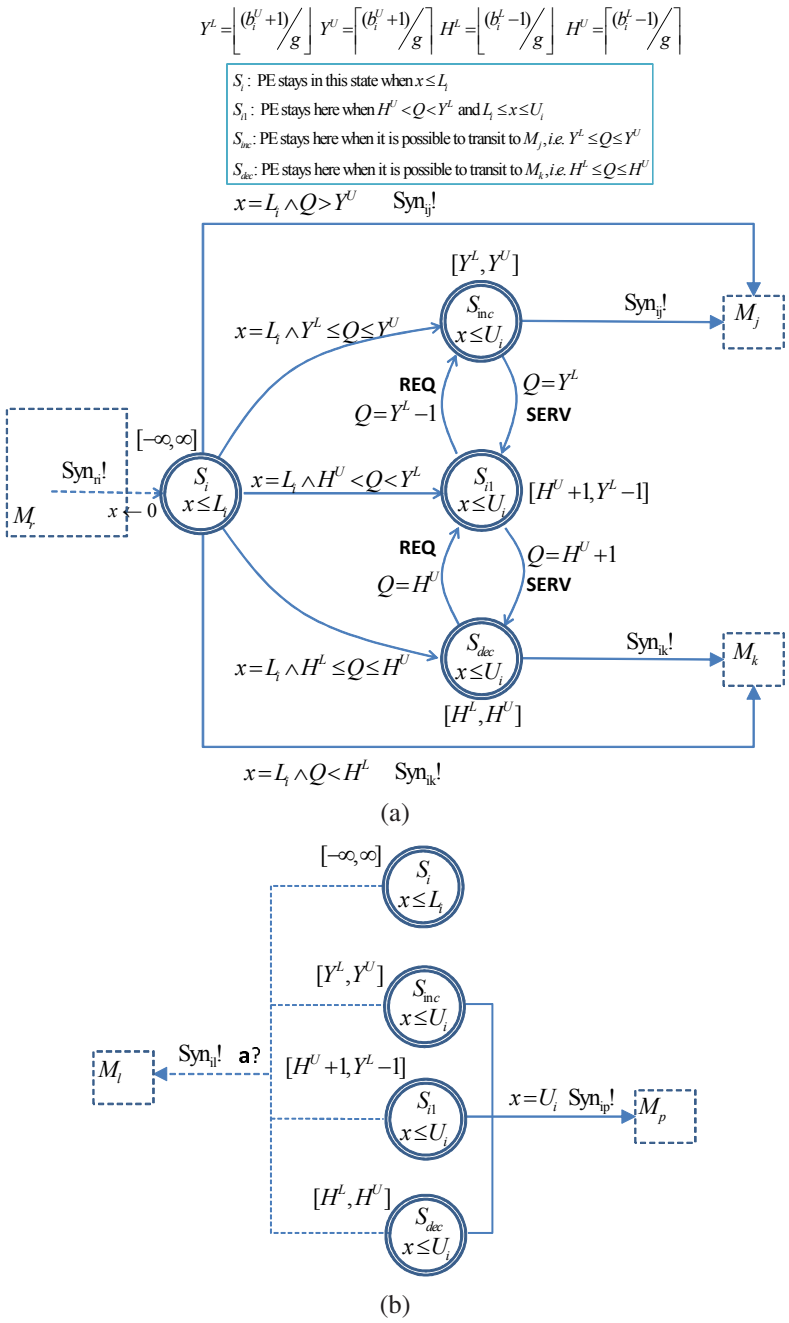

Figure 13: Decomposed simple coarse models of the PE [in (a) and (b), two states with same notations just refer to the same state of the full PE model].

$H^L - 1$). Hence, it is possible that $\mathcal{P}_c$ transits out at the same time as $\mathcal{P}_0$ in this case. In other cases of transiting out, the time to transit out is same for both $\mathcal{P}_c$ and $\mathcal{P}_0$, which is equal to L_i, U_i or the time receiving synchronization signal 'a?'.

It is also clear that the constructed coarse output stream $\breve{T}$ is an abstract of the fine one $\breve{\tau}$. Therefore, the proof obligation 1 is validated. □

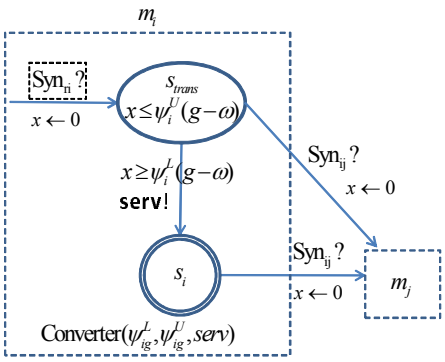

Figure 14: Coarse service model with parameter ω.

