Rodrigo Lange

MÉTODOS DE ESCALONAMENTO DE MENSAGENS PARA O SISTEMA DE COMUNICAÇÃO FLEXRAY

Tese submetida ao Programa de Pós Graduação em Engenharia de Automação e Sistemas para a obtenção do Grau de Doutor.
Orientador: Rômulo Silva de Oliveira: Prof. Dr.
Coorientador: Francisco Manuel Madureira e Castro Vasques de Carvalho: Prof. Dr.

Florianópolis

2015

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Lange, Rodrigo
Métodos de Escalonamento de Mensagens Para o Sistema de Comunicação FlexRay / Rodrigo Lange ; orientador, Rômulo Silva de Oliveira ; coorientador, Francisco Manuel Madureira e Castro Vasques de Carvalho. - Florianópolis, SC, 2015.
185 p.

Tese (doutorado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Engenharia de Automação e Sistemas.

Inclui referências

1. Engenharia de Automação e Sistemas. 2. FlexRay. 3. Sistemas Automotivos. 4. Sistemas de Tempo Real. I. Silva de Oliveira, Rômulo. II. Carvalho, Francisco Manuel Madureira e Castro Vasques de. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia de Automação e Sistemas. IV. Título.

Rodrigo Lange

MÉTODOS DE ESCALONAMENTO DE MENSAGENS PARA O SISTEMA DE COMUNICAÇÃO FLEXRAY

Esta Tese foi julgada aprovada para a obtenção do Título de "Doutor", e aprovada em sua forma final pelo Programa de Pós Graduação em Engenharia de Automação e Sistemas.

Florianópolis, 01 de Setembro 2015.

Prof. Dr.
Rômulo Silva de Oliveira
Universidade Federal de Santa Catarina

Banca Examinadora:

Prof. Dr.
Orientador: Rômulo Silva de Oliveira

Prof. Dr.
Coorientador: Francisco Manuel Madureira e Castro Vasques de Carvalho

Prof. Dr. Carlos Eduardo Pereira
Departamento de Engenharia Elétrica, Universidade Federal do Rio Grande do Sul

Prof. Dr. Rafael Rodrigues Obelheiro
Departamento de Ciências da Computação, Universidade do Estado de Santa Catarina

Prof. Dr. Cristian Koliver
Instituto de Informática e Estatística, Universidade Federal de Santa Catarina

Prof. Dr. Carlos Barros Montez
Departamento de Automação e Sistemas, Universidade Federal de Santa Catarina

Prof. Dr. Marcelo Ricardo Stemmer
Departamento de Automação e Sistemas, Universidade Federal de Santa Catarina

Este trabalho é dedicado à minha família, aos meus amigos e a todos os professores que tive ao longo dos anos, em especial aos meus orientadores.

RESUMO

Este trabalho se insere na área de protocolos de tempo real, abordando especificamente o Sistema de Comunicação FlexRay, um protocolo de tempo real para usos automotivos. O objeto de estudo deste trabalho foram os mecanismos de escalonamento de fluxos de mensagens para o FlexRay, bem como as técnicas utilizadas na análise de tempo de resposta em sistemas que utilizam tal protocolo. O objetivo geral desta tese foi a elaboração e a avaliação de mecanismos para o escalonamento e análise de tempo de resposta de sistemas que utilizem o Sistema de Comunicação FlexRay. São apresentadas quatro propostas. As duas primeiras propostas estão relacionadas ao segmento Estático do FlexRay. Ambas demonstram a viabilidade de se definir a alocação de *slots* estáticos para cada nodo utilizando técnicas tradicionais para a análise de tempo de resposta considerando-se os requisitos temporais impostos pelo conjunto de fluxos de mensagens de cada nodo, e são métodos capazes de considerar conjuntos de fluxos com períodos que não são múltiplos de FC, sendo também capazes de considerar o caso em que a geração de mensagens nos fluxos não está sincronizada com o FC. São também apresentadas duas propostas que abordam a questão do escalonamento de fluxos de mensagens aperiódicos no Segmento Dinâmico do FlexRay. Foram apresentados dois mecanismos para métodos de arbitragem do DN que tiram vantagem da flexibilidade que fluxos aperiódicos possuem em relação a restrições de tempo real. Em ambos os mecanismos, os fluxos de mensagens aperiódicos de um sistema são associados com uma probabilidade de *backoff*, e um *middleware* de tempo real específico utiliza tal probabilidade de *backoff* para definir se uma mensagem gerada por um fluxo aperiódico irá competir ou não pelo barramento no ciclo de comunicação atual, influenciando nas chances que mensagens com prioridades mais baixas tem de serem transmitidas.

Palavras-chave: FlexRay. Sistemas Automotivos. Sistemas de Tempo Real

ABSTRACT

This work addresses the FlexRay Communication System, a digital serial bus for automotive applications designed to meet the demands of X-by-Wire systems. It provides flexibility, bandwidth and determinism by combining static and dynamic approaches for message transmission, incorporating the advantages of synchronous and asynchronous protocols. The area of interest of this work is scheduling mechanisms for FlexRay, being the overall objective of this thesis the development and evaluation of new techniques for scheduling and timing analysis for FlexRay. In this document four proposals are presented. Two proposals are related to FlexRay Static Segment. These two proposals demonstrate the feasibility of defining the static slot allocation for each node using traditional Response Time Analysis (RTA) techniques, and thus considering the timing requirements imposed by the set of message streams allocated to each node. The proposed techniques are able to deal with message stream sets where periods are not multiples of the FlexRay cycle duration, nor the messages generation is synchronized with the FlexRay cycle. They are also presented two proposals addressing the scheduling of aperiodic message streams in FlexRay Dynamic Segment. Both mechanisms use a probabilistic approach that takes advantage of the flexibility of aperiodic message streams regarding real-time constraints. In the proposed methods, a real-time middleware in each network node manages the transmission of messages generated by aperiodic streams in Dynamic Segment. Whenever a RT-middleware senses that aperiodic messages may be indefinitely postponed, it enters backoff mode. In backoff mode, a RT-middleware randomly defines whether an aperiodic message that is waiting to be transmitted will be sent to the bus in the current FC or if that message will be postponed to another FC, affecting the transmission chances of messages generated by streams with lower priorities have of being transmitted.

Keywords: FlexRay. Sistemas Automotivos. Sistemas de Tempo Real.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

LISTA DE SÍMBOLOS

$|\mathbb{S}_h^{CAN}|$ Número de fluxos de mensagens do nodo N_h que transmitem em um barramento CAN. 66

$|\mathbb{S}_h^{DN}|$ Número de fluxos de mensagens do nodo N_h que transmitem no Segmento Dinâmico de um barramento FlexRay. 65

$|\mathbb{S}_h^{ST}|$ Número de fluxos de mensagens do nodo N_h que transmitem no Segmento Estático de um barramento FlexRay. 65

$|\mathbb{T}_c^{\varphi}|$ Número de tarefas na atividade φ_c. . 68

$|\mathbb{T}_h|$ Número de tarefas do nodo N_h. 62

$B_{h,j}^{CAN}$ Máximo bloqueio sofrido por um fluxo de mensagens $S_{h,j}^{CAN}$. 60

$B_{h,j}^{ST}$ Atraso inicial em segundos na transmissão de um quadro gerado pelo fluxo de mensagens $S_{h,j}^{ST}$. 103

c Número da atividade φ_c. 68

$C_{h,j}^{ap}$ Tamanho em segundos de um quadro gerado pelo fluxo de mensagens $S_{h,j}^{ap}$, incluindo o cabeçalho e a cauda necessários. 126

$C_{h,j}^{CAN}$ Tamanho em segundos de um quadro gerado pelo fluxo de mensagens $S_{h,j}^{CAN}$, incluindo o cabeçalho e cauda necessários. 59

$C_{h,j}^{DN}$ Tamanho em segundos de um quadro gerado pelo fluxo de mensagens $S_{h,j}^{DN}$, incluindo o cabeçalho e cauda necessários. 38

$C_{h,j}^{ST}$ Tamanho em segundos de um quadro gerado pelo fluxo de mensagens $S_{h,j}^{ST}$, incluindo o cabeçalho e cauda necessários. 38

C_{max}^{ST} Tempo em segundos necessário para transmitir a maior mensagem gerada por um fluxo em $\mathbb{S}^{ST}$. 78

$C_{h,i}^{\Gamma}$ Tempo de execução em segundos da tarefa $\Gamma_{h,i} nopiorcaso$. 63

SUMÁRIO

1 INTRODUÇÃO

A indústria automotiva é, atualmente, um dos maiores setores econômicos mundiais, produzindo anualmente dezenas de milhões de veículos e contribuindo significativamente para a receita de vários países. E nos últimos anos esta indústria tem observado um crescimento exponencial no número das funções computadorizadas embarcadas em veículos. Toda uma classe de funções eletrônicas para sistemas de navegação, controle de tração, controle de estabilidade, sistemas de segurança ativa, multimídia e muitos outros têm sido desenvolvidos, sendo que veículos modernos possuem mais de 70 controladores eletrônicos trocando 2.500 sinais que representam informações elementares como velocidade ou temperatura do motor (NAVET; SIMONOT-LION, 2008).

