MODELAGEM E VERIFICAÇÃO FORMAL DOS TEMPOS DE RESPOSTA DE MENSAGENS CAN COM *OFFSETS* ESTÁTICOS

Jadsonlee da Silva Sá*, Antonio Marcus Nogueira Lima†, Angelo Perkusic†, José Sérgio da Rocha Neto†

*Doutorando do Programa de Pós-Graduação em Engenharia Elétrica da UFCG - COPELE
Colegiado de Engenharia de Computação da UNIVASF
Av. Antonio Carlos Magalhães, 510 - Santo Antonio - 48902-300
Juazeiro – BA – Brasil

†Departamento de Engenharia Elétrica da UFCG
Laboratório de Instrumentação Eletrônica e Controle
Av. Aprígio Veloso, 882 - Bodocongó - 58109-970
Campina Grande – PB – Brasil

Emails: jadsonlee@ee.ufcg.edu.br, amnlima, perkusic, zesergio@dee.ufcg.edu.br

Abstract— This paper presents an exact analysis of the response times of CAN messages with static offsets using the technique of model checking. The model was developed for timed automata in the UPPAAL tool. Two case studies for different settings offsets were presented and the results compared with traditional analysis of response times of CAN messages. Then, it was found that the use of static offsets significantly reduce response times of the messages.

Keywords— CAN, Response Time Analysis, Offset, Model-Checking

Resumo— Neste trabalho apresenta-se uma análise exata dos tempos de resposta de mensagens CAN com *offsets* estáticos utilizando a técnica de verificação de modelos. O modelo foi desenvolvido em autômatos temporizados na ferramenta UPPAAL. Dois estudos de caso para diferentes configurações de *offsets* foram apresentados e os resultados comparados com a análise tradicional de tempos de resposta de mensagens CAN. Então, foi verificado que a utilização de *offsets* estáticos reduzem significantemente os tempos de resposta das mensagens.

Palavras-chave— CAN, Análise de Tempos de Resposta, *Offset*, Verificação de Modelos

1 Introdução

Controller Area Network (CAN) (Bosch, 1991) é uma rede de comunicação utilizada em aplicações envolvendo automação e controle embarcado em tempo real.

Uma rede CAN é constituída por nós que trocam mensagens via um meio de comunicação compartilhado (barramento) com largura de banda limitada. As mensagens possuem níveis de prioridades fixas que são utilizadas no mecanismo de controle de acesso ao meio para resolver as possíveis colisões no meio de comunicação. Esse mecanismo garante um comportamento temporal previsível do trafégo na rede sob condições ideais (e.g. sem erros de transmissão). Assim, é possível determinar os tempos de resposta no pior caso de mensagens periódicas utilizando, por exemplo, técnicas analíticas (Tindell et al., 1995; Davis et al., 2007), técnicas de simulação (RTaW, 2012) e técnicas de verificação de modelos (Waszniowsky et al., 2009; Sá et al., 2010; Sá et al., 2012).

A ideia utilizada para determinar os tempos de resposta no pior caso, em particular, nas técnicas analíticas, consiste em analisar o cenário de pior caso de cada mensagem que inicia no chamado instante crítico (Davis et al., 2007). A possibilidade da ocorrência desse cenário, baseia-se no fato de que a primeira liberação de cada mensagem ocorre em um instante de tempo arbitrário chamado de *offset* dinâmico. Na prática, o cenário de pior caso nem sempre acontece para cada mensagem. Entretanto, esses tempos são considerados durante o projeto do sistema por questões de segurança.

O problema da abordagem com *offset* dinâmico é que os tempos de resposta das mensagens com baixa prioridade aumentam muito quando a carga da rede aumenta. Uma solução para resolver esse problema é escalonar as mensagens com *offsets* estáticos de modo a evitar o cenário de instante crítico. Nessa abordagem, o instante da primeira liberação de cada mensagem é conhecido e possui uma referência de tempo comum entre os nós da rede.

Alguns estudos sobre a análise de tempos de resposta de mensagens CAN com *offsets* estáticos foram desenvolvidos. Em alguns trabalhos utilizando técnicas analíticas ((Szakaly, 2003), (Du and Xu, 2009), (Chen et al., 2011)), as análises desenvolvidas são de baixa complexidade computacional, mas os resultados são aproximados. Grenier et al. (2008) desenvolveram um algoritmo para atribuição de *offsets* estáticos e verificaram a sua eficiência utilizando a ferramenta de simulação NETCAR-Analyser (Braun et al., 2007) para determinar os tempos de resposta exatos no pior caso. Basicamente, a ferramenta realiza uma simulação durante um ou mais intervalos de tempo

chamados de hiperperíodo[1], que equivale ao mínimo múltiplo comum de todos os períodos das mensagens (Audsley, 1991), e verifica nesse intervalo o maior tempo de resposta de cada mensagem. Recentemente, Yomsi et al. (2012) adaptaram o trabalho de Palencia and Harbour (1998) para desenvolver uma análise dos tempos de resposta no pior caso para mensagens CAN com *offset* estático. Porém, a análise não é exata.