3.4 Coarse Models with ω and Its Optimization

When the PE switches from a mode M_r to M_i, we use a variable ω to denote the number of fine events that have been processed since last 'serv?', satisfying $0 \le \omega < g$. In the new mode, $(g-\omega)$ more fine events need to be processed before the next 'serv!' is generated. We modify the coarse service model to take into account the variable ω. As shown in Figure 14, since entering the new mode, the time to generate the next 'serv!' is lower and upper bounded by $\psi_i^L(g-\omega)$ and $\psi_i^U(g-\omega)$ respectively. Whenever the PE receives a signal 'serv?', ω is reset to zero.

In Ineq. (3), the total number of fine events n_s that has been served at anytime can now be expressed as $gN_s + \omega \le n_s < g(N_s+1)$. The relationship between the coarse and fine buffer fill levels Q and q, as shown in Ineq. (5), is now updated to

$$\lfloor (q+w)/g \rfloor \le Q \le \lceil q/g \rceil \tag{7}$$

If the mode M_i is a sleep mode, i.e. no service is provided, we have $n_s = gN_s + \omega$. The bounds on Q can be improved with

$$\lfloor (q+w)/g \rfloor \le Q \le \lfloor (q+w)/g \rfloor \tag{8}$$

The values of $Y^{L/U}$ and $H^{L/U}$ are updated following Figure 15.

Since the arrival and service patterns for the corresponding fine stream should also follow the specification of the fine curves $\hat{\xi}$ and ψ_i, it is possible to optimize the coarse models of the PE. We can estimate a tighter bound on ω when it switches the modes and then can compute tighter bounds on the time to transit from S_{inc} to M_j or from S_{dec} to M_k, for the decomposed model shown in Figure 13 (a).

Assume that when entering S_{inc} or S_{dec}, the coarse PE has received N_r 'req?' and N_s 'serv?' signals. We define a clock t_a which is reset whenever a 'req?' signal is received, and a clock t_s which is reset whenever a 'serv?' is received or when it enters a new mode. If no subsequent events arrives since 'req?' and the service is stopped since when t_s is reset, the fine buffer fill level q is just equal to $(gQ-\omega)$. Before transiting to M_j from S_{inc}, q should increase by $d_1 = b_i^U + 1 - (gQ-\omega)$. Suppose that q reaches b_i^U+1 before the next 'serv?' is received and ω' fine events have been processed since when t_s is reset. There must be $(d_1+\omega')$ more fine events having arrived since last 'req?'. Considering both the arrival and service patterns of the fine events, we can estimate a lower and upper bound on ω'.

If $\hat{\xi}^L(d_1+j) > \psi_i^U(j) + t_a - t_s$ is true, it implies that the next (d_1+j) fine events cannot arrive before the next j events are processed in any case, hence it is impossible for q to reach (b_i^U+1) when j more fine events are processed. Similarly, if $\hat{\xi}^U(d_1+j) < \psi_i^L(j) + t_a - t_s$ is true, it implies that the next (d_1+j) fine events always arrive before j more fine events are processed, i.e. q must have reached b_i^U+1 when (d_1+j) more events arrive. We can then compute (denoted by $\underline{\omega'_{inc}}$)

$$\begin{array}{rcl} i_s & = & \min_{0 \le j \le g}\{j | \hat{\xi}^L(d_1+j) \le \psi_i^U(j) + t_a - t_s\} \\ i_e & = & \min_{i_s \le j \le g}\{j | \hat{\xi}^U(d_1+j) \le \psi_i^L(j) + t_a - t_s\} \end{array} \tag{9}$$

where $d_1 = b_i^U + 1 - (gQ-\omega)$. ω' is lower and upper bounded by i_s and i_e respectively.

