TRAFFIC SCHEDULING ANOMALIES IN TEMPORAL PARTITIONS

Luís Almeida, Paulo Pedreiras, Ricardo Marau
LSE – IEETA / DET, Universidade de Aveiro, 3810-193 Aveiro, Portugal

Abstract: Many network protocols rely on temporal partitions to provide isolation between different nodes (TDMA slots) or different traffic classes (multi-phase cyclic frameworks). Typically, the duration of the slots or phases is not correlated with the duration of packet transmissions, which is variable and non-preemptive. Thus, it is possible that the limit of the slot or phase be overrun by an on-going packet transmission or, if this cannot be tolerated, idle-time must be inserted at the end of the slot or phase whenever a packet does not fit in. Nevertheless, both situations lead to scheduling anomalies in which the worst-case network delay does not occur necessarily with the synchronous release of all other packets, or just the higher priority ones. This paper highlights two such anomalies showing their origin and indicating that, in such circumstances, it is not possible to determine the worst-case network delay with exactitude in the general case. However, it is still possible to upper bound the network delay and the paper shows non-optimal solutions for those cases.

Keywords: real-time systems, real-time scheduling, real-time communication, distributed real-time systems

1. INTRODUCTION

The expression *scheduling anomaly* in real-time processor scheduling is typically used to refer to a counter intuitive phenomenon in which increasing the systems resources or relaxing the application constraints can lead to increased schedule length. In a real-time scope, this means that previously schedulable sets may then become non-schedulable. Probably the first such anomalies being identified were the multiprocessor scheduling anomalies described in (Graham, 1976) and known as Richard's anomalies (Stankovic

etal., 1995). The cause for such anomalies was related to the use of mutually exclusive shared resources. A simple illustrative example is shown in (Stankovic *etal*., 1995) in which the reduction of the execution time of one task inverts the order by which two subsequent competing tasks, allocated to different processors, access a shared resource. Such inversion may cause an extra delay in the global schedule, possibly violating deadlines.

In off-line scheduled applications, the anomalies were often hidden by optimization algorithms that would implicitly avoid them by acting upon the task offsets and inserting idle-time where needed to delay the execution of certain tasks. However such kind of optimization algorithms are too complex to be used online and thus, under dynamic conditions, e.g. sporadic arrivals, anomalies can indeed occur and generate worst-case response times that are worse than expected. Buttazzo (2002) has shown that, when using variable speed processors to reduce energy consumption, the anomalies can occur even in uniprocessor systems whenever tasks synchronize in the access to shared resources. To prevent such situation, he proposes using non-synchronized task interactions through asynchronous buffers. He also refers to a previous work by Mok (2000) that shows that anomalies can also occur in variable speed uniprocessor systems with non-preemptive tasks. Recently, Chen *etal* (2005) showed more anomalies in uniprocessor systems arising from hardware configuration changes, such as processor upgrading, with mutually exclusive shared resources and active I/O devices. In their work, they propose three rules to prevent scheduling anomalies, namely inserting idle-time, enforcing the same order when accessing resources and dealing separately with passive and active resources.

Another form of anomaly can occur when using temporal partitions in hierarchical scheduling frameworks. These have been recently studied by Shin and Lee (2003), Bini and Lipari (2004), Almeida and Pedreiras (2004), Davis and Burns (2005) and others. In these studies, it has become clear that the response time of a task within a partition depends not only on the set of interfering tasks that run within the same partition but also on the sequence of availability periods of the respective partition. Particularly, Davis and Burns (2005) show that the partition utilization as a function of the partition period, subject to assuring the schedulability of a given task set, presents local minima that can be very sharp in special cases. These cases correspond to the situations in which the tasks periods are integer multiples of the partition period and the tasks are released synchronously with the partition instances. This leads to a good packing of the tasks within the partition and any slight variation, either reduction or increase, in the partition period will require a significant increase in the partition size to fulfill the same task requirements. This is also an anomaly since reducing the partition period, i.e. increasing computing resources, may turn schedulable tasks unschedulable.

In this paper, we will address a similar case to the previous one, but considering non-preemptive execution. This model is typical in real-time networks used for control applications, in which communication is achieved through single packet transfers over shared broadcast buses. In this scope, temporal partitions have long been used either to control the access to the shared medium, e.g. TDMA, or to separate different classes of traffic, e.g. the multi-phase cyclic framework.