O crescimento na complexidade das arquiteturas eletrônicas, aliado a restrições relacionadas a sensores e atuadores, levou à adoção de uma abordagem distribuída para a implementação das várias funções existentes. Neste contexto, as redes e protocolos de comunicação têm importância fundamental pois servem como suporte para a integração de funções, reduzindo o custo e a complexidade do cabeamento e fornecendo, também, um meio para a implementação de métodos de tolerância a falhas.

Tipicamente, o sistema embarcado em um veículo é dividido em domínios funcionais que correspondem a diferentes características e restrições. Em dois destes domínios, *power train* (relacionado ao controle do motor e transmissão) e *chassis* (relacionado ao controle da suspensão, direção e freios), aspectos relacionados à segurança são fundamentais e as soluções técnicas devem garantir a confiabilidade do sistema, mantendo-se, ao mesmo tempo, rentáveis (NAVET; SIMONOT-LION, 2008). Para esses domínios, o protocolo de comunicação *Controller Area Network* (CAN) tem sido o padrão de fato, com milhões de dispositivos com *interface* para este protocolo fabricados a cada ano (DAVIS et al., 2007).

Por motivos econômicos e tecnológicos, atualmente existe na indústria automotiva a tendência de substituir sistemas mecânicos ou hidráulicos por outros eletrônicos. Do ponto de vista econômico, o custo de componentes de *hardware* vem caindo, enquanto a confiabilidade dos mesmos aumenta, o que justifica tais substituições. Já do ponto de vista tecnológico, o que motiva a substituição é o fato de que sistemas eletrônicos permitem a adoção de novas funções cujo desenvolvimento seria impossível se utilizados apenas sistemas mecânicos ou hidráulicos.

Essa tendência resultou no conceito *X-by-Wire*, onde sistemas eletrônicos ou hidráulicos presentes em veículos são substituídos por outros puramente eletrônicos. Por exemplo, sistemas de aceleração puramente eletrônicos já são comuns em veículos atuais, e sistemas eletrônicos para o câmbio (também chamados *Shift-by-Wire* ou *Gear-by-Wire*) já estão presentes em alguns veículos de luxo como os das séries 5 e 7 da BMW (ZURAWSKI, 2005).

Segundo vários autores, o CAN não é apropriado para futuras aplicações *X-by-Wire* por possuir limitações em relação a aspectos como desempenho e segurança, e por isso foram desenvolvidos novos protocolos para aplicações automotivas, como por exemplo o *Byteflight* da BMW (ZURAWSKI, 2005; SCHEDL, 2007). Desses novos protocolos, o **Sistema de Comunicação FlexRay** se tornou um padrão para uso em aplicações com altos requisitos de desempenho, determinismo e confiabilidade (NAVET; SIMONOT-LION, 2008; OUEDRAOGO; KUMAR, 2014; ZURAWSKI, 2005).

1.1 MOTIVAÇÃO

O Sistema de Comunicação FlexRay é um barramento serial digital projetado para atender às demandas de aplicações automotivas com altos requisitos de largura de banda, confiabilidade, determinismo e sincronização, tais como as aplicações relacionadas a sistemas *X-by-Wire*.

Como será descrito no Capítulo 2, o protocolo FlexRay foi desenvolvido para acomodar a transmissão de mensagens geradas por fluxos com diferentes características temporais. O método de controle de acesso ao meio do FlexRay é baseado em um ciclo de comunicação de tamanho fixo. Este ciclo é dividido em quatro segmentos que também possuem tamanho fixo. Um destes segmentos é chamado de segmento Estático e foi projetado para acomodar mensagens geradas por sistemas *time-triggered*[1]. Outro segmento é chamado segmento Dinâmico e foi projetado para acomodar a transmissão de mensagens geradas por sistemas *event-triggered*. Por fim, existem dois segmentos, chamados *Network Idle Time* e *Symbol Window*, que são utilizados para a troca de mensagens de controle e para a sincronização do sistema.

Um dos pontos motivadores desta tese é que, apesar da flexibi-

[1] Os termos em inglês serão traduzidos para português quando isso não acarretar prejuízo ao entendimento do trabalho. Na literatura de tempo real, os termos *time-triggered* e *event-triggered* são bastante comuns na sua forma original.

lidade do FlexRay em relação às características temporais de fluxos de mensagens, a literatura em geral considera modelos de sistemas restritivos, como mostra o levantamento do estado da arte que será apresentado no Capítulo 3. Por exemplo, as propostas que abordam o segmento Estático em geral consideram modelos de sistema onde não só existe uma sincronização entre a geração de mensagens e o ciclo de comunicação, mas também os períodos dos fluxos de mensagens são múltiplos inteiros do tamanho do ciclo de comunicação. No caso dos trabalhos relacionados ao segmento Dinâmico, grande parte das propostas consideram apenas modelos de sistema onde os fluxos de mensagens possuem características críticas de tempo real. Mas a definição do escalonamento de um sistema considerando restrições temporais artificiais, como, por exemplo, a redução dos períodos dos fluxos de mensagens, pode resultar em uma subutilização da rede, pois é necessário reservar recursos para os fluxos de mensagens, mesmo que eles não utilizem tais recursos na sua totalidade.

Outro ponto motivador desta tese está relacionado ao fato de que, no futuro próximo, é esperada a coexistência do FlexRay e de outros protocolos em um mesmo veículo. Apesar do trabalho em (STEINBACH et al., 2010) sugerir que o *Time-Triggered Ethernet* (TTEthernet) (TTTech Computertechnik AG, 2015) poderá vir a ser um substituto para todas as atuais redes veiculares, e apesar do protocolo CAN ainda atender às necessidades de diversas classes de aplicações com um custo relativamente baixo, segundo Lo Bello (2011), nos próximos anos FlexRay provavelmente seguirá sendo utilizado nos sistemas de *powertrain* e no gerenciamento da dinâmica de um veículo. Neste cenário, o FlexRay poderá ser utilizado em diversas situações: em aplicações que requeiram alta velocidade ou como *backbone* entre segmentos de rede que podem ou não utilizar diferentes protocolos (NAVET; SIMONOT-LION, 2008) (Figuras 1 e 2).

1.2 OBJETIVOS

O objetivo geral desta tese é a elaboração e a avaliação de mecanismos para o escalonamento e a análise de tempo de resposta de sistemas que utilizem o Sistema de Comunicação FlexRay. Para atender o objetivo geral desta tese, os seguintes objetivos específicos foram definidos:

- Desenvolver um mecanismo para escalonar, no segmento Estático do FlexRay, fluxos de mensagens cujos períodos não são múltiplos

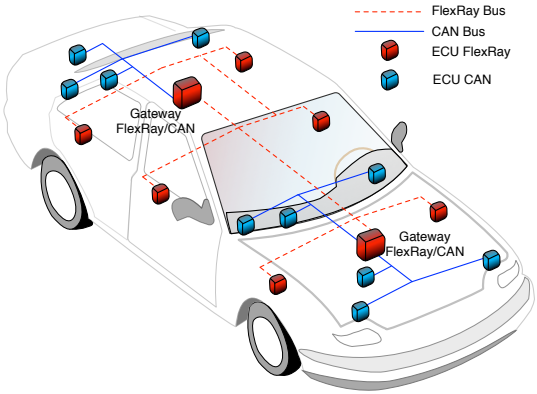

Figura 1 – Exemplo de rede intra-veicular com *gateways*. Baseado em Navet e Simonot-Lion (2008)

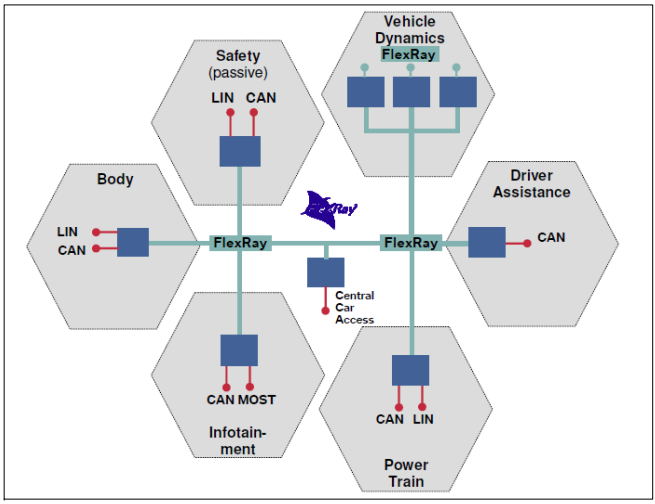

Figura 2 – FlexRay como Backbone (SCHEDL, 2007)

inteiros do ciclo de comunicação e cuja geração de mensagens não está sincronizada com o ciclo de comunicação;

- Desenvolver um mecanismo para escalonar, no segmento Dinâmico do FlexRay, fluxos de mensagens que não possuem características temporais rígidas, como por exemplo fluxos de mensagens aperiódicos.