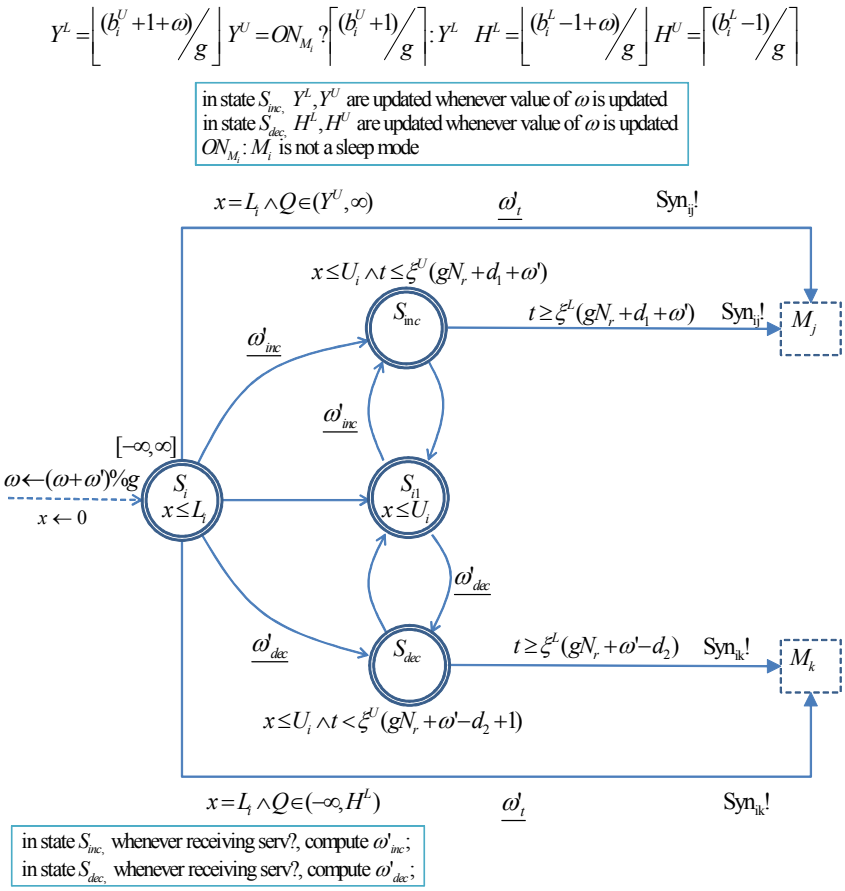

Figure 15: Optimization of the decomposed simple coarse PE model shown in Figure 13 (a).

Based on a similar rationale, we can compute i_s and i_e when the PE enters S_{dec} as follows (denoted by $\underline{\omega'_{dec}}$),

$$\begin{array}{rcl} i_s & = & \min_{0 \le d_2 + j \le g}\{d_2 + j | \hat{\xi}^U(j) \ge \psi_i^L(d_2 + j) + t_a - t_s\} \\ i_e & = & \min_{i_s \le d_2 + j \le g}\{d_2 + j | \hat{\xi}^L(j) \ge \psi_i^U(d_2 + j) + t_a - t_s\} \end{array} \quad (10)$$

where $d_2 = gQ - w - (b_i^L - 1)$.

As shown in Figure 15, the invariant of S_{inc} becomes a conjunction of $x \le U_i$ and $t \le \hat{\xi}(gN_r + d + \omega')$. A guard of $t \ge \hat{\xi}^L(gN_r + d_1 + \omega')$ is imposed on the previously un-guarded transition from S_{inc} to M_j. Similarly, an item of $t < \hat{\xi}_i^U(gN_r + \omega' - d_2 + 1)$ is added to the invariant of S_{dec} and a guard of $t \ge \hat{\xi}_i^L(gN_r + \omega' - d_2)$ is put on the previously un-guarded transition from S_{dec} to M_k.

For the decomposed model in Figure 13 (b), where the time to transit out from M_i is determined (i.e. when $x = L_i$, $x = U_i$ or receiving a signal 'a?'), we can compute $[i_s, i_e]$ based on the time t_s as follows (denoted by $\underline{\omega'_t}$):

$$\begin{array}{rcl} i_s & = & \min_{0 \le j \le g}\{j | \psi_i^U(j) \le t_s\} \\ i_e & = & \max_{i_s \le j \le g}\{j | \psi_i^L(j) \le t_s\} \end{array} \quad (11)$$

The value of ω is then updated to be $(\omega + \omega')\%g$ when it is leaving to a new mode. The optimized coarse models can also be validated, following a similar procedure of validating the simple coarse models as shown in section 3.3.

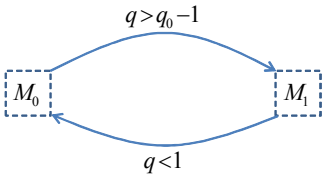

Figure 16: Despcriptive model of the example PE.