We will start by presenting our reference model and then highlighting two anomalies that may occur in these systems. We will terminate by presenting one technique for each case that allows bounding the anomaly duration and consequently its worst-case impact on network induced delay.

2. SCHEDULING MODEL

We consider a communication system based on a shared broadcast bus. The bus conveys messages which, for the purpose of this work, are considered as single packet and thus are transmitted non-preemptively. This is common in control networks or networks of sensors and actuators. Moreover, we consider the bus to be time-partitioned, meaning that specific types of communication can only occur within pre-defined intervals of time, being suspended outside those intervals. Examples of time-partitions are TDMA slots as in TT-CAN and TTP/C, and periodic/aperiodic phases, sometimes referred to as synchronous/asynchronous, as in WorldFIP, FlexRay, Ethernet POWERLINK, or the FTT protocols. Token passing protocols, e.g. PROFIBUS, also constrain the intervals during which nodes may access the bus, i.e., the token holding intervals. When the token holding time expires the nodes do not issue further transmissions no matter the number or priority of their ready messages. In this sense, the token holding intervals of a given node also form a temporal partition.

Therefore, we consider a time-partitioned system characterized by a set M of messages that will be scheduled within a partition Π. The message set M is defined as in (1), containing N periodic messages. Message m_i has maximum transmission time C_i, period T_i, initial offset Φ_i and priority Pr_i either fixed or variable depending on the particular scheduling policy.

$$M \equiv \{m_i (C_i, T_i, \Phi_i, Pr_i),\ i=1..N\} \tag{1}$$

The partition Π is formed by an infinite periodic sequence of time intervals or windows, during which the bus is available for transmitting the messages of set M. The period of the windows, also called the partition period, is π, and the windows duration, also called partition capacity, is

called w (2). In TDMA protocols, π is the TDMA round and w is the slot duration. In multi-phase cyclic frameworks, π is the micro-cycle and w is the phase duration. Moreover, as expressed in (2), we consider two types of partitions depending on whether the partition capacity is constant (type 1) or variable (type 2). TDMA protocols use partitions of type 1 only, while multi-phase cyclic frameworks normally use partitions of both types, the periodic phase being typically of type 1 and the aperiodic phase of type 2 since it reclaims the capacity not used by the periodic phase (Fig. 1).

$$\Pi^1 = \Pi^1(\pi, w=\text{cons}^{te}) \; ; \quad \Pi^2 = \Pi^2(\pi, w_{min}<w<w_{max}) \tag{2}$$

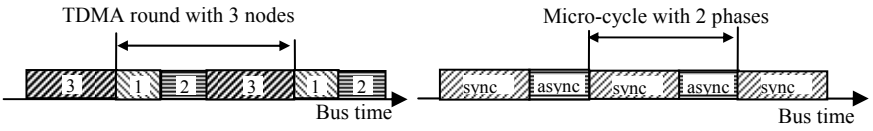

Figure 1. Examples of time-partitioned buses.

Moreover, we consider the following constraints on the message set, which are typically found in the communication systems referred before. Firstly, the message transmission time is always substantially shorter than the partition capacity ($\forall m_i,\ C_i<<w$). Secondly, the message periods and initial offsets are integer multiples of the partition period ($\forall m_i, \exists n \in \{1,2,\ldots\}$: $P_i=n*\pi$ and $\exists l \in \{0,1,\ldots\}: \Phi_i=l*\pi$) and the messages are released synchronously with the start of partition windows. This model is typically found in TDMA and master-slave networks that are based on cyclic frameworks.