Neste trabalho, desenvolvemos um modelo em autômatos temporizados (Alur and Dill, 1994) utilizando a ferramenta UPPAAL (Berhmann et al., 2004) para determinar, via a técnica de verificação de modelos (Larsen et al., 1995), os tempos de resposta exatos no melhor e pior caso de mensagens CAN com *offsets* estáticos. O modelo é uma adaptação do modelo de Sá et al. (2012), que trata de mensagens CAN com *offsets* dinâmicos e erros de transmissão. No modelo, enfatizamos a necessidade de atribuir prioridades aos autômatos, de modo a modelarmos corretamente a rede CAN e evitar inversões de prioridade no processo de disputa do meio de comunicação durante a verificação do modelo. Então, realizamos alguns estudos de caso e comparamos nossos resultados com os resultados obtidos pela análise de Davis et al. (2007) para *offsets* dinâmicos.

Apesar da técnica de simulação determinar tempos de resposta exatos para o caso apresentando neste trabalho (veja, por exemplo, as ferramentas NETCAR-Analyser (RTaW, 2009) e RTaW-Sim (RTaW, 2012)), acreditamos que a técnica de verificação de modelos e suas extensões (e.g. verificação de modelos estatísticos (David et al., 2011) e (Bulychev et al., 2012)) evoluiram significantemente, de forma a abrager um maior número de aplicações complexas. Além disso, essa técnica permite também verificar propriedades funcionais e caracterizar comportamentos internos do sistema tais como, desvios dentro do código dependentes de dados e *loops*, ao contrário das outras técnicas.

Na Seção 2, apresenta-se as principais características da rede CAN, e uma notação e modelo das mensagens. Na Seção 3, define-se a técnica de verificação de modelos. Na Seção 4, apresenta-se a definição e sintaxe de autômatos temporizados. Na Seção 5, apresenta-se a ferramenta UPPAAL e sua linguagem de especificação. Na Seção 6, descreve-se o modelo em autômatos temporizados da rede CAN. Na Seção 7, apresentam-se os estudos de caso e a análise dos resultados obtidos, e na Seção 8 as conclusões.

2 Rede CAN

Uma rede CAN é constituída por nós conectados a um barramento serial que trocam mensagens a uma taxa de transmissão de até 1 M$bits$/s. Um nó CAN é formado por uma unidade de processamento (UP) e pelos dispositivos de rede (controlador e *transceiver* CAN). Para transmitir uma mensagem[2], uma tarefa de aplicação, executada na UP, prepara os dados a serem transmitidos, armazena-os no *buffer* de transmissão e envia uma requisição de transmissão ao controlador CAN. Assim que o barramento estiver livre, o controlador tentará transmitir a mensagem. Cada mensagem possui em seu cabecário um número único (identificador) que indica o nível de prioridade de acesso ao meio.

Devido o acesso ao meio ser assíncrono, é possível que mais de um nó tente transmitir uma mensagem simultaneamente. Para controlar as possíveis colisões durante o acesso ao barramento, CAN utiliza o mecanismo CSMA/DCR (*Carrier-Sense Multiple Access/Deterministic Collision Resolution*). Nesse mecanismo, os nós tentam transmitir uma mensagem somente se o barramento estiver livre. Então, todos os nós com mensagens prontas para serem transmitidas enviam um *bit* dominante (SOF - *Start Of Frame*) e iniciam o processo de disputa pelo acesso total do meio, chamado de processo de arbitragem. O resultado da arbitragem é determinada pela comparação dos *bits* dos identificadores das mensagens baseado no mecanismo *wired-AND*[3]. Portanto, a mensagem que tiver o identificador com menor valor numérico (maior prioridade) vencerá o processo de arbitragem e transmitirá o restante da mensagem. Os nós que perderam a disputa e outros possíveis nós, tentarão transmitir suas mensagens assim que o barramento estiver livre novamente.

2.1 Notação e Modelo das Mensagens CAN

Considere uma rede CAN com um único barramento e um conjunto com n mensagens periódicas m_0, m_1,..., m_{n-1}. Uma mensagem m_i, onde i assume valores de 0 a $n-1$, consiste de um número infinito de instâncias $k \in \mathbb{Z}^+ = \{0, 1, 2, 3, ...\}$.

Cada mensagem possui uma tarefa periódica relacionada τ_i com período T_i, que armazena a mensagem no *buffer* de transmissão do controlador CAN do seu respectivo nó. A primeira instância dessa tarefa é iniciada em um instante chamado de *offset* estático O_i, que assume algum valor fixo no intervalo $[0, T_i[$[4]. Então, uma instância k de uma mensagem é ativada a cada instante $a_i = O_i + kT_i$ e liberada (pronta para transmissão) assim que a tarefa relacionada armazenála no *buffer* de transmissão. Essa tarefa gasta uma quantidade de tempo igual a C_i, chamado de *atraso* de armazenamento ou *jitter* de liberação da

[1] É o período em que a escala do conjunto de mensagens se repete ao longo do tempo.

[2] Consideraremos apenas mensagens de dados e todos os nós são sempre ativos. Para mais detalhes, veja a referência (Bosch, 1991).

[3] O nível lógico '0' sobrepõe o nível lógico '1' no barramento.

[4] Devido a natureza periódica do escalonamento, um *offset* $O_i >= T_i$ é equivalente a $O_i \ mod \ T_i$, onde *mod* é o operador módulo (Goossens, 2003).

mensagem. Por simplicidade consideraremos que $C_i = 0$.