1.3 ORGANIZAÇÃO DO TEXTO

Este documento está estruturado em sete capítulos, sendo que este primeiro apresentou os aspectos que motivaram esta pesquisa e os objetivos desta Tese.

No Capítulo 2 serão descritos os conceitos relacionados ao Sistema de Comunicação FlexRay e, também, conceitos relevantes relacionados ao padrão industrial AUTOSAR. De forma complementar ao Capítulo 2, no Capítulo 3 serão revisitados os trabalhos que representam o estado da arte relacionado ao FlexRay. O levantamento bibliográfico apresentado nos Capítulos 2 e 3 foi parcialmente publicado em (LANGE; OLIVEIRA, 2013).

No Capítulo 4 são apresentados o modelo de sistema, as definições e os conceitos que serão utilizados nos demais capítulos desta Tese.

No Capítulo 5 é apresentada uma nova abordagem para o escalonamento do segmento Estático do FlexRay. A proposta inclui dois métodos que utilizam as técnicas tradicionais para a análise de tempo de resposta na definição da alocação de *slots* estáticos para nodos FlexRay. Tais métodos são chamados respectivamente de Método de Alocação Proporcional e Método de Alocação Proporcional Adaptativo, e ambos são capazes de considerar conjuntos de fluxos com períodos que não são múltiplos de FC, sendo também capazes de considerar o caso em que a geração de mensagens nos fluxos não está sincronizada com o FC. Os métodos propostos no Capítulo 5 foram parcialmente publicados em (LANGE et al., 2012) e (LANGE et al., 2015).

O Capítulo 6 aborda a questão do escalonamento, no Segmento Dinâmico do FlexRay, de fluxos de mensagens aperiódicos que possuem características temporais não críticas ou que não possuem restrições temporais. São apresentadas propostas para dois métodos probabilísticos que tiram vantagem da flexibilidade que fluxos aperiódicos possuem em relação a restrições de tempo real. Nos mecanismos propostos, é utilizada uma *probabilidade de backoff* para definir, durante a execução

do sistema, se uma mensagem gerada por um fluxo de mensagens que possui características de tempo real não críticas será enviada para o barramento no ciclo de comunicação atual, ou se a mesma será postergada para outro ciclo de comunicação, contribuindo, assim, para aumentar a chance de transmissão de mensagens com prioridades mais baixas. O trabalho apresentado no Capítulo 6 foi parcialmente publicado em (LANGE; OLIVEIRA, 2014).

Esta tese é finalizada com o Capítulo 7, que apresenta a revisão dos objetivos, o resumo do trabalho realizado, as contribuições ao estado da arte e as perspectivas para futuras pesquisas.

2 FUNDAMENTOS TEÓRICOS

Antes de formalizar as propostas desta Tese, é necessário apresentar os conceitos teóricos que fundamentam tais propostas, o que é feito neste capítulo.

Serão apresentados, na Seção 2.1, os conceitos básicos do Sistema de Comunicação FlexRay. Na sequência, são resumidos conceitos básicos do padrão AUTOSAR, um padrão de desenvolvimento utilizado pela indústria automotiva (Seção 2.2). Por fim, na Seção 2.3 são feitas considerações sobre o material apresentado.

Cabe ressaltar que, quando um conceito específico for necessário para apenas uma das propostas apresentadas nos demais capítulos que compõem esta tese, tal conceito será discutido no capítulo relacionado a tal proposta.

2.1 SISTEMA DE COMUNICAÇÃO FLEXRAY

O Sistema de Comunicação FlexRay (*FlexRay Communication System*) é um barramento serial digital para aplicações automotivas desenvolvido para atender às demandas de desempenho e disponibilidade que são esperadas nas próximas gerações de sistemas *X-by-wire* (FLEXRAY, 2015). Entre outras características, ele fornece flexibilidade, determinismo e largura de banda através da combinação de abordagens estáticas e dinâmicas para a transmissão de mensagens, incorporando as vantagens tanto de protocolos com métodos de arbitragem síncronos como de protocolos com métodos de arbitragem assíncronos (NAVET; SIMONOT-LION, 2008; ZURAWSKI, 2005).

O FlexRay foi desenvolvido entre os anos 2000 e 2010 por uma aliança de fabricantes que incluiu, entre outros, BMW, DaimlerChrysler e Bosch. Esse consórcio concluiu seus trabalhos com a finalização e a divulgação da Versão 3.0 da Especificação do Sistema de Comunicação FlexRay. Essa versão eventualmente se tornou um conjunto de normas ISO sob os números ISO 17458-1 a 17458-5. Estas normas incluem definições para o Protocolo de Comunicação (*Protocol Specification*), Camada Física (*Electrical Physical Layer Specification*), Camada de Enlace de Dados (*FlexRay Data Link Layer Conformance Test Specification*) e *Bus Guardian* (*Node-Local and Central Bus Guardian Specification*) (FLEXRAY, 2015).

Serão apresentadas, a seguir, as principais características do Flex-

Ray. Também serão discutidos os trabalhos da literatura sobre análise de escalonamento e tempo de resposta no FlexRay considerados relevantes para esta tese.

Visando manter a clareza e coerência do texto em relação à especificação do FlexRay, muitos termos foram mantidos na sua forma original em inglês, como, por exemplo, no caso da expressão *"cluster"*.

2.1.1 FlexRay: principais características

2.1.1.1 Topologias de rede

O FlexRay é um protocolo com dois canais de comunicação (denominados canal A e canal B) sendo que cada canal suporta taxas de transmissão de até 10 Mbit/s. Um sistema composto de múltiplos nodos interconectados através de um barramento FlexRay com até dois canais é chamado *cluster*. Em um *cluster*, cada nodo da rede pode estar conectado a um ou a ambos os canais. Em uma situação normal (livre de falhas), todos os nodos conectados ao Canal A são capazes de se comunicar com todos os outros nodos no Canal A e, da mesma forma, todos os nodos ligados ao Canal B são capazes de se comunicar com todos os outros nodos ligados ao Canal B.

Um *cluster* FlexRay pode ser configurado em diferentes topologias que permitem tanto aumentar a largura de banda como introduzir um canal redundante para aumentar o nível de tolerância a falhas. Tais topologias podem ser um barramento com um ou dois canais, uma em estrela com um ou dois canais, ou diversas combinações híbridas de topologias tipo barramento ou estrela (FLEXRAY, 2015). Na Figura 3 são ilustrados exemplos de possíveis combinações para a topologia do *cluster*.

Na Figura 3.a é apresentado um exemplo de topologia com dois barramentos, sendo que cada canal de comunicação forma um barramento diferente. Na Figura 3.b é apresentado um exemplo de topologia tipo estrela. E na Figura 3.c é apresentado um exemplo de topologia híbrida, onde o canal A forma um barramento, e o canal B forma uma estrela.

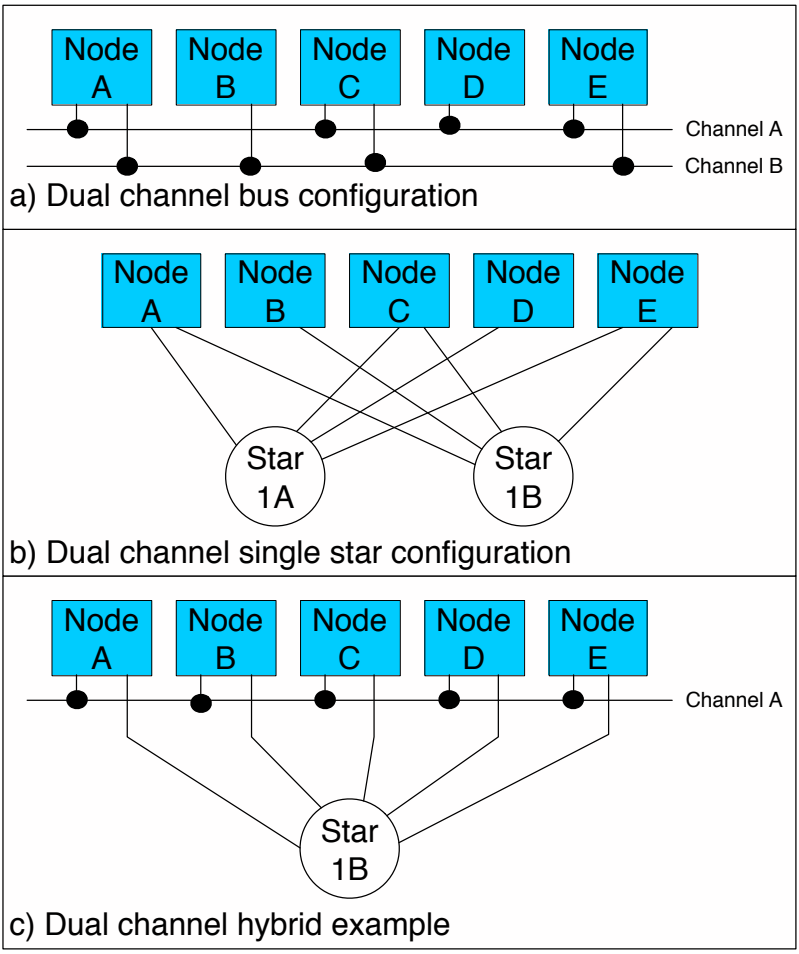

Figura 3 – Exemplos de topologias de rede (FLEXRAY, 2015)