A helpful definition that applies to both types of partitions is to consider the infinite sequence of partition windows, each characterized by a start and finish instant as in (3).

$$\Pi \equiv \{w_k(w_k^s, w_k^f),\ k=0..\infty\} \tag{3}$$

Then, we can define the capacity of each partition window as $w_k=w_k^f-w_k^s$. Notice that $w_k=w\ \forall_k$ for type 1 partitions. Also, we can define the load of each partition window as in (4), corresponding to the total transmission time of the messages scheduled within w_k. Notice that w_k will be used indistinctively to refer to the $(k+1)^{th}$ partition window as well as to its length.

$$load(w_k) = \sum_{m_i \in w_k} C_i \tag{4}$$

An important aspect is that w_k is normally uncorrelated with $\{C_i, i=1..N\}$ and thus two situations can occur when scheduling messages within a given

partition window, depending on whether an overrun of the partition capacity is allowed or not. If it is not allowed, as in TDMA systems in which there must be a strict temporal isolation between slots, a message m_i is transmitted within window w_k if its transmission completes before or at w_k^f. Otherwise, it is delayed to the following windows. This can lead to the insertion of idle-time at the end of window w_k (Figure 2, top line). In other systems, typically in periodic phases, any message that can start transmission before w_k^f is actually transmitted, thus overrunning the w_k limit (Figure 2, lower line). Despite seeming different, this latter approach can be easily converted to the former inserted idle-time approach by considering an equivalent $w_k^{f'}$ that equals w_k^f plus the longest overrun and an equivalent $w_k' = w_k^{f'} - w_k^s$.

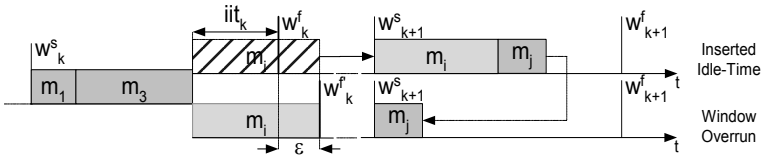

Figure 2. Inserting idle-time (top line) or overrunning the window limit (lower line).

Therefore, whenever idle-time is inserted in window w_k, we have $load(w_k) < w_k$ even when there are more messages ready for transmission. The maximum inserted idle-time depends on the combinations of messages that are scheduled within each window. Calculating its exact value requires building a traffic timeline over an entire macro-cycle. However, it can be upper bounded by the maximum transmission time among all messages or, in some special cases, among a subset (Almeida and Fonseca, 2001).

Finally, we define the network delay nd_i of message m_i as the time span between message release and end of the respective transmission.

3. SYNCHRONOUS RELEASE ANOMALY

The scheduling anomalies that we show in this section refer to the definition of *critical instant*. This is a key concept in schedulability analysis of task or message sets with arbitrary offsets. By definition (Liu and Layland, 1973), the critical instant is such that releasing a message at that instant will result in its largest network delay. When such critical instant can be determined it is sufficient to compute the message network delay for such instant to assess the schedulability of the message set for any phasing condition. Liu and Layland (1973) proved that, for preemptive tasks running on a single non-idling processor, the critical instant happens when a task is released simultaneously with all others (Earliest Deadline First scheduling),

or with all higher priority ones (Rate-Monotonic scheduling). This property also holds for Fixed-Priorities scheduling in general, as well as for time partitioned systems, as long as preemption is allowed. However, for the non-preemptive case, as in our model, such property does not hold.

In this paper we will present two anomalies related to the critical instant. The former one is called *direct synchronous release anomaly* and refers to a situation in which the network delay of a message within a given partition increases when the phasing of the remaining messages is non-synchronous. The latter is called *indirect synchronous release anomaly* and refers to a situation in which the partition capacity is variable and dependent on the effective use of the capacity of another partition that has higher priority in using the bus. This is typical in aperiodic phases with respect to periodic ones, which are normally given higher priority. In this case, the maximum network delay of an aperiodic message might not occur when the message is released together with all the periodic messages. These two counter intuitive phenomena are the motivation for this paper and will be described next.

3.1 DIRECT SYNCHRONOUS RELEASE ANOMALY

Proposition 1: When scheduling a message set M (non-preemptively) within a temporal partition $\Pi(\pi, w)$ of type 1, i.e., with fixed capacity, using fixed-priorities or deadlines-based criteria, the critical instant may not coincide with the synchronous release of the messages in M.