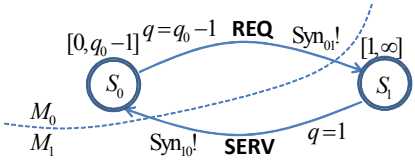

Figure 17: Fine model of the example PE.

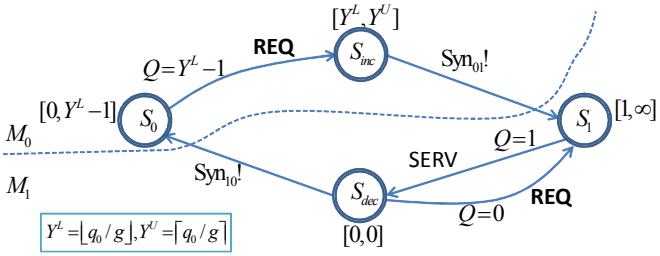

Figure 18: Simple coarse model of the example PE.

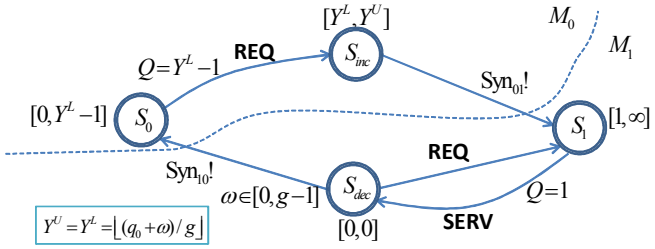

Figure 19: Coarse model with ω of the example PE.

4 Illustrative Example

In this section, we illustrate with the fine and coarse models of a simple but common PMC. The PE runs at two modes: sleep and run. It only processes events and consumes power at run mode. Whenever there are not enough number of events waiting for processing in the buffer, it tries to stay in sleep mode to save energy. Initially the input buffer is empty, the PE switches to run mode when the buffer fill level q (i.e. the number of fine events available in the buffer) reaches a predefined threshold q_0. The PE switches to sleep mode when the buffer becomes empty again. Figure 16 shows the descriptive model of this example PE.

Since the mode switch is only dependent on the buffer fill level, we can simplify the models of the PE. In all the fine and coarse models, the states S_i and S_{i1} are integrated into a single state S_i. Some transitions can be removed then. Figure 17 shows the fine model of the example PE. The sleep mode M_0 consists of only one state S_0 and the run mode M_1 consists of only S_1. The corresponding simple coarse model, coarse model with ω and optimized coarse model of the PE are shown in Figures 18, 19 and 20 respectively.

In Figure 21, we take an example input stream and show its execution in the fine and coarse PE models.

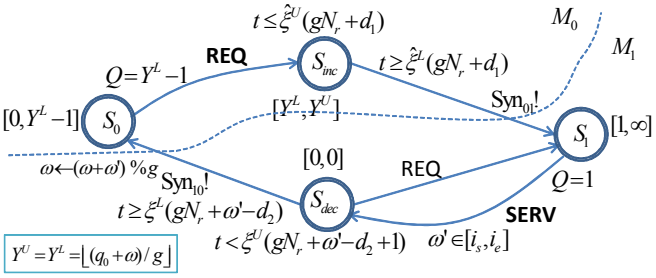

Figure 20: Optimized coarse model of the example PE.

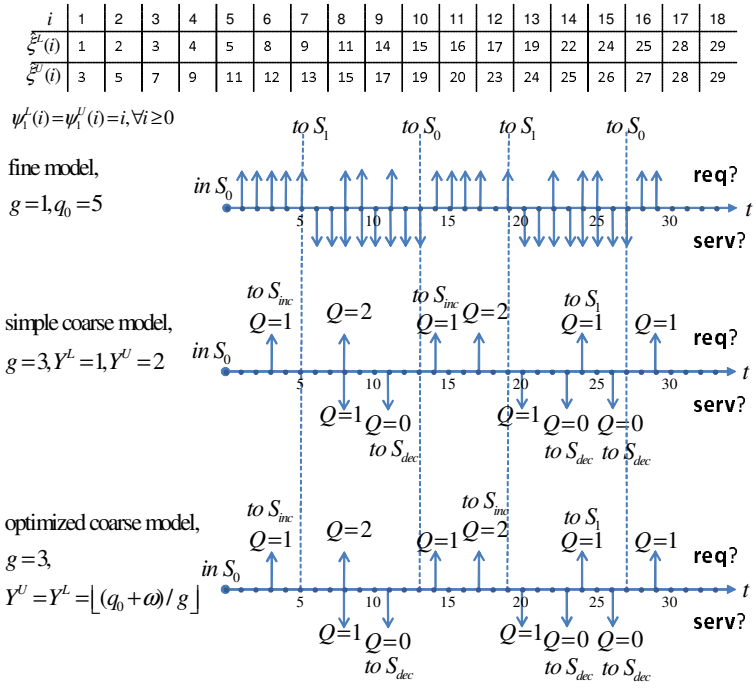

Figure 21: An illustrated flow of the example PE.