2.1.1.2 Controle de acesso ao meio

O controle de acesso ao meio (em inglês: *Media Access Control*, MAC) do FlexRay é baseado em um ciclo de comunicação (ciclo FlexRay ou *FlexRay Cycle*, FC) com tamanho pré-determinado que é repetido periodicamente a partir do início do sistema. A especificação do FlexRay define que o tamanho do FC é dado em μs e representado pelo parâmetro de sistema *gdCycle*. O tamanho do FC pode ser representado de forma diferente em alguns trabalhos. Por exemplo, (POP et al., 2008) representa o mesmo como FC_{bus}. Por questões de compatibilidade com a norma, quando o tamanho do ciclo FlexRay for dado em μs será utilizado *gdCycle*, caso contrário será utilizado FC_{bus}. O número do FC atual é incremental, e seu valor atual é armazenado na variável *vCycleCounter*. *vCycleCounter* tem valor máximo 63, sendo reiniciado para 0 toda vez que este valor é ultrapassado.

Cada ciclo FlexRay é estruturado como uma sequência de quatro segmentos: um segmento Estático (*Static Segment*, ST), um segmento Dinâmico (*Dynamic Segment*, DN) e dois segmentos de controle, *Symbol Window* (SW) e *Network Idle Time* (NIT). Na Figura 4 é ilustrado um Ciclo FlexRay típico.

Uma rede FlexRay é composta por um mínimo de dois nodos, sendo que o ciclo de comunicação deve conter ao menos o segmento Estático e o NIT, e cada nodo deve estar associado a ao menos um *slot* estático.

O ST de um ciclo de comunicação utiliza um método de controle de acesso ao meio baseado no *Time Division Multiple Access* (TDMA). O ST é composto por um número $gNumberOfStaticSlots$ de *slots*, sendo que cada *slot* é formado por uma quantia idêntica de *macroticks* definidos em *gdStaticSlot*. Alternativamente, neste trabalho o tamanho do ST poderá ser representado como ST_{bus}. O *macrotick* (MT) é a unidade básica de tempo em um *cluster* FlexRay, e seu conceito será aprofundado na Seção 2.1.1.5. Em um *cluster* tanto $gNumberOfStaticSlots$ como *gdStaticSlot* são valores constantes globais definidos na fase de projeto.

Cada *slot* do ST possui um identificador de *slot* $FrameID$ e pertence exclusivamente a um controlador para a transmissão de um quadro (*frame*), mas esta posse está relacionada apenas a um canal de comunicação. Como o FlexRay é um protocolo que suporta até dois canais de comunicação, podem surgir as seguintes combinações para um único nodo no segmento Estático: no *slot* 1 o nodo transmite um quadro no canal A e um no canal B. No *slot* 2 transmite apenas um

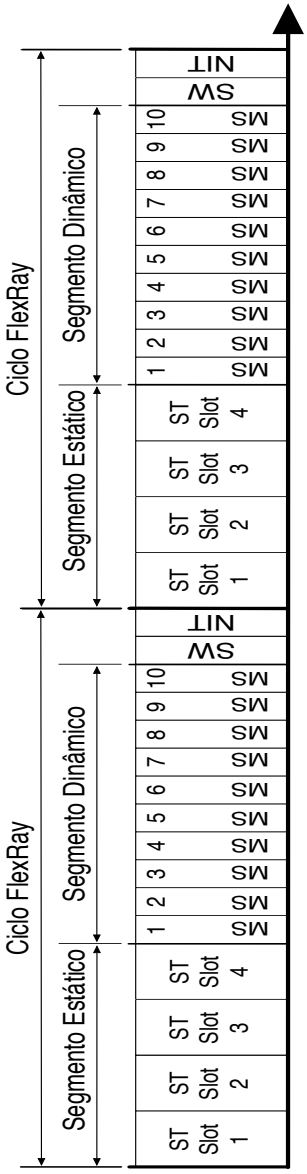

Figura 4 – Ciclo FlexRay (FLEXRAY, 2015)

quadro no canal A (ou no canal B). No *slot* 3 não há transmissão de quadros. Na Figura 5 são ilustradas estas possíveis combinações.

Na Figura 6 é apresentado outro exemplo de comunicação utilizando os dois canais do FlexRay. Neste exemplo, um sistema é composto por cinco nodos de rede e por um conjunto $\mathbb{S}^{ST}$ contendo $|\mathbb{S}^{ST}|$ fluxos de mensagens que transmitem no ST. Cada fluxo $S^{ST}_{h,j} \in \mathbb{S}^{ST}$, onde h é o número do nodo ao qual o fluxo está associado e j representa o número do fluxo dentro do nodo, está associado com um $FrameID$ e um ou dois canais de comunicação. Neste exemplo, considera-se que a associação entre os fluxos de mensagens e os $FrameIDs$ é arbitrária. Os nodos de rede e os fluxos de comunicação serão formalizados no Capítulo 4.

Neste sistema os nodos N_1 e N_3 estão conectados a ambos os canais de comunicação. Os nodos N_2 e N_5 estão conectados apenas ao canal B, e o nodo N_4 está conectado apenas ao canal A. O nodo N_1 possui os fluxos de mensagens $S^{ST}_{1,1}$ e $S^{ST}_{1,2}$ (os fluxos de mensagens serão formalizados no Capítulo 4), o nodo N_2 possui o fluxo $S^{ST}_{2,1}$, o nodo N_3 possui o fluxo $S^{ST}_{3,1}$, o nodo N_4 possui o fluxo $S^{ST}_{4,1}$ e o nodo N_5 possui o fluxo $S^{ST}_{5,1}$. O fluxo de mensagens $S^{ST}_{1,1}$ utiliza o *slot* estático 1 de ambos os canais para a transmissão redundante de mensagens. De forma semelhante, o fluxo $S^{ST}_{3,1}$ utiliza o *slot* 3 de ambos os canais. O fluxo $S^{ST}_{2,1}$ utiliza o apenas o *slot* 4 do canal B e o fluxo $S^{ST}_{5,1}$ utiliza apenas o *slot* 9 do canal B. O *slot* 7 possui uma situação diferente: no canal A, o *slot* 7 está associado ao fluxo $S^{ST}_{4,1}$, e no canal B, ao fluxo $S^{ST}_{1,2}$. No exemplo, os demais *slots* estáticos não são utilizados. Esse exemplo foi baseado em (NAVET; SIMONOT-LION, 2008).

Para manter a ordem do escalonamento das mensagens, cada nodo mantém contadores de *slots* $vSlotCounter$ (um para cada canal). Estes contadores são iniciados com 1 no início de cada ciclo e incrementados simultaneamente no final de um *slot*.

A natureza cíclica do ciclo FlexRay aliada ao tamanho fixo dos *slots* estáticos permite que o segmento seja visto como uma matriz, onde as colunas são os *slots* estáticos, e as linhas são ciclos, existindo no máximo 64 linhas devido à restrição imposta por $vCycleCounter$. Cada coluna é associada exclusivamente a um nodo, mas os fluxos estão associados a um *ciclo base* e uma *repetição*. O ciclo base representa o primeiro dentre os 64 ciclos no qual as mensagens de um fluxo de mensagens serão transmitidas, e a repetição é o intervalo (em números inteiros de FC) entre duas associações entre um fluxo de um nodo e um *slot* estático associado a este nodo. Um exemplo da visão do ST como uma matriz é apresentado na Figura 7.