Proof: We prove the proposition with an example. Consider a communication system with a partition $\Pi(\pi, w)$ with capacity $w = 5$ time units and period π>5 time units. Consider a set $M=\{m_i, i=1..5\}$ of periodic messages, sorted by decreasing priorities or later deadlines, with transmission times $C_i=\{2,2,1,3,2\}$ time units and period $T_i>3*\pi$. Let us consider the first 3 availability windows $\{w_0, w_1, w_2\}$

i) When the messages are released synchronously ($\Phi_i=0$, $i=1..5$) messages m_1 to m_3 fit in w_0, and messages m_4 and m_5 fit in slot w_1 (Figure 3). Thus, $nd_4= w_1^s+C_4$ and $nd_5= w_1^s+C_4+C_5$.

ii) Consider now that message m_3 has initial offset $\Phi_3=\pi$. After the transmission of message m_2, messages m_4 and m_5 are ready but are not scheduled since they do not fit in w_0. Then, message m_3 becomes ready at w_1^s and is scheduled before m_4 and m_5 because it has higher priority or an earlier deadline. Message m_4 still fits in w_1 but m_5 is pushed forward to w_2 (Figure 3). Thus, $nd_4= w_1^s+C_3+C_4$ and $nd_5= w_2^s+C_5$.

In this scenario, messages m_4 and m_5 experience a longer response time when $\Phi_i=\{0,0,\pi,0,0\}$ than with the synchronous release $\Phi_i=0$, $i=1..5$ thus proving the proposition. ■

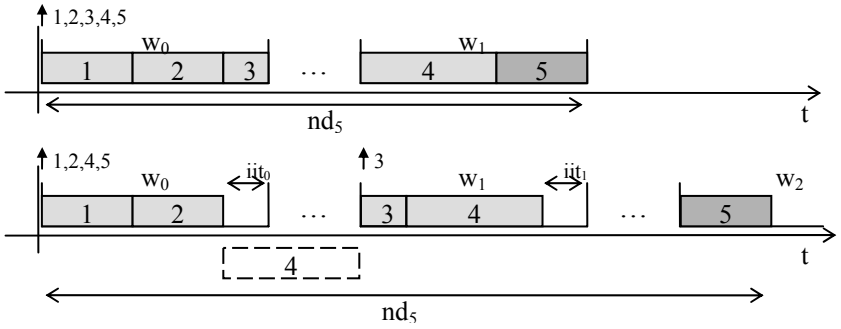

Figure 3. Direct synchronous release anomaly.

The reason for the anomaliy is the idle-time inserted in w_0 and w_1, where the difference w_k-*load*(w_k)>0 is added to the schedule length. This inserted idle-time can vary significantly with just small variations in any C_i. Moreover, it is virtually impossible to compute it exactly for all partition windows w_k unless generating the message schedule for the whole macro-cycle, i.e., for a number of windows equal to *lcm*(T_i), i=1..N. Finally, different messages may exhibit their worst-case network delays for different phasings not just one particular critical instant.

3.2 INDIRECT SYNCHRONOUS RELEASE ANOMALY

Proposition 2: When scheduling a message set M^L (non-preemptively) within a temporal partition $\Pi^L(\pi, w_{min}, w_{max})$ of type 2, i.e., with variable capacity resulting from reclaiming unused capacity from another partition $\Pi^H(\pi, w^H)$ of type 1 that has higher priority, using fixed-priorities or deadlines-based criteria, the critical instant for the set M^L may not coincide with the synchronous release of the set M^H scheduled within the Π^H.

Proof: Again, we will use an example to prove this proposition. Consider the set $M^H=\{m_i^H,\ i=1..5\}$, with $C_i^H=\{2,2,3,2,2\}$ and period $T_i^H>3*\pi$, scheduled within a partition $\Pi^H(\pi=6, w^H=5)$. Now consider a partition $\Pi^L(\pi=6,\ w^L_{min}=1, w^L_{max}=6)$ within which we schedule a message set $M^L=\{m_1^L(C_1^L=2,\ T_1^L>3*\pi)\}$ with just one message. Notice that w^L stretches and shrinks to reclaim the unsued capacity of Π^H, including any inserted idle-time. Let us consider the 3 first partition periods, with the first 3 windows of each partition $\{w_0^H, w_0^L, w_1^H, w_1^L, w_2^H, w_2^L\}$.