Além disso, toda mensagem possui: um tempo de transmissão no pior caso C_i^m; um *deadline* D_i, onde $D_i \leq T_i$, e um nível de prioridade fixa p_i. Assume-se que o nível de prioridade decresce de acordo com o aumento do número da mensagem, ou seja, a mensagem m_0 possui a maior prioridade e a mensagem m_{n-1} a menor prioridade.

3 Técnica de Verificação de Modelos

A verificação de modelos é uma técnica que permite verificar propriedades de um sistema de forma automática por meio de ferramentas computacionais (Larsen et al., 1995).

Para verificar as propriedades do sistema, a ferramenta computacional recebe como entrada a descrição do modelo do sistema seguindo algum método formal (e.g. autômatos temporizados) e uma descrição da especificação dos requisitos expressas em fórmulas de lógica temporal. Então, a ferramenta utiliza algoritmos simbólicos eficientes para realizar uma pesquisa exaustiva do espaço de estados do modelo e verificar se as propriedades são satisfeitas para um determinado estado inicial do modelo.

4 Autômatos Temporizados

Autômatos temporizados são um tipo de autômatos de estados finitos que consideram quantitativamente o tempo na transição dos estados. A progressão do tempo ocorre de acordo com um conjunto finito de relógios sincronizados na mesma taxa, que utilizam o modelo de tempo denso[5]. Os valores dos relógios podem ser comparados com números inteiros e podem ser resetados.

Um conjunto C de relógios possui algumas restrições $B(C)$, que consistem de equações na forma $x \sim c$ ou $x - y \sim c$, onde: $x, y \in C$; $c \in \mathbb{N} = \{0, 1, 2, ...\}$; e a relação $\sim$ é um dos seguintes símbolos $\{\leq, \geq, =, <, >\}$.

Formalmente, podemos definir um autômato temporizado $\mathcal{A}$ pela sêxtupla $\{L, l_0, C, A, E, I\}$, onde: L é o conjunto de lugares; $l_0 \in L$ é o lugar inicial; C é o conjunto de relógios; A é um conjunto de ações; $E \subseteq L \times A \times B(C) \times 2^C \times L$ é um conjunto de bordas (transições) entre os lugares com uma ação a (canal de sincronização), um guarda g (restrição de tempo) e um conjunto de relógios r a serem resetados; e $I : L \rightarrow B(C)$ é a atribuição de invariantes de estado (restrições de tempo) aos lugares.

5 Ferramenta UPPAAL

UPPAAL é uma ferramenta para modelagem, simulação e verificação de modelos de sistemas em tempo real baseado na teoria de autômatos temporizados (Berhmann et al., 2004).

[5]Os valores dos relógios são números reais.

No UPPAAL, um sistema é modelado como uma rede de autômatos temporizados em paralelo. Cada autômato é constituído por lugares (representados por circulos) e bordas ou transições (representadas por arcos). A linguagem de modelagem é estendida, com relação a linguagem de autômatos temporizados, de modo a considerar constantes, variáveis inteiras limitadas, *arrays*, funções, diferentes tipos de lugares e canais de sincronização, entre outros.

5.1 Linguagem de Especificação de Requisitos

A linguagem de especificação do UPPAAL é baseada em um subconjunto da lógica TCTL(*Timed Computation Tree Logic*) (Henzinger, 1994) e é composta pelos seguintes operadores: A (para todo caminho), E (existe um caminho), [] (sempre), <> (futuramente), $\rightarrow$ (conduz a), *and*, *or*, *not*, *imply* e *deadlock*. A partir dessa linguagem de especificação, é possível verificar propriedades de alcançabilidade, segurança, vivacidade, vivacidade limitada e avaliação de desempenho.

Recentemente, os operadores *sup* (*supremum*) e inf (*infimum*) foram integrados para verificar propriedades de vivacidade limitada. Esses operadores serão utilizados para determinar os tempos de resposta no melhor e pior caso.

6 Modelos em Autômatos Temporizados

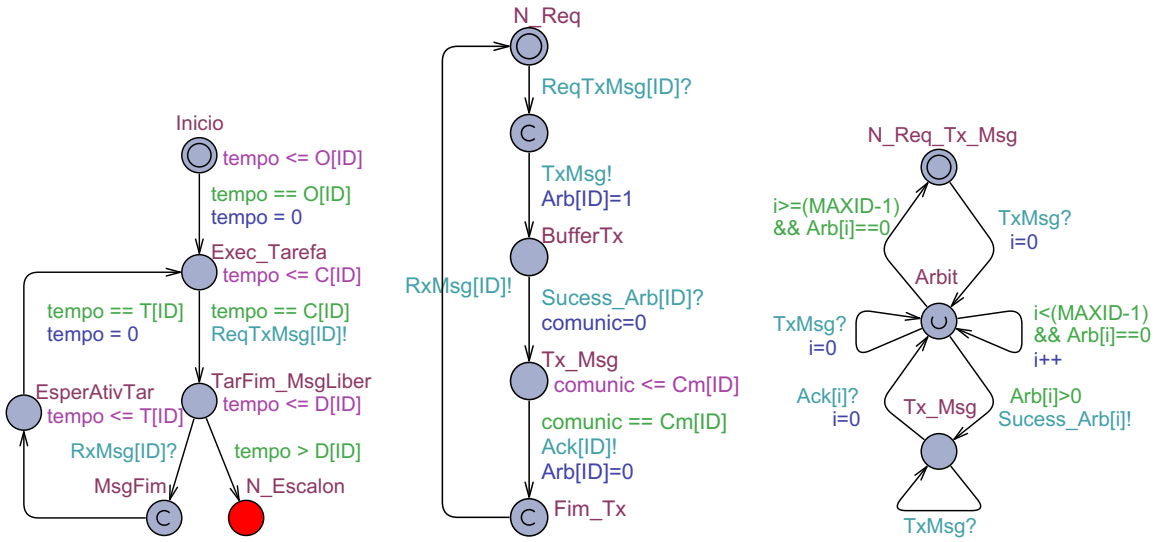
Nesta seção, apresenta-se o conjunto de autômatos temporizados utilizado para modelar a rede CAN de acordo com a notação e o modelo de mensagens apresentado na Seção 2.1. O modelo é constituído por três autômatos (Figura 1): Tarefa, Controlador CAN e Arbitragem. Para cada mensagem do sistema, a ferramenta UPPAAL gera automaticamente um autômato Tarefa e um autômato Controlador CAN. Apenas um autômato Arbitragem é gerado para o sistema.