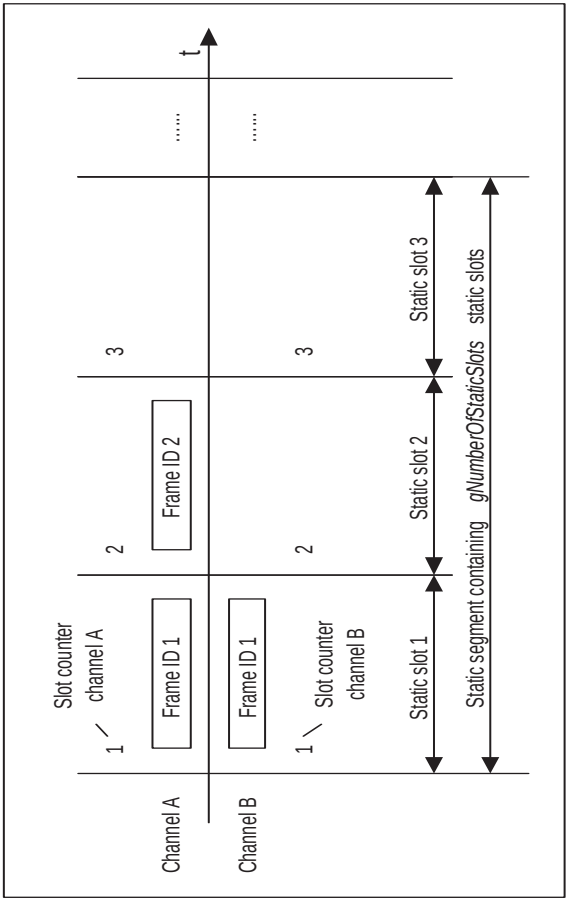

Figura 5 – Estrutura do segmento Estático (FLEXRAY, 2015)

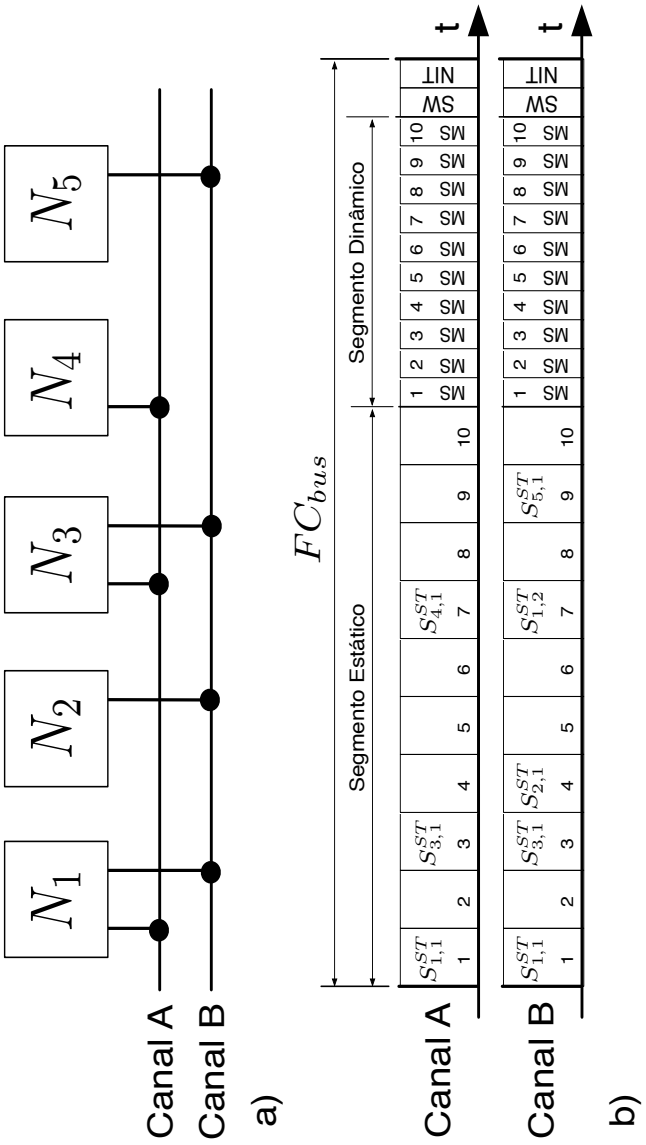

Figura 6 – Comunicação no segmento Estático. (a) *Cluster* FlexRay. (b) Transmissão de quadros em *slots* estáticos. Baseado em (NAVET; SIMONOT-LION, 2008)

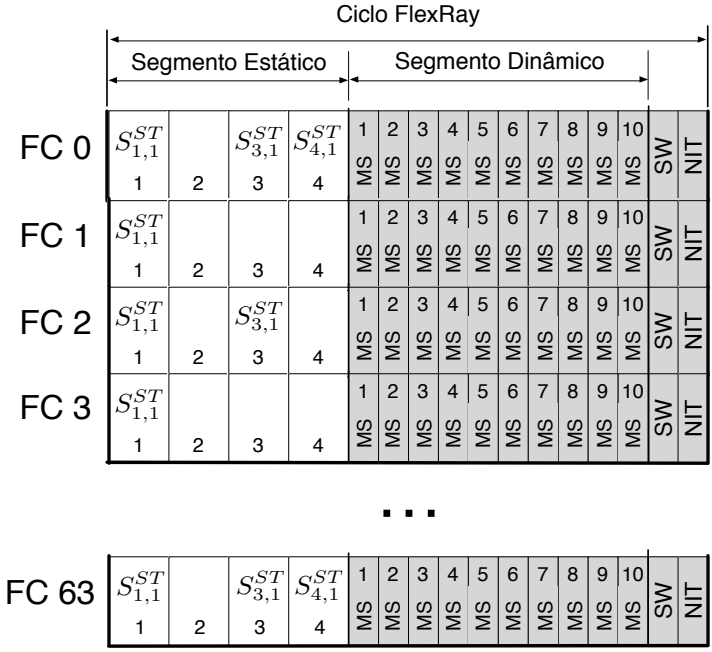

Figura 7 – Visão do Segmento Estático como uma matriz

No exemplo apresentado na Figura 7, o fluxo de mensagens $S^{ST}_{1,1}$ está associado ao *slot* 1, e transmite em todos os ciclos de comunicação. O fluxo $S^{ST}_{3,1}$ está associado ao *slot* 3 e transmite a cada dois FCs. Por fim, o fluxo $S^{ST}_{4,1}$ está associado ao *slot* 4 e transmite a cada quatro FCs.

Para cada nodo de um *cluster* FlexRay, um dos *slots* estáticos (definido pela constante $pKeySlotId$) pode ser designado para conter quadros do tipo especial *sync frame* relativos à sincronização e identificados por *pKeySlotUsedForSync*. Pode-se alocar *sync frames* específicos para serem quadros de inicialização da rede, recebendo a identificação de *pKeySlotUsedForStartup*.

O ST fornece um tempo de comunicação determinista, dado que é possível determinar exatamente quando um quadro será transmitido em um canal de comunicação, fornecendo, portanto, uma garantia de latência (ZURAWSKI, 2005).

Assim como o ST, o Segmento Dinâmico tem tamanho fixo e é dividido em *slots* (chamados de *slots dinâmicos*) que são alocados para os nodos da rede durante a fase de projeto.

Entretanto, a arbitragem no DN é baseada em um método mais flexível chamado *Flexible TDMA* (FTDMA).

Para que se possa detalhar o mecanismo do FTDMA, inicialmente é necessário explicar que o DN é dividido em um número de *minislots* (MS) que possuem o mesmo tamanho. A duração de um *minislot* é relativamente curta (quando comparados aos *slots* do ST), variando, segundo a especificação, entre 2 e 378 μs. O tamanho DN_{bus} do DN é dado em *minislots*, e o número de MS que compõe o DN de um dado *cluster* FlexRay é definido por $gNumberOfMinislots$. Cada MS contém um número $gdMinislot$ idêntico de *macroticks*. Tanto $gNumberOfMinislots$ como $gdMinislot$ são constantes globais para um dado *cluster* FlexRay. O número do MS atual em um dado FC é armazenado no contador local $zMinislot$. Apesar de existir um contador para cada canal de comunicação, eles são atualizados de forma simultânea.

Para cada canal de comunicação, cada nodo de rede mantém um contador de *slots* $vSlotCounter$. O valor armazenado em $vSlotCounter$ captura o valor do *slot* dinâmico atual e é atualizado ao final deste.

Cada nodo de rede possui um parâmetro $pLatestTx$. Este parâmetro é definido para cada nodo e representa o último instante no qual

o nodo pode transmitir no DN. O valor de $pLatestTx$ é dado por

$$\begin{aligned} pLatestTx &= gNumberOfMinislots \\ &\quad - aMinislotPerDynamicFrame + 1 \end{aligned} \tag{2.1}$$

sendo que $aMinislotPerDynamicFrame$ é o número de MS necessários para a transmissão da maior mensagem gerada no nodo (SCHMIDT; SCHMIDT, 2010b).