t	ω	d_1/d_2	$t_a - t_s$	(i_s, i_e)	ω'	N_r	bounds on mode switch
3	0	$d_1 = 2$	/	(0,0)	0	1	$[5, 11]$
11	0	$d_2 = 0$	3	(2,3)	2	2	$[11, 17)$
17	2	$d_1 = 1$	/	(0,0)	0	4	$[19, 24]$
23	0	$d_2 = 0$	6	(3,3)	3	4	$[15, 27)$
26	0	$d_2 = 0$	2	(1,3)	1	5	$[25, 28)$

Table 1: The computed values of variables for the illustrated flow.

The threshold q_0 is chosen to be 5, and the granularity is 3. It can be observed that the coarse PE always switches from M_0 to M_1 when it stays in S_{inc} and switches from M_1 to M_0 when it stays in S_{dec}. It then illustrates that the non-determinism is introduced in the coarse models. The update of variables for optimized coarse PE model are also shown in Table 1.

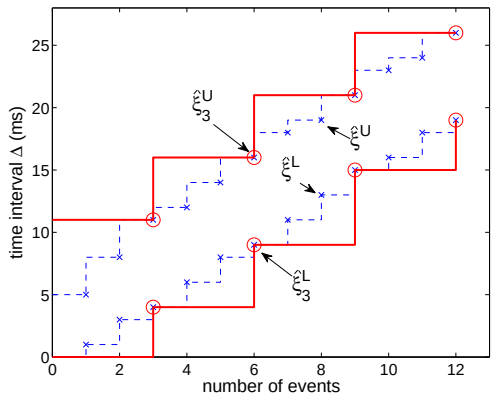

Figure 22: Example of fine and coarse input arrival curves $\hat{\xi}^{L/U}$ and $\hat{\xi}^{L/U}_3$.

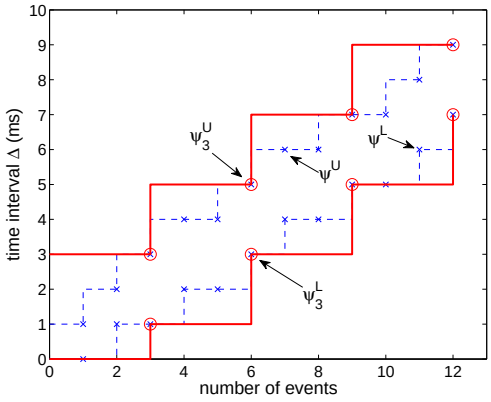

Figure 23: Example of fine and coarse service curves $\psi^{L/U}$ and $\psi^{L/U}_3$.

5 Experimental Validation

We have conducted experiments to validate our proposed framework, using the example of a PMC presented in section 4. We applied the TA modeling and verification tool UPPAAL CORA [16, 17], to model the generator, observer, PMC and to analyze the output arrival curves by model checking. We implemented the mathematical refinement algorithm in Matlab, which can be simply integrated with the RTC toolbox [18]. We used Perl scripts to describe the interface between the specification of input arrival curves and generator, to manage the multiple runs of analyzing the TA models of a PMC at different granularities, and to pass the analyzed coarse output arrival curves to the mathematical refinement algorithm.

First we show the results for an example event stream, to validate that the abstracted coarse model $\mathcal{P}_c$ of the PMC provides accurate abstraction of its fine one $\mathcal{P}_0$. The arrival patterns of the fine stream are lower and upper bounded by input arrival curves $\hat{\xi}^L$ and $\hat{\xi}^U$ respectively, as shown in Figure 22. The coarse arrival curves $\hat{\xi}^L_g$ and $\hat{\xi}^U_g$ at granularity of g (e.g. $g = 3$) were obtained by sampling the fine curves, i.e. $\hat{\xi}^L_g(i) = \hat{\xi}^L(gi)$ and $\hat{\xi}^U_g(i) = \hat{\xi}^U(gi)$ for $i \geq 0$. Similarly, the execution time patterns of the fine event stream at the run mode are bounded by service curves $\psi^{L/U}$, which can also be sampled to obtain coarse service curves $\psi^{L/U}_g$, as shown in Figure 23.