6.1 Autômato Tarefa

A execução do sistema é iniciada no lugar *Inicio* do autômato Tarefa (Fig. 1(a)). Esse autômato possui um relógio local *tempo*, utilizado para contar o *offset* estático da tarefa associada a mensagem, e utilizado implicitamente para marcar o período $T[ID]$ e o tempo de resposta da mensagem. ID é uma variável utilizada para indicar a prioridade e o identificador da mensagem.

A primeira instância da tarefa relacionada a mensagem ID é liberada no lugar *Inicio* com um *offset* igual a $O[ID]$. As próximas instâncias são iniciadas periodicamente no lugar *EsperAtivTar*. O *offset* é implementado pela invariante de estado *tempo <= O[ID]* e o guarda *tempo* == $O[ID]$.

Após iniciada, a tarefa é executada (lugar *Exec_Tarefa*) durante $C[ID]$ unidades de tempo (consideramos que $C[ID] = 0$) e, em seguida, requisita a transmissão da mensagem com identificador *ID* modelada pelo canal *ReqTxMsg[ID]!*. Então, o autômato vai para o lugar *TarFim_MsgLiber*



(a) Tarefa. (b) Controlador CAN. (c) Arbitragem.

Figura 1: Conjunto de autômatos temporizados de uma rede CAN.

e aguarda a finalização da transmissão da mensagem. A finalização é indicada por um sinal via o canal *RxMsg[ID]?* enviado pelo autômato Controlador CAN (Fig. 1(b)). Após receber esse sinal, o autômato permanece no lugar *EsperAtivTar* até iniciar um novo período. Caso o valor do relógio *tempo* seja maior que o *deadline* $D[ID]$ enquanto o autômato estiver no lugar *TarFim_MsgLiber*, o autômato vai para o lugar *N_Escalon*, indicando que o *deadline* não foi satisfeito.

6.2 Autômato Controlador CAN

O autômato Controlador CAN (Fig. 1(b)) permanece no seu lugar inicial *N_Req* até receber uma requisição de transmissão de mensagem via o canal *ReqTxMsg[ID]?*. Quando isso ocorrer, um sinal via o canal *TxMsg!* é enviado para o autômato arbitragem (Fig. 1(c)) e o *array Arb[ID]* indexado com o valor do identificador da mensagem é ajustado para '1', indicando que essa mensagem está pronta para entrar no processo de arbitragem. O autômato permanece no lugar *BufferTx* até receber um sinal via o canal *Sucess_Arb[ID]?*. Esse sinal indica que a mensagem venceu o processo de arbitragem e pode continuar a transmissão. Então, o relógio *comunic* é zerado e o lugar *Tx_Msg* é alcançado. Após o relógio *comunic* contar *Cm[ID]* unidades de tempo, a mensagem é totalmente transmitida. Um sinal de reconhecimento é enviado pelo canal *ack[ID]!* ao autômato Arbitragem, e o *array Arb[ID]* é zerado. Em seguida, um sinal via o canal *RxMsg[ID]!* é enviado para o autômato Tarefa que requisitou a transmissão da mensagem e o autômato retorna para o lugar inicial *N_Req*.

6.3 Autômato Arbitragem

O controle de acesso ao meio da rede CAN é modelado pelo autômato Arbitragem. Esse autômato permanece no lugar inicial *N_Req_Tx_Msg* até receber um sinal via o canal *TxMsg?* de um dos autômatos Controlador CAN. Quando isso ocorre, a variável i é zerada e o autômato passa para o lugar urgente *Arbit*. Nesse lugar, a mensagem com maior prioridade que requisitou uma transmissão é escolhida para ser transmitida. O *array Arb[i]* é verificado item por item, com o índice i variando de 0 à *MAXID-1*, onde *MAXID* equivale ao número de identificadores (mensagens) do sistema. Os identificadores das mensagens são ordenados de forma crescente. A mensagem com maior prioridade possui $ID = 0$, e a mensagem com menor prioridade $ID = MAXID\text{-}1$. Então, o primeiro elemento do *array Arb[i]* com valor maior que 0 vencerá o processo de arbitragem, e um sinal via o canal *Sucess_Arb[i]* será enviado para o autômato Controlador CAN. O autômato permanece no lugar *Tx_Msg* até receber o sinal de reconhecimento via *ack[ID]?*, indicando que a mensagem foi transmitida. Então, a variável i é zerada e o autômato retorna para o lugar *Arbit*, onde verifica se existe alguma mensagem pendente para ser transmitida. Caso não exista, o autômato retorna para o seu lugar inicial *N_Req_Tx_Msg*.