When the messages of M^H are released synchronously ($\Phi_i^H=0,\ i=1..5$) messages m_1^H and m_2^H fit in w_0^H, messages m_3^H and m_4^H fit in slot w_1^H and message m_5^H fits in w_2^H (Figure 4). Thus, the effective capacity of partition Π^L in its 3 first windows is $w_0^L=2$, $w_1^L=1$ and $w_2^L=4$.

i) Suppose we release message m_1^L also synchronously, i.e., with Φ_1^L=0. Since C_1^L=w_0^L=2, it fits in w_0^L and its network delay is given by nd_1^L= $w_0^{L,s}$+C_1^L (which equals π).

ii) Now suppose we release message m_1^L with Φ_1^L=π. Since C_1^L>w_1^L=1, it does not fit in w_1^L being pushed to w_2^L. Thus, its network delay is now given by nd_1^L= $w_2^{L,s}$+C_1^L-π (which equals π+C_5^H+C_1^L).

Since nd_1^L is larger in the second case, i.e., when the message is released in an instant different from the synchronous release of the M^H set in the partition Π^H the proposition is proved. ■

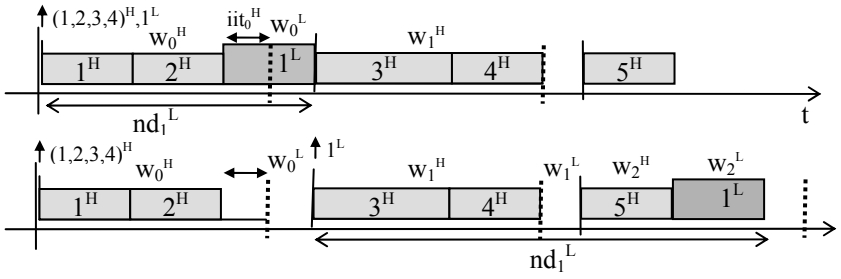

Figure 4. Indirect synchronous release anomaly.

This anomaly is also caused by idle-time inserted within Π^H, whose effect is propagated to Π^L because of the reclaiming mechanism. If M^L included more messages we would obtain a superposition of both anomalies, i.e., the effect of the idle-time inserted directly in Π^L because of the scheduling of set M^L, and the idle-time inserted in Π^H caused by the scheduling of set M^H and propagated to Π^L via the reclaiming mechanism.

Moreover, it is also virtually impossible to determine the exact phasing for a message within Π^L that will cause its worst-case network delay without actually building the schedule of M^H within Π^H for the respective macro-cycle. Finally, there is not one single phasing that causes the worst-case network delay for all messages in M^L.

4. WORST-CASE NETWORK DELAY

In the previous section we have seen that using inserted idle-time, either directly or indirectly, within temporally partitioned networks may lead to situations in which we lose the ability to determine an exact critical instant. This has a drastic consequence in terms of schedulability analysis, and it shows that it is not possible, in the general case, to determine exact worst-case network delays under the referred conditions.

However, the duration of the anomalies can be upper bounded and so can the network delays. In recent years, a few analysis were developed that already overcome these anomalies. On the other hand, the anomalies were not recognized as such and were not formally presented, thus not illustrating their full impact. Other analysis were presented that fail in the cases shown in this paper, because the anomalies were not taken into account.

For example, Almeida *etal.* (2002) present a simple approach for the direct anomaly considering the maximum idle-time being inserted every window and noticing that its magnitude cannot be larger than the longest message. This corresponds to considering one extra message $m_{iit}(C_{iit}=C_{max}, T_{iit}=\pi)$, i.e., occurring every partition window and with a length equal to the transmission time of the longest message in the set. This was proposed for scheduling synchronous messages under EDF within FTT-CAN. They also present another approach that is more efficient with utilization-based schedulability tests for rate-monotonic scheduling. It consists in inflating the transmission times of the messages to $C_i*w/(w-C_{max})$. It is also shown that, with fixed-priorities scheduling, the maximum inserted idle-time can be determined from within a subset of messages. With the adapted C_i and the referred scheduling model, any existing analysis for preemptive fixed-priorities scheduling can be used in this case but always generating sufficient non-necessary schedulability conditions (Almeida and Fonseca, 2001).