Using the UPPAAL CORA tool, we can analyze the minimum cost to reach the "Stop" state of the observer model, which can be assigned to the lower output curve $\breve{\xi}^L_g(k)$ (if $g = 1$, $\breve{\xi}_1$ just denotes $\breve{\xi}$ computed from the fine model $\mathcal{P}_0$). Since it is still not implemented in the tool to analyze the maximum cost, we used a varied model of the *observer* as shown in Figure 6 (b), in order to compute the upper output curve. Similar to the analysis of $\breve{\xi}^L_g$, we can compute $\breve{\alpha}^L_g(\Delta)$ for $\Delta \geq 0$, which provides the lower bound on the number of coarse events at granularity of g that can be produced within any time interval of length Δ. We can then compute a pseudo-inverse curve of $\breve{\alpha}^L_g(\Delta)$ to obtain $\breve{\xi'}^U_g(k)$, which actually provides an upper bound on $\breve{\xi}^U(gk-1)$.

By setting the threshold q_0 to be 5 (i.e. the PE starts to run when the number of fine events in the buffer reaches 5), we computed output curves $\breve{\xi}^L_4$-s/$\breve{\xi'}^U_4$-s, $\breve{\xi}^L_4$-ω/$\breve{\xi'}^U_4$-ω, $\breve{\xi}^L_4$-opt/$\breve{\xi'}^U_4$-opt using the simple coarse PMC model $\mathcal{P}_c$-s, coarse PMC model with ω $\mathcal{P}_c$-ω and optimized coarse PMC model $\mathcal{P}_c$-opt respectively, as shown in Figure 24. It can be observed that $\mathcal{P}_c$-ω helps to obtain tighter coarse output curves than $\mathcal{P}_c$-s, which are further improved by $\mathcal{P}_c$-opt.

We analyzed the coarse output curves $\breve{\xi}^L_g$/$\breve{\xi'}^U_g$ at granularity of $g = 2, 3, 4$ from the optimized coarse PMC $\mathcal{P}_c$-opt and compared with the fine output curves $\breve{\xi}^L$/$\breve{\xi'}^U$ analyzed from the fine PMC $\mathcal{P}_0$, as shown in Figure 25. It can be observed that $\breve{\xi}^L_g(k)$ provides a lower bound on $\breve{\xi}^L(gk)$ and $\breve{\xi'}^U_g(k)$ provides an upper bound on $\breve{\xi'}^U(gk)$. It then validates lemma 1.

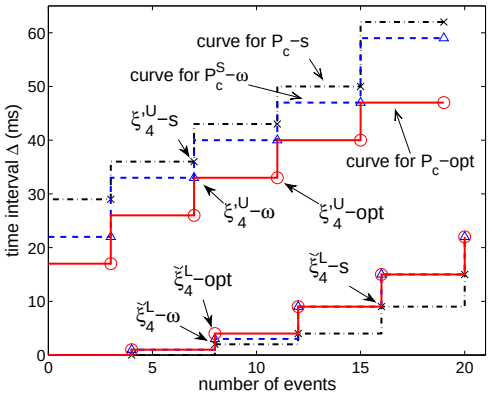

Figure 24: Comparison of output arrival curves at $g = 4$ using simple coarse PMC $\mathcal{P}_c$-s, coarse PMC with ω $\mathcal{P}_c$-ω and optimized PMC $\mathcal{P}_c$-opt.

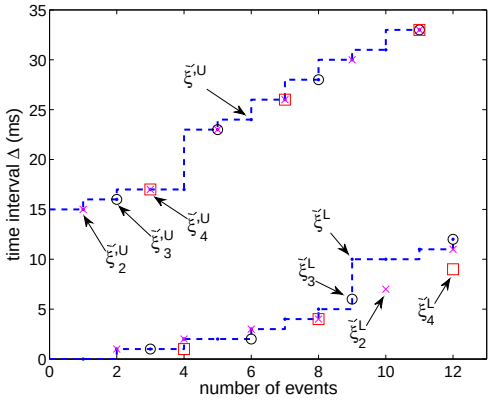

Figure 25: Comparison between fine and coarse output arrival curves computed from fine and optimized coarse PMC models respectively.