6.4 Atribuição de Prioridade aos Autômatos

Durante os testes do modelo, verificamos em algumas situações a ocorrência de inversões de prioridade nas mensagens CAN. Para ilustrar esse problema, considere o conjunto de mensagens da Tabela 1 utilizado no trabalho de Davis et al. (2007), onde $\tau_{bit} = 8$ (Taxa de transmissão igual a 125 Kbps). Considere também que o *offset* de cada mensagem e o tempo de computação são iguais a zero.

De acordo com os atributos definidos, veja

[3]Os relógios locais param de contar.

que as transições *Inicio* → *Exec_Tarefa* → *Tar-Fim_MsgLiber* e *N_Req* → *Buffer_Tx* → *Tx_Msg*, respectivamente, dos autômatos Tarefa e Controlador CAN de cada mensagem, além das transições *N_Req_Tx_Msg* → *Arbit* → *Tx_Msg*, devem ocorrer no instante de tempo 0. Para esse cenário, observamos, utilizando o simulador da ferramenta, que o lugar *Tx_Msg* do autômato Controlador CAN da mensagem de menor prioridade, poderá ser alcançado antes que as mensagens de maior prioridade nos seus respectivos autômatos Controlador CAN, ou seja, essa mensagem poderá ser transmitida antes das mensagens de maior prioridade indicarem que estão prontas para entrar no processo de arbitragem.

Tabela 1: Atributos do conjunto de mensagens utilizado no artigo de Davis et al. (2007) (em μs).

ID	T	C_m	D
0	2500	$125\tau_{bit}$	2500
1	3500	$125\tau_{bit}$	3500
2	3500	$125\tau_{bit}$	3500

Para resolver esse problema, utilizamos a capacidade da ferramenta de atribuir prioridades aos autômatos. Dessa forma, durante um cenário onde mais de uma transição de diferentes autômatos estiverem habilitadas, a ferramenta executará as transições seguindo os níveis de prioridade de cada autômato. Para definir a prioridade de cada autômato, tomamos como referência a prioridade de acesso ao meio de cada mensagem, e seguimos as relações de precedência entre os autômatos da mesma mensagem, conforme indicado a seguir, onde: A, CON e TAR são, respectivamente, os autômatos Arbitragem, Controlador CAN e Tarefa, e n o número de mensagens:

$$A < CON_{n-1} < CON_{n-2} < ... < CON_0 < TAR_{n-1} < TAR_{n-2} < ... < TAR_0$$

Definimos o autômato A com a menor prioridade, de modo a garantir que a transição *Arbit* → *Tx_Msg*, que decide qual mensagem será transmitida, nunca ocorrerá antes de todos os autômatos Controlador CAN alcançarem o lugar *BufferTx*, ou seja, antes de todas as mensagens prontas para serem transmitidas serem armazenadas no *buffer* de transmissão em um mesmo instante de tempo.

Para verificar o comportamento temporal do modelo, determinamos graficamente os tempos de resposta no melhor e pior caso do conjunto de mensagens da Tabela 1, examinando a escala de dois hiperperíodos (Figura 2) para duas diferentes configurações de *offset*: $O_0 = O_1 = O_2 = 0$ (*offsets* iguais - Figura 2(a)); e $O_0 = 0$, $O_1 = 1\ ms$ e $O_2 = 2\ ms$ (*offsets* diferentes - Figura 2(b)). Então, comparamos com os resultados obtidos pela verificação das propriedades $sup\{TAR_i.MsgFim\} : TAR_i.tempo$ e $inf\{TAR_i.MsgFim\} : TAR_i.tempo$, com e sem a atribuição de prioridades, onde $i = 0$, 1 e 2. Essas propriedades podem ser lidas da seguinte forma: "qual o maior/menor valor do relógio *tempo*, se o lugar *Msg_Fim* do autômato TAR_i foi alcançado?".

A verificação dessas propriedades foi realizada pela ferramenta UPPAAL versão 4.1.15 em um *notebook* com processador Intel core i3-350M com 2 GB de memória física e sistema operacional Windows 7 de 32 *bits*. Os resultados para o pior caso e melhor caso estão, respectivamente, nas Tabelas 2 e 3, para a configuração *offsets* iguais. Para a configuração *offsets* diferentes, os resultados para o pior caso e melhor caso estão, respectivamente, nas Tabelas 4 e 5.

Tabela 2: Tempos de resposta no pior caso (em μs) obtidos graficamente, e por verificação de modelos sem e com prioridade - *Offsets* iguais.

ID	Grafic.	S/Prior.	C/Prior.
0	1500	**2000**	1500
1	2000	**3000**	2000
2	3500	3500	3500

Tabela 3: Tempos de resposta no melhor caso (em μs) obtidos graficamente, e por verificação de modelos sem e com prioridade - *Offsets* iguais.