For the indirect anomaly, it can be avoided when analysing the worst-case network delay by not reclaiming the capacity left free within the higher priority partition whenever idle-time could have been inserted. Thus, whenever $w_k \leq load(w_k) \leq w_k - C_{max}$ the analysis considers $load(w_k)=w_k$. This has been proposed in (Almeida *etal.*, 2002) to analyse the network delay of asynchronous messages in FTT-CAN. With these approaches, for the direct and indirect anomalies, the worst-case network delay obtained considering the synchronous release will always be longer than the network delays occurring at run-time with any phasing, thus resulting in a safe upper-bound.

Examples of analysis that may fail to provide a safe upper-bound for the worst-case network delay are the timeline analysis for cyclic frameworks (Almeida and Fonseca, 2001), the analysis for PROFIBUS in (Cavalieri *etal.*, 2002) as noted in (Ferreira, 2005), and the analysis for WorldFIP in (Tovar and Vasques, 2000).

5. CONCLUSION

Scheduling messages non-preemptively within temporaly-partitioned networks, e.g. in TDMA, token-passing or multi-phase cyclic buses, will generally lead to the insertion of idle-time at the end of the partitions, either

slots or phases. This inserted idle-time may create scheduling anomalies in which the critical instant is not coincident with the synchronous release of the scheduled messages. In this paper we exposed and formalized two anomalies of this kind, one related to scheduling messages within a partition of constant width and another one within a partition of variable width. The paper also showed existing approaches for traffic schedulability analysis that are robust with respect to the anomalies, as well as approaches that may fail because the anomalies were not considered.

REFERENCES

L. Almeida, J.A. Fonseca (2001). "Analysis of a Simple Model for Non-Preemptive Blocking-Free Scheduling". In Proceedings of the 13th Euromicro Conference on Real-Time Systems, Delft, The Netherlands.

L. Almeida, P. Pedreiras, J.A. Fonseca (2002). "The FTT-CAN Protocol: Why and How". In IEEE Transactions on Industrial Electronics, 49(6).

L. Almeida, P. Pedreiras (2004). "Scheduling within temporal partitions: response-time analysis and server design", ACM Int Conf on Embedded Software (EMSOFT 2004), Pisa.

G. Buttazzo (2002). "Scalable Applications for Energy-Aware Processors". 2nd Int. Conf. on Embedded Software (EMSOFT 2002), Grenoble, France, LNCS 2491, Springer-Verlag.

S. Cavalieri, S. Monforte, E. Tovar, F. Vasques (2002). Multi-Master Profibus-DP Modelling and Worst-Case Analysis Based Evaluation. 15th IFAC World Congress, Barcelona, Spain.

Y.-S. Chen, L.-P. Chang, T.-W. Kuo, A.K. Mok (2005). "Real-time task scheduling anomaly: observations and prevention", ACM Symp on Applied Computing, Santa Fe, New Mexico.

R.I. Davis, A. Burns (2005). "Hierarchical Fixed Priority Scheduling", Proceedings of the 26th IEEE Real-Time Systems Symposium (RTSS). Miami, USA.

L. Ferreira (2005). "A Multiple Logical Ring Approach to Real-time Wireless-enabled PROFIBUS Networks". PhD Thesis, Universidade do Porto, Portugal.

R. Graham (1976). "Bounds on the performance of scheduling algorithms". In E. G. Coffman, Jr., ed, Computer and Job-Shop Scheduling Theory. John Wiley & Sons, New York.

Giuseppe Lipari, Enrico Bini (2004). "A Methodology for Designing Hierarchical Scheduling Systems". In Journal of Embedded Computing 1(2).

C. L. Liu, J. W. Layland (1973). "Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment". In Journal of the ACM, 20(1).

A. Mok (2000). "Scalability of Real-Time Applications". Keynote address at the 7th International Conference on Real-Time Computing Systems and Applications.

I. Shin, I. Lee (2003). "Periodic Resource Model for Compositional Real-Time Guarantees". Proceedings of 24th IEEE Real-Time Systems Symposium (RTSS), Cancun, Mexico.

J. A. Stankovic, M. Spuri, M. Di Natale, G. C. Buttazzo (1995). "Implications of Classical Scheduling Results for Read-Time Systems". In Computer, June, 1995, pp. 1625.

E. Tovar, F. Vasques (2000). "Distributed Computing for the Factory-floor: a Real-Time Approach Using WolrdFIP Networks", Computers in Industry, 44(1), Elsevier Science.