output arrival curves	analysis time at granularity of g [sec]									
	$g=1$	$g=2$			$g=3$			$g=4$		
	$\mathcal{P}_0$	$\mathcal{P}_c$-s	$\mathcal{P}_c$-ω	$\mathcal{P}_c$-opt	$\mathcal{P}_c$-s	$\mathcal{P}_c$-ω	$\mathcal{P}_c$-opt	$\mathcal{P}_c$-s	$\mathcal{P}_c$-ω	$\mathcal{P}_c$-opt
$\breve{\xi}^L_g(k)$	23674.4	56.1	62.9	223.6	7.90	12.9	189.7	0.76	1.21	28.6
$\breve{\xi'}^U_g(k)$	411515.3	746.2	583.2	4217.7	68.6	120.9	1465.4	7.43	12.5	269.1
distance	/	4.17	2.46	0.63	5.50	2.88	0.63	9.08	5.08	1.25

Table 2: Analysis time to compute fine and coarse output arrival curves $\breve{\xi}^L_g(k)$ and $\breve{\xi'}^U_g(k)$, plus the distance between coarse and fine curves.

Table 2 summarizes the total time to compute the fine and coarse output curves $\breve{\xi}^L_g(k)/\breve{\xi'}^U_g(k)$ for $g = 1, 2, 3, 4$ and $k \leq \llcorner 24/g \lrcorner$. It also shows the distance measured between coarse and fine curves, which is computed by

$$mean(mean(\breve{\xi}^L(gk) - \breve{\xi}^L_g(k)), mean(\breve{\xi}^U_g(k) - \breve{\xi}^U(gk)))$$

where $mean$ is to compute the average of all the elements for $1 \leq k \leq \llcorner 24/g \lrcorner$. It can be generally observed that for a given granularity, simple model $\mathcal{P}_c$-s has the shortest analysis time and lowest precision and optimized $\mathcal{P}_c$-opt has the longest analysis time and highest precision. As the granularity is coarser, the analysis time decreases for a given abstraction model, with less precision too. For different selected coarse granularities, even the optimized model reduces the analysis time by at least 99% than that for a fine curve.

To demonstrate the advantage of our mathematical refinement algorithm, we show the results for another example of event stream with $q_0 = 21$. We analyzed the optimized coarse PMC $\mathcal{P}_c - opt$ at two granularities of $g = 9, 10$. We can obtain a fine output arrival curve $\tilde{\xi}$-granu, simply from the bounds given by $\breve{\xi}^L_g/\breve{\xi'}^U_g$. Using the mathematical refinement algorithm, we can improve on $\tilde{\xi}$-granu and obtain a tighter curve $\tilde{\xi}$-ref. As shown in Figure 26, we know that the value of $\breve{\xi}^U(10)$ (from fine PMC $\mathcal{P}_0$) should be at least $\breve{\xi'}^U_{10}(1) = 91$ and at most $\breve{\xi'}^U_9(2) = 108$. $\tilde{\xi}^U$-granu provides an upper bound of 108 on $\breve{\xi}^U(10)$, while $\tilde{\xi}^U$-ref provides 102, which improves by 35.3% relative to the lower bound of 91 on $\breve{\xi}^U(10)$. It is expected to get tighter $\tilde{\xi}$-ref if we analyze with more different granularities.

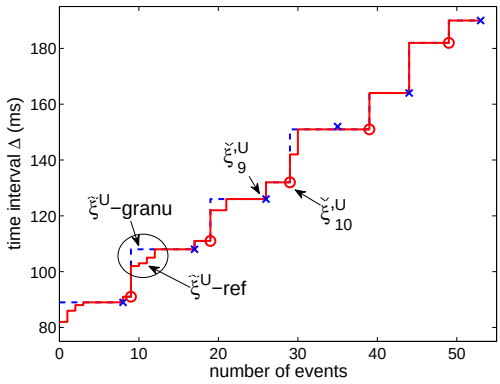

Figure 26: Computed fine upper output arrival curves $\tilde{\xi}^U$-granu (with simple scheme) and $\tilde{\xi}^U$-ref (with mathematical refinement algorithm).