ID	Grafic.	S/Prior.	C/Prior.
0	1000	1000	1000
1	1000	1000	1000
2	2500	**1000**	2500

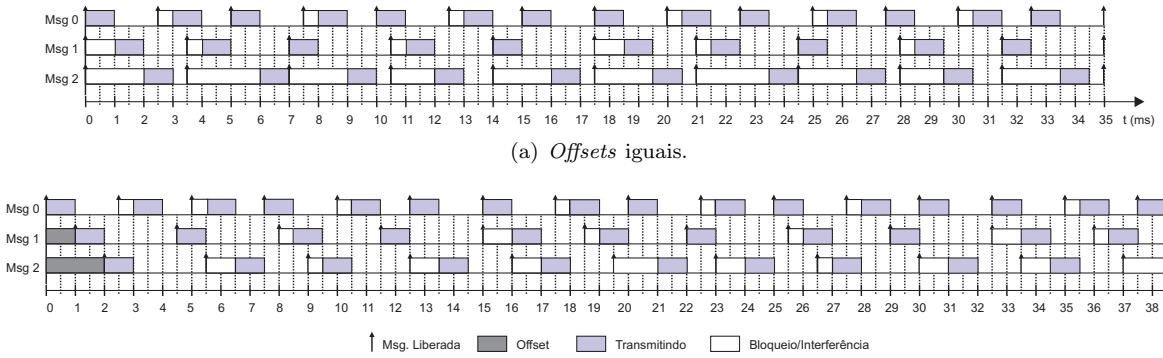
Tabela 4: Tempos de resposta no pior caso (em μs) obtidos graficamente, e por verificação de modelos sem e com prioridade - *Offsets* diferentes.

ID	Grafic.	S/Prior.	C/Prior.
0	1500	**2000**	1500
1	2000	2000	2000
2	2500	2500	2500

Tabela 5: Tempos de resposta no melhor caso (em μs) obtidos graficamente, e por verificação de modelos sem e com prioridade - *Offsets* diferentes.

ID	Grafic.	S/Prior.	C/Prior.
0	1000	1000	1000
1	1000	1000	1000
2	1000	1000	1000

Observe que os tempos de resposta atribuindo prioridades aos autômatos são idênticos aos obtidos graficamente, ao contrário dos tempos de resposta obtidos com os autômatos sem prioridade (com exceção dos tempos de resposta no melhor caso para *offsets* diferentes (Tabela 5)). Então, consideramos que a política de atribuição de prioridades aos autômatos do modelo conduz ao comportamento correto da rede, evitando as inversões de prioridade. Outros experimentos para diferentes configurações de *offsets* foram realizados e os resultados foram consistentes.



(a) *Offsets* iguais.

(b) *Offsets* diferentes.

Figura 2: Escala do conjunto de mensagens com *offsets* estáticos.

7 Estudos de Caso e Análise dos Resultados

Nesta seção, apresenta-se os estudos de casos para um conjunto de 12 mensagens de uma rede CAN instalada no protótipo de um veículo da PSA Peugeot-Citroën (Navet et al., 2000), e para um conjunto de 17 mensagens de uma simplificação feita por Tindell et al. (1995) de um *benchmark* SAE (*Society Automotive Engineering*). Os resultados para duas diferentes configurações de *offsets* estáticos foram comparados com os resultados da análise de Davis et al. (2007) para *offsets* dinâmicos. Utilizamos as mesmas propriedades e o mesmo ambiente computacional descrito na Seção 6.4 para determinar os tempos de resposta.

7.1 Mensagens - PSA Peugeot-Citroën

Os atributos para esse conjunto de mensagens estão indicados na Tabela 6. A taxa de transmissão é de 250 Kbps ($\tau_{bit} = 4$) e o fator de utilização da rede é igual a 21,5%.

Tabela 6: Atributos do conjunto de mensagens da PSA Peugeot-Citroën (em μs).

ID	T	C_m	D
0	10000	$135\tau_{bit}$	10000
1	14000	$85\tau_{bit}$	14000
2	20000	$85\tau_{bit}$	20000
3	15000	$75\tau_{bit}$	15000
4	20000	$105\tau_{bit}$	20000
5	40000	$105\tau_{bit}$	40000
6	15000	$95\tau_{bit}$	15000
7	50000	$105\tau_{bit}$	50000
8	20000	$95\tau_{bit}$	20000
9	100000	$125\tau_{bit}$	100000
10	50000	$105\tau_{bit}$	50000
11	100000	$65\tau_{bit}$	100000

Na configuração *offset* 1, todos os *offsets* das mensagens são iguais a zero, e na configuração *offset* 2, utilizamos a seguinte regra para atribuição de *offsets*: $O_0 = 0$ e $O_{ID} = O_{ID-1} + C^m_{ID-1}$, resultando nos seguintes valores em μs: $O_0 = 0$, $O_1 = 540$, $O_2 = 880$, $O_3 = 1220$, $O_4 = 1520$, $O_5 = 1940$, $O_6 = 2360$, $O_7 = 2740$, $O_8 = 3160$, $O_9 = 3540$, $O_{10} = 4040$, $O_{11} = 4460$.

O tempo de verificação para cada configuração foi de aproximadamente 30 s. As curvas dos tempos de resposta no melhor e pior caso versus o identificador da mensagem estão representadas na Figura 3. Observe que para cada configuração de *offset*, os tempos de resposta no pior caso (curvas vermelha e azul com circulos) são menores que os tempos de resposta com *offset* dinâmico (curva preta). Na configuração *offset* 2, obtemos tempos de resposta muito baixos comparados com as outras configurações. Esse resultado deve-se aos valores dos *offsets* e, principalmente, ao baixo fator de utilização da rede. Então, foi possível distribuir ao longo do tempo a carga de trabalho na rede. Para a mensagem com identificador 11 (menor prioridade), o tempo de resposta no pior caso para a configuração *offset* 2 foi mais de 85% menor que os tempos de resposta no pior caso com *offset* dinâmico e com a configuração *offset* 1.