Diferente dos *slots* do segmento Estático que possuem tamanho fixo, a duração dos *slots* do segmento Dinâmico pode variar durante a execução do sistema, e seu tamanho é especificado em múltiplos inteiros de MS. Esse tamanho depende da transmissão ou não de quadros em determinado canal por um controlador, podendo resultar nas seguintes situações:

- Se não existem dados para transmitir e $zMinislot$ não atingiu $gdNumberOfMinislots$ (ou seja, o fim do DN), o *slot* dinâmico tem o tamanho de apenas um MS.
- Se existe um quadro para ser transmitido, e $zMinislot$ é menor ou igual que $pLatestTx$ do nodo, o *slot* dinâmico utiliza um número de MS grande o suficiente para acomodar a transmissão do quadro.
- Se existe um quadro para ser transmitido, $zMinislot$ não atingiu $gdNumberOfMinislots$ mas sim $pLatestTx$, então o quadro não é enviado e o *slot* dinâmico tem o tamanho de apenas um MS.

Segundo a especificação do FlexRay, diferente do que ocorre no ST, não é necessária a associação de um nodo com *slots* dinâmicos, nem mesmo a própria existência do DN em um *cluster*.

2.1.1.3 Formato do quadro

A visão geral de um quadro FlexRay é dada na Figura 8. Como pode ser observado, ele é composto por três segmentos: cabeçalho ou *header*, carga ou *payload* e cauda ou *trailer*.

O **cabeçalho** de um pacote FlexRay consiste de 5 bytes divididos em campos que contêm informações definidas durante a fase de projeto. A função de cada campo é listada abaixo:

Reserved bit: bit reservado para uso futuro do protocolo. Deve ser ignorado pelos receptores, e mantido em nível lógico 0 pelos nodos transmissores.

Payload preamble indicator: indica se o segmento de carga contém o vetor opcional para gerenciamento de rede ou o identificador de mensagem estendido.

Null frame indicator: utilizado para informar se o quadro atual contém dados utilizáveis.

Sync frame indicator: informa se o quadro está sendo utilizado pelos controladores para sincronização global de relógios.

Startup frame indicator: bit responsável por sinalizar se o quadro está envolvido em um processo de inicialização do FlexRay.

Frame ID: define em qual *slot* o quadro deve ser transmitido. Um mesmo *frame ID* não é utilizado mais de uma vez em cada canal durante um ciclo de comunicação. Sua faixa de valores é de 1 a 2047, e o ID 0 é considerado inválido. A associação entre um *frame ID* e um fluxo de mensagens é definida durante o projeto de um sistema. Métodos para definir tal associação serão discutidos no Capítulo 3, Seções 3.1 e 3.2.

Payload length: utilizado para indicar o tamanho do segmento de carga. O tamanho codificado neste campo equivale à metade do número de bytes de dados no segmento de carga, e pode variar de acordo com os seguintes fatores: quadros diferentes em um ciclo do segmento Dinâmico e canal de comunicação utilizado.

Header CRC: contém um código de verificação de redundância cíclica que é computado sobre os campos *sync frame indicator*, *startup frame indicator*, *frame ID* e *payload length*.

Cycle count: indica no momento da transmissão do quadro o valor de *vCycleCounter* (contador local que armazena o número do ciclo) do nodo transmissor.

O **segmento de carga** (*payload*) de um quadro FlexRay contém de 0 a 254 bytes, sendo estes os dados que estão sendo efetivamente trocados durante um ciclo de comunicação.

Para os quadros transmitidos no segmento Dinâmico, os dois primeiros bytes podem ser opcionalmente utilizados como um campo identificador de mensagem, permitindo aos nodos receptores filtrar ou direcionar dados com base em seu conteúdo. O bit *Payload preamble indicator* do cabeçalho indica o uso ou não desta funcionalidade.

De forma similar ao segmento Dinâmico, o segmento Estático do FlexRay também permite o uso opcional de parte do *payload* para funções de gerenciamento. Nesse caso, os doze primeiros bytes podem ser utilizados como um vetor de gerenciamento. O tamanho deste vetor é definido durante a fase de configuração do protocolo, e seu uso é informado através do bit *Payload preamble indicator* (FLEXRAY, 2015).

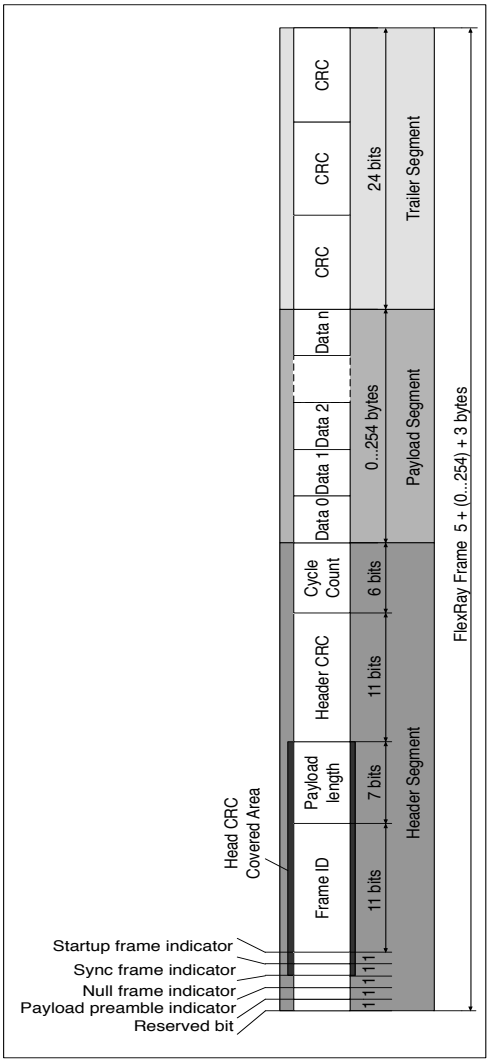

Figura 8 – Formato de um quadro FlexRay (FLEXRAY, 2015)

Um quadro FlexRay completo é protegido pelo ***trailer***, que consiste de um CRC com 24 bits.

O tamanho $C_{h,j}^{ST}$ em segundos de um quadro com $lm_{h,j}^{ST}$ bytes a ser transmitido no Segmento Estático do FlexRay, onde h indica o número de um nodo e j é o identificador do fluxo de mensagens, incluindo os *overheads*, é dado por:

$$C_{h,j}^{ST} = (88 \text{ bit} + lm_{h,j}^{ST} \cdot 10 \text{ bit})\tau_{bit}^{FR} \tag{2.2}$$

onde τ_{bit}^{FR} é o tempo necessário para a transmissão de um bit em um barramento FlexRay.

Já o tamanho $C_{h,j}^{DN}$ em segundos de um quadro com $lm_{h,j}^{DN}$ bytes a ser transmitido no DN (onde h indica o número de um nodo e j é o identificador do fluxo de mensagens), incluindo os *overheads* é dado por:

$$C_{h,j}^{DN} = (lm_{h,j}^{DN} \cdot 16 \text{ bit} + lm_{h,j}^{DN} \cdot 4 \text{ bit} + O_F)\tau_{bit}^{FR} \tag{2.3}$$

onde O_F é o *offset* do fluxo de mensagens.

2.1.1.4 Representação do tempo

A representação do tempo dentro de um nodo FlexRay é baseada em ciclos, *macroticks* (MT) e *microticks*, sendo que um *macrotick* é composto por um número inteiro de *microticks*, e um ciclo é composto por um número inteiro de *macroticks*, conforme pode ser observado na Figura 9.

Os *microticks* são unidades de tempo derivadas diretamente do relógio do controlador de comunicação de um nodo FlexRay, e seu tamanho pode variar de um controlador para outro.

Já os *macroticks* são unidades de tempo que são sincronizadas dentro de um *cluster* FlexRay e, dentro das tolerâncias previstas pelo protocolo, têm a mesma duração para todos os nodos de rede. A duração local de um *macrotick* pode variar de um nodo para outro, pois os *microticks* dependem do relógio local. Em cada nodo, o valor local de *macroticks* é armazenado na variável *vMacrotick*, sendo que este valor vai de 0 a $gMacroPerCycle - 1$.