5.1 Summary of the Abstraction

When the fine stream is abstracted into a coarse level, we do not know the exact index of the fine event when the mode switch happens. With the information of coarse buffer fill levels, we add non-determinism into the coarse PMC models to over-approximate the behavior of the fine PMC. On one hand, the time (i.e. the number of fine events having arrived or been processed, since the arrival or process of last coarse event) to leave a mode is approximated. On the other hand, after switching into the destination mode, the time to complete processing next coarse event cannot be captured by the converter of state s_i, as shown in Figure 12 and 14. We have to model the service time for the fine events before next coarse event and the subsequent coarse events separately, which is over-approximate compared to the corresponding fine models. Our framework provides various trade-offs based on different granularities. A qualitative result can be obtained for the precision and analysis time of computing the output curves from coarse models. However, it is very challenging to quantify them.

6 Conclusion

In this paper, we have proposed a novel framework of granularity-based interfacing between RTC and TA performance models, which complements the existing work and reduces the complexity of analyzing a state-based component modeled by TA. We have illustrated with an example to show how the model of a component is abstracted to work with an event stream at coarse granularity and how the abstraction is validated. Experimental results show that the time to analyze the coarse models reduces at least 99% of that for the fine models. It is also demonstrated that our proposed mathematical refinement algorithm improves on the bounds on the arrival patterns of the fine output stream, by using the results from multiple runs of analyzing the coarse models at different granularities.

In the future, we will continue to study the problem of speeding up the analysis of a state-based component by abstraction. It is interesting to explore the possibility of adopting RTC theory in the state-space exploration of the TA modeled component. Along this direction, it may refer to the work of multi-mode RTC [10]. On the other hand, it is interesting, but more challenging, to work on abstraction techniques for analyzing the energy consumption of a component.

References

[1] L. Benini, A. Bogliolo, and G. D. Micheli, "A survey of design techniques for system-level dynamic power management," *IEEE Transactions on VLSI Systems*, vol. 8, no. 3, pp. 299–316, 2000.

[2] A. Bogliolo, L. Benini, E. Lattanzi, and G. D. Micheli, "Specification and analysis of power-managed systems," *Proceedings of the IEEE*, vol. 92, no. 8, pp. 1308–1346, 2004.

[3] 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.

[4] S. Chakraborty, S. Künzli, and L. Thiele, "A general framework for analysing system properties in platform-based embedded system designs," in *DATE*, 2003.

[5] S. Perathoner, E. Wandeler, and L. T. etc., "Influence of different system abstractions on the performance analysis of distributed real-time systems," in *EMSOFT*, 2007.

[6] R. Alur and D. L. Dill, "A theory of timed automata," *Theoretical Computer Science*, vol. 126, pp. 183–235, 1994.

[7] L. D. Alfaro and T. A. Henzinger, "Interface theories for component-based design," in *EMSOFT*, 2001.

[8] S. Künzli, A. Hamann, R. Ernst, and L. Thiele, "Combined approach to system level performance analysis of embedded systems," in *CODES/ISSS*, 2007.

[9] S. Schliecker, S. Stein, and R. Ernst, "Performance analysis of complex systems by integration of dataflow graphs and compositional performance analysis," in *DATE*, 2007.

[10] L. T. Phan, S. Chakraborty, and P. S. Thiagarajan, "A multi-mode real-time calculus," in *RTSS*, 2008.

[11] L. T. Phan, S. Chakraborty, P. S. Thiagarajan, and L. Thiele, "Composing functional and state-based performance models for analyzing heterogeneous real-time systems," in *RTSS*, 2007.

[12] S. Chakraborty, T. X. L. Phan, and P. Thiagarajan, "Event count automata: A state-based model for stream processing systems," in *RTSS*, 2005.

[13] "CATS tool," `www.timestool.com/cats/`.

[14] S. M. Yardi, K. Channakeshava, M. S. Hsiao, T. L. Martin, and D. S. Ha, "A formal framework for modeling and analysis of system-level dynamic power management," in *International Conference on Computer Design*, 2005.

[15] W. Haid and L. Thiele, "Complex task activation schemes in system level performance analysis," in *CODES/ISSS*, 2007.

[16] "UPPAAL CORA," `www.cs.aau.dk/ behrmann/cora/`.

[17] K. Larsen, G. Behrmann, E. Brinksma, A. Fehnker, T. S. Hune, P. Pettersson, and J. Romijn, "As cheap as possible: Efficient cost-optimal reachability for priced timed automata," in *Computer-Aided Verification (CAV)*, 2001.

[18] E. Wandeler and L. Thiele, "Real-Time Calculus (RTC) Toolbox," `www.mpa.ethz.ch/Rtctoolbox`.