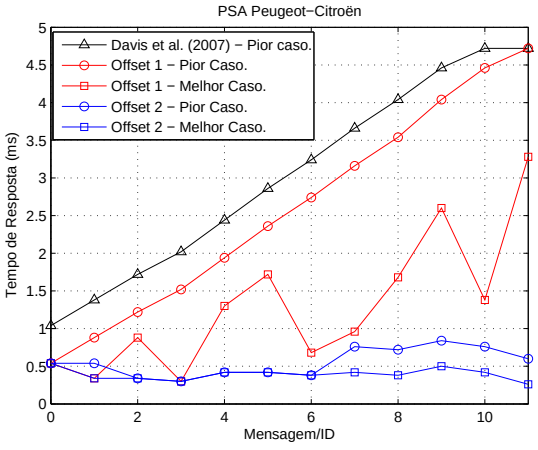


Figura 3: Curvas dos tempos de resposta no melhor e pior caso do conjunto de mensagens da PSA Peugeot-Citroën.

7.2 Benchmark *SAE*

Neste estudo de caso, o fator de utilização da rede é igual a 85%, que é considerado alto. Os atributos para esse conjunto de mensagens estão indicados

na Tabela 7. A taxa de transmissão é de 125 Kbps ($\tau_{bit} = 8$).

Tabela 7: Atributos do conjunto de mensagens do *benchmark* SAE (em μs).

ID	T	C_m	D
0	1000000	$65\tau_{bit}$	5000
1	5000	$75\tau_{bit}$	5000
2	5000	$65\tau_{bit}$	5000
3	5000	$75\tau_{bit}$	5000
4	5000	$65\tau_{bit}$	5000
5	5000	$75\tau_{bit}$	5000
6	10000	$115\tau_{bit}$	10000
7	10000	$65\tau_{bit}$	10000
8	10000	$75\tau_{bit}$	10000
9	10000	$75\tau_{bit}$	10000
10	100000	$65\tau_{bit}$	100000
11	100000	$95\tau_{bit}$	100000
12	100000	$65\tau_{bit}$	100000
13	100000	$65\tau_{bit}$	100000
14	1000000	$85\tau_{bit}$	1000000
15	1000000	$65\tau_{bit}$	1000000
16	1000000	$65\tau_{bit}$	1000000

Na configuração *offset* 1, todos os *offsets* das mensagens são iguais a zero, e na configuração *offset* 2 (mesma regra utilizada no primeiro estudo de caso), as mensagens possuem os seguintes *offsets* em μs: $O_0 = 0$, $O_1 = 520$, $O_2 = 1120$, $O_3 = 1640$, $O_4 = 2240$, $O_5 = 2760$, $O_6 = 3360$, $O_7 = 4280$, $O_8 = 4800$, $O_9 = 5400$, $O_{10} = 6000$, $O_{11} = 6520$, $O_{12} = 7280$, $O_{13} = 7800$, $O_{14} = 8320$, $O_{15} = 9000$, $O_{16} = 9520$.

O tempo de verificação para cada configuração foi de aproximadamente 30 s. As curvas dos tempos de resposta no melhor e pior caso versus o identificador da mensagem estão representadas na Figura 4. Assim como no estudo de caso anterior, os tempos de resposta no pior caso (curvas vermelha e azul com circulos) foram menores que os tempos de resposta com *offset* dinâmico (curva preta). Mais uma vez, os resultados com a configuração *offset* 2 foram melhores. Observe que as curvas dos tempos de resposta no melhor e pior caso para ambas as configurações de *offsets* são parecidas. Acreditamos que esse comportamento foi provocado pelo alto fator de utilização da rede. Apesar do alto fator de utilização, conseguimos obter uma redução significante dos tempos de resposta no pior caso utilizando *offsets* estáticos.

8 Conclusões

Neste trabalho, apresentou-se como alternativa à técnica de simulação, a utilização da técnica de verificação de modelos para determinar os tempos de resposta exatos no melhor e pior caso de mensagens CAN com *offsets* estáticos. Foi verificado que os tempos de resposta no pior caso são reduzidos significantemente quando as mensagens são

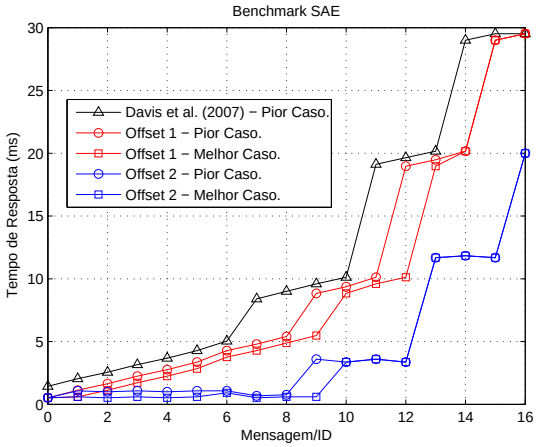


Figura 4: Curvas dos tempos de resposta no melhor e pior caso das mensagens do *benchmark* SAE.

escalonadas com *offsets* estáticos. Demonstrou-se, para o modelo desenvolvido, a necessidade de utilizar a capacidade da ferramenta UPPAAL em atribuir prioridades aos autômatos, afim de evitar possíveis inversões de prioridade no acesso ao meio pelas mensagens. Observamos que o algoritmo desenvolvido para atribuição de prioridades, reduziu o espaço de estados do modelo, de modo a eliminar estados que na prática não acontecem.