O número de *macroticks* por ciclo é fixo e deve ser idêntico para todos os nodos em um *cluster*. O número do ciclo é armazenado localmente em um nodo pela variável *vCycleCounter*, sendo que essa é incrementada no início de cada ciclo de comunicação. O valor de *vCy-*

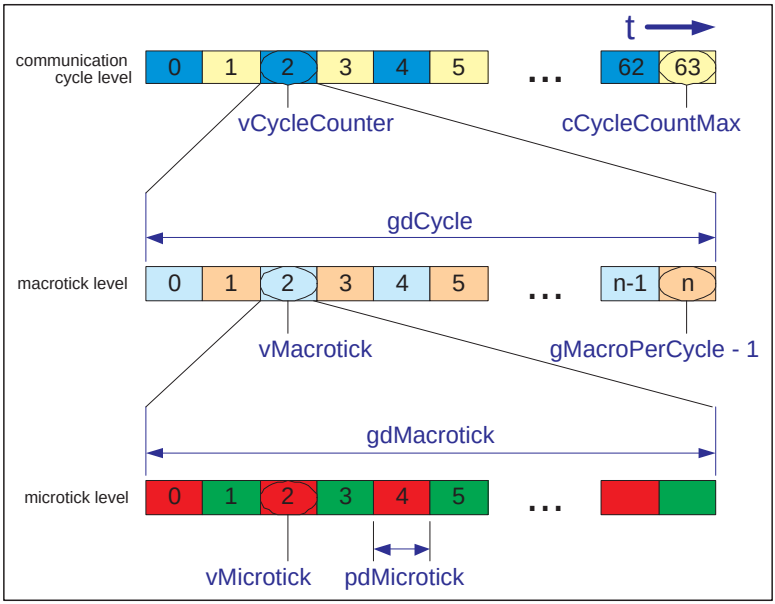

Figura 9 – Representação do tempo no FlexRay (FLEXRAY, 2015)

cleCounter vai de 0 a *cCycleCountMax*. Quando *cCycleCountMax* é atingido, *vCycleCounter* é reiniciada para zero no início do ciclo de comunicação seguinte.

O *tempo global* de um *cluster* é o tempo utilizado por todos os dispositivos dentro desse *cluster*. Segundo a especificação, o FlexRay não possui um valor absoluto ou de referência para o tempo global, sendo que cada nodo possui sua própria informação local para o tempo global.

O *tempo local* é o valor do relógio de um dispositivo específico no *cluster*, e é representado pelas variáveis *vCycleCount*, *vMacrotick* e *vMicrotick*. O tempo local é baseado na visão que o dispositivo tem do tempo global. Cada nodo conectado à rede usa o algoritmo de sincronização do FlexRay para adaptar seu tempo local ao tempo global.

2.1.1.5 Sincronização de relógios

Em um sistema distribuído, cada dispositivo possui seu próprio relógio. Devido a vários fatores, como oscilações na temperatura ou diferenças entre os componentes que definem a frequência, os relógios de diferentes dispositivos irão divergir. Em sistemas que utilizam redes com arbitragem do tipo TDMA, é importante que os relógios dos dispositivos estejam, dentro de certas tolerâncias, sincronizados. O limite máximo para a diferença de valores entre dois relógios é chamado de *precisão*.

O protocolo de comunicação FlexRay utiliza um mecanismo de sincronização de relógios no qual cada nodo da rede sincroniza a si mesmo em relação ao *cluster* com base na observação das transmissões feitas por outros nodos. Esta sincronização é feita através de dois processos principais. O processo de geração de *macroticks* (*Macrotick Generation Process*, MTG) controla os contadores de ciclo e *macroticks*, aplicando as correções de valores. O processo de sincronização de relógios (*Clock Synchronization Process*, CSP) realiza a medição e armazenamento de valores de desvio e o cálculo do *offset* e valores de taxa de correção.

A tarefa primária do processo de sincronização de relógios é garantir que as diferenças entre os nodos da rede fiquem dentro da precisão definida para o *cluster*. O FlexRay utiliza para a correção de relógios um método que combina correção de *offset* e correção de taxa (taxa ou *rate*, também chamada correção de frequência). Existem condições que devem ser seguidas para a sincronização dos relógios dos nodos em

uma rede FlexRay:

- As correções de *offset* e *rate* devem ser feitas do mesmo modo em todos os nodos.
- As correções de *offset* só podem ser realizadas durante o NIT e em ciclos com número ímpar. Deve iniciar no instante determinado pelo parâmetro *gOffsetCorrectionStart* e deve terminar antes do início do ciclo de comunicação seguinte.
- As modificações de *offset* são descritas pela variável *vOffsetCorrection*, onde é indicado o número de *microticks* que devem ser adicionados ao segmento de correção do NIT. O valor de *vOffsetCorrection* pode ser negativo e é calculado pelo algoritmo de sincronização de relógios. O cálculo de *vOffsetCorrection* é realizado a cada ciclo, mas as correções são aplicadas apenas no final de ciclos de comunicação com número ímpar.
- As correções de *rate* são descritas pela variável *vRateCorrection*, sendo um número inteiro de *microticks* (podendo ser negativo) que são adicionados ao número de *microticks* que foi configurado para cada ciclo de comunicação (*pMicroPerCycle*). O cálculo de *vRateCorrection* é feito pelo algoritmo de sincronização de relógios e inicia após o final do segmento Estático de ciclos com número ímpar.

Os valores para a correção do *rate* e *offset* são calculados através de um algoritmo chamado *Fault-Tolerant Midpoint Algorithm* (FTM). Após o cálculo dos valores pelo algoritmo, estes são verificados em relação aos valores limite definidos para as correções. Se os valores estiverem dentro dos limites estabelecidos, o relógio local é modificado de forma a sincronizar o nodo ao relógio global. Essa modificação é feita pelo ajuste do número de *microticks* que compõem cada *macrotick*.

2.2 AUTOMOTIVE OPEN SYSTEM ARCHITECTURE (AUTOSAR)

O *Automotive Open System Architecture* (AUTOSAR) é um padrão industrial aberto proposto por um consórcio de fabricantes e fornecedores de componentes para automóveis. O objetivo do consórcio é desenvolver e estabelecer um padrão de fato para arquiteturas eletrônicas para fins automotivos (AUTOSAR, 2012).

No AUTOSAR, uma aplicação é modelada como uma composição de componentes interconectados. Um mecanismo de comunicação

chamado *Virtual Functional Bus* (VFB) permite a interação entre os componentes. O VFB é ilustrado na metade superior da Figura 10. Durante a fase de projeto do sistema, os componentes são mapeados para as ECUs, e as conexões virtuais entre elas são traduzidas em conexões locais (em uma única ECU) ou em conexões que utilizam tecnologias de rede como os protocolos FlexRay ou CAN (Figura 10, metade inferior).

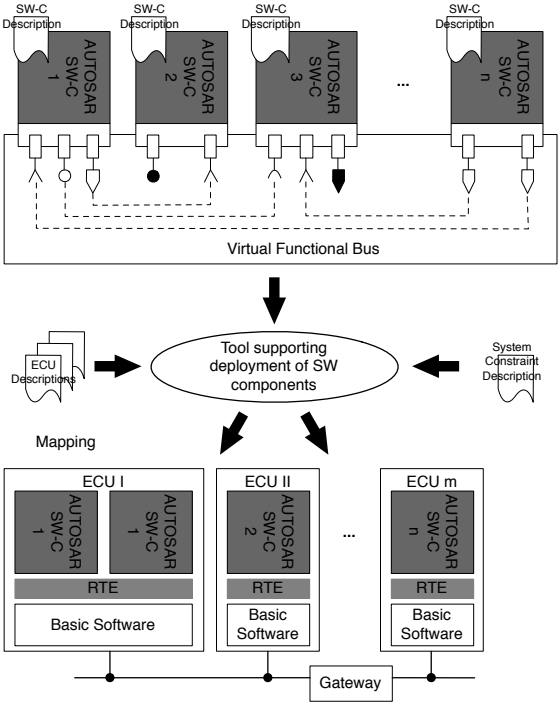

Figura 10 – *Virtual Functional Bus* (AUTOSAR, 2012)

O AUTOSAR utiliza uma arquitetura em camadas que garante a separação entre os aspectos funcionais e o *hardware* utilizado. Para cada ECU, o AUTOSAR define uma arquitetura com três camadas: ***Application Layer***, composta pelos *Software Components* (SW-C) que encapsulam total ou parcialmente uma funcionalidade automotiva; ***Real-Time Environment*** (RTE), que define a interface entre os SW-Cs e o restante do sistema da ECU; e o ***Basic Software Layer***, relativo aos componentes padronizados de software que não implementam funcionalidades específicas, mas oferecem serviços dependentes do *hard-*

ware para a camada superior (RTE). Devido à diversidade de módulos que compõe o AUTOSAR, neste documento a discussão será restrita àqueles que estão relacionados à comunicação de dados via rede.

A camada *Basic Software Layer* é composta pelas camadas *Services Layer*, *ECU Abstraction Layer*, *Complex Drivers* e *Microcontroller Abstraction Layer*.

- ***Services Layer*** oferece funcionalidades de sistema operacional, comunicação intra-veicular, serviços de gerenciamento de memória, serviços de diagnóstico e gerenciamento do estado da ECU.

- ***ECU Abstraction Layer*** é responsável pelas interfaces dos *drivers* do *Microcontroller Abstraction Layer*. É composta por *Onboard Device Abstraction*, *Memory Hardware Abstraction*, *Communication Hardware Abstraction* e *I/O Hardware Abstration*.