Pretende-se em trabalhos futuros desenvolver um algoritmo para atribuição de *offsets* que minimize o máximo possível os tempos de resposta das mensagens, e incrementar o modelo para considerar a transmissão de mais de uma mensagem por nó, erros de transmissão nas mensagens e as variações nos relógios dos nós.

Agradecimentos

Os autores agradecem o apoio financeiro fornecido pela CAPES e CNPq.

Referências

Alur, R. and Dill, D. L. (1994). A theory of timed automata, *Theor Comput Sci* **126**: 183–235.

Audsley, N. C. (1991). Optimal priority assignment and feasibility of static priority tasks with arbitrary start times, *YCS164, Dept. of Computer Science, University of York* .

Berhmann, G., David, A. and Larsen, K. G. (2004). A tutorial on uppaal, formal methods for the design of real-time systems, *In SFM-RT, Springer-Verlag* **965**: 200–236.

Bosch, R. (1991). Can specification version 2.0. Robert Bosch GmbH, Postfach 30 02 40, D-70442 Stuttgart.

Braun, C., Havet, L. and Navet, N. (2007). Netcarbench: a benchmark for techniques and

tools used in the design of automotive communication systems, *In Proc. of the 7th IFAC International Conference on Fieldbuses and Networks in Industrial and Embedded Systems* .

Bulychev, P. E., David, A., Larsen, K. G., Mikucionis, M., Poulsen, D. B., Legay, A. and Wang, Z. (2012). Statistical model checking for priced timed automata, *In proceedings of the 10th workshop on quantitative aspects of programming languages* .

Chen, Y., Kurachi, R., Zeng, G. and Takada, H. (2011). Schedulability comparison for can message with offset: Priority queue versus fifo queue, *In Proc. of the 19th International Conference on Real-Time and Network Systems* .

David, A., Larsen, K. G., Legay, A., Mikucionis, M. and Wang, Z. (2011). Time for statistical model checking of real-time systems, *In proceedings of the 23rd international conference on computer aided verification* .

Davis, R. I., Burns, A., Bril, R. J. and Lukkien, J. J. (2007). Controller area network schedulability analysis: Refuted, revisited and revised, *Real-Time Systems* **35**(3): 239–272.

Du, L. and Xu, G. (2009). Worst case response time analysis for can messages with offsets, *In IEEE International Conference on Vehicular Electronics and Safety* pp. 41–45.

Goossens, J. (2003). Scheduling of offset free systems, *Real-Time Systems* **24**(2): 239–258.

Grenier, M., Havet, L. and Navet, N. (2008). Pushing the limits of can-scheduling frames with offsets provides a major performance boost, *In Proc. of The 4th European Congress Embedded Real Time Software* .

Henzinger, T. A. (1994). Symbolic model checking for real-time systems, *Information and Computation* **111**: 193–244.

Larsen, K. G., Pettersson, P. and Yi, W. (1995). Model-checking for real-time systems, *In Proceedings of the 10th international conference on fundamentals of computation theory* **965**: 62–88.

Navet, N., Song, Y.-Q. and Simonot, F. (2000). Worst-case deadline failure probability in real-time applications distributed over controller area network, *Journal of Systems Architecture* **46**: 607–617.

Palencia, J. and Harbour, M. G. (1998). Schedulability analysis for tasks with static and dynamic offsets, *In Proc. The 19th IEEE Real-Time Systems Symposium* .

RTaW (2009). Netcar-analyzer: Timing analysis and resource usage optimization for controller area network. Available at http://www.realtimeatwork.com/software.

RTaW (2012). Rtaw-sim: Controller area network simulation. Available at https://www.realtimeatwork.com/software.

Sá, J. S., Lima, A. M. N., Perkusic, A. and Neto, J. S. R. (2010). Análise dos tempos de resposta fim a fim de sistemas de controle via rede baseado na técnica de verificação de modelos, *XVIII Congresso Brasileiro de Automática - CBA 2010* .

Sá, J. S., Lima, A. M. N., Perkusic, A. and Neto, J. S. R. (2012). Análise probabilística dos tempos de resposta de uma rede can utilizando a técnica de verificação de modelos estatísticos, *XIX Congresso Brasileiro de Automática - CBA 2012* .

Szakaly, A. (2003). *Response time analysis with offsets for can*, Master's thesis, Department of Computer Engineering, Chalmers University of Technology, Sweden.

Tindell, K. W., Burns, A. and Wellings, A. J. (1995). Calculating controller area network (can) message response times, *Controle Engineering Practice* **3**(8): 1163–1169.

Waszniowsky, L., Hanzálek, Z. and Hanzálek, Z. (2009). Case study on distributed and fault tolerant system modeling based on timed automata, *The journal of systems and software* (82): 1678–1694.

Yomsi, P. M., Bertrand, D., Navet, N. and Davis, R. I. (2012). Controller area network (can): Response time analysis with offsets, *In Proc of The 9th IEEE International Workshop on Factory Communication System* .