- ***Complex Drivers*** implementa o controle de sensores e atuadores que possuem acesso direto ao microcontrolador (através de interrupções específicas e/ou por serem periféricos dedicados do microcontrolador).

- ***Microcontroller Abstraction Layer*** é a camada mais baixa do *Basic Software*, e contém os *drivers* internos do sistema, como drivers para memória, comunicação e I/O.

As camadas do *Basic Software* são também divididas em grupos funcionais. Exemplos de grupos funcionais são o *Memory Services* (que fornece serviços de gerenciamento e controle de memória) e *Communication Services* (que fornece serviços de comunicação).

A comunicação entre componentes do AUTOSAR utiliza pacotes de dados chamados *Protocol Data Units* (PDUs). Cada PDU possui um prefixo que varia de acordo com a camada do AUTOSAR. Por exemplo, uma mensagem gerada por uma tarefa de um SW-C na camada de aplicação é empacotada em um $ISignalIPdu$. Um módulo chamado AUTOSAR COM empacota um $ISignalIPdu$ em um *Interaction Layer PDU* (I-PDU). Se um I-PDU é enviado através do módulo TP dos serviços de comunicação, o mesmo é empacotado em um *Network Layer PDU* (N-PDU). A interface de protocolo envia *Data Link Layer PDUs* (L-PDUs) para o *driver* do protocolo. E, finalmente, o *driver* do protocolo empacota o L-PDU em quadros que serão enviados para o barramento de rede. A convenção completa para a nomenclatura de PDUs pode ser encontrada em (AUTOSAR, 2012).

O grupo funcional chamado *Communication Services* agrupa os módulos responsáveis pela comunicação de rede intra-veicular. Estes módulos são interligados com os *drivers* de comunicação através da interface de comunicação do *driver*. O AUTOSAR Specification 4.0 define módulos para FlexRay, CAN, LIN e Ethernet.

Dentre outros, o *Communication Services* contém também os seguintes módulos:

- *AUTOSAR COM*, responsável pela interface entre o RTE e as camadas inferiores do sistema. Este módulo realiza o empacotamento/desempacotamento de sinais AUTOSAR individuais (ou grupos de sinais) em I-PDUs.

- *PDU Router*, responsável por rotear I-PDUs entre diferentes abstrações de controladores de comunicação e camadas superiores.

- O módulo *Transport Protocol* relacionado a cada protocolo suportado. O módulo TP do FlexRay (*FlexRay Transport Protocol*, ou FlexRay TP) é responsável pela segmentação e montagem de I-PDUs que não cabem no FlexRay L-PDU associado.

- *IPDU Multiplexer*, que fornece a possibilidade de adicionar informações para habilitar a multiplexação de I-PDUs (I-PDUs com conteúdos diferentes mas mesmo identificador).

No AUTOSAR, os módulos relacionados ao FlexRay são agrupados no *FlexRay Communication Services*. De forma similar, os módulos relacionados ao CAN são agrupados no *CAN Communication Services*. A relação entre os módulos destes dois grupos é ilustrada na Figura 11. Os prefixos que um PDU recebe em cada nível da arquitetura também estão representados na figura.

Em um nível inferior ao *Communication Services* existe a camada chamada *Communication Hardware Abstraction* (Comm HW Abstraction). Esta camada agrupa os módulos que abstraem a localização dos controladores de comunicação, e contém interfaces que fornecem mecanismos padronizados e idênticos para acesso a um barramento através de um controlador de comunicação. Por exemplo, o *FlexRay Interface (FrIf)* é a interface para o protocolo FlexRay. A especificação do FlexRay define módulos e interfaces para os protocolos FlexRay, CAN, TTCAN, LIN e Ethernet. As interfaces não fazem nenhum tipo de roteamento que depende do *payload* de um PDU, e o escalonamento das transmissões/recepções deve ser definidas durante a fase de configuração do sistema (essa definição também deve ser feita para

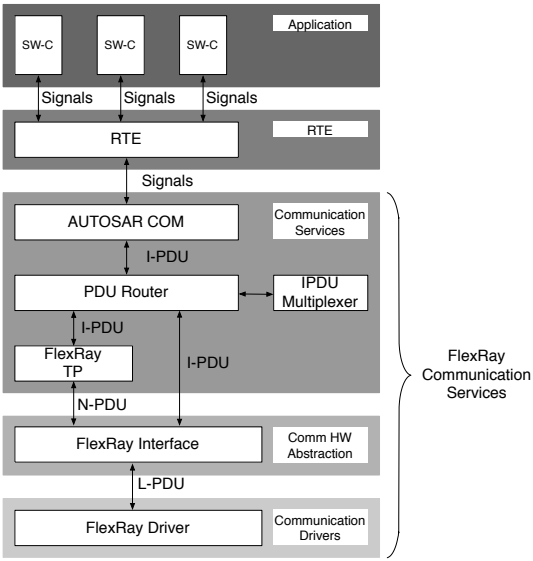

Figura 11 – AUTOSAR Basic Software relacionado ao FlexRay e ao CAN (Baseado em AUTOSAR (2012))

comunicações *event-driven*). O escalonamento das mensagens deve respeitar as particularidades de cada módulo. Por exemplo, se um I-PDU ou N-PDU deve ser transmitido em um barramento FlexRay, o mesmo deve estar associado com exatamente um identificador FlexRay, um ciclo base e uma repetição, e esta associação não pode ser modificada durante o modo de operação normal do FlexRay.

O AUTOSAR fornece definições para operações de *gateway*. A nível de PDUs, as operações de *gateway* são realizadas pelo módulo *PDU Router*. Este módulo pode ser utilizado para receber I-PDUs de um barramento fonte e enviar estes I-PDUs para um ou mais barramentos destino. Quando utilizado como *gateway*, o *PDU Router* pode:

- Encaminhar um I-PDU de um módulo de interface de comunicação para outro módulo de interface de comunicação (*gateway* 1:1).

- Encaminhar um I-PDU de um módulo de interface de comunicação para outros módulos de interface de comunicação (*gateway* 1:n).

- Encaminhar um I-PDU de um módulo TP para outro módulo TP (*gateway* 1:1).
- Encaminhar um I-PDU de um módulo TP para vários módulos TP (*gateway* 1:n).

No entanto, existem restrições no uso do *PDU Router* como *gateway*. Por exemplo, o *PDU Router*:

- Não suporta transferência de I-PDUs entre módulos TP e módulos de interface de comunicação (o I-PDU pode ser transferido apenas ou entre módulos TP ou apenas entre módulos de interface). Por exemplo, um I-PDU não pode ser recebido por uma interface CAN e transferido para o módulo TP do FlexRay.
- Não modifica o conteúdo do I-PDU.
- Não toma decisões de roteamento/*gatewaying* baseadas no conteúdo (*payload*) de um PDU.
- Não suporta roteamento de I-PDUs entre interfaces de comunicação quando é necessária a conversão da taxa de transmissão. Entretanto, esta funcionalidade é suportada em cooperação com os módulos da camada superior, como por exemplo o módulo COM.

O roteamento e o *gatewaying* de I-PDUs é realizado com base em tabelas estáticas que descrevem os atributos de cada I-PDU a ser transferido (origem e destino do I-PDU). Quando suportado, as tabelas podem ser atualizadas quando uma ECU está no modo de programação, ou selecionadas na inicialização do *PDU Router*, e são geralmente definidas durante o projeto do sistema.

2.3 SUMÁRIO

Neste capítulo foram apresentados os principais conceitos relacionados ao protocolo de comunicação FlexRay. Também foi feita uma breve apresentação do padrão AUTOSAR, um padrão bastante utilizado no desenvolvimento de sistemas automotivos.

No próximo capítulo serão descritos e comentados trabalhos da literatura que estão relacionados com a análise de escalonamento e de tempo de resposta do protocolo FlexRay.

3 PRINCIPAIS TRABALHOS SOBRE ESCALONAMENTO EM SISTEMAS FLEXRAY

No capítulo anterior, foram apresentadas as principais características do protocolo FlexRay.

De forma complementar ao Capítulo 2, neste capítulo é apresentada uma revisão do estado da arte relacionado a propostas de técnicas para análise de escalonabilidade e tempo de resposta nos segmentos Estático e Dinâmico do FlexRay.

Neste capítulo, procurou-se priorizar a apresentação de artigos publicados em periódicos de editoras reconhecidas como, por exemplo, IEEE (IEEE, 2015) e Springer (SPRINGER, 2015). É importante salientar que, quando existem duas versões de um mesmo artigo, uma publicada em um congresso e outra publicada em um periódico, apenas a versão publicada no periódico é apresentada. Também é importante salientar que, assim como no Capítulo 2, este capítulo apresenta conceitos que são comuns a todas as propostas apresentadas nesta Tese. Quando um conceito específico for necessário para apenas uma das propostas apresentadas nos demais capítulos que compõem esta tese, tal conceito será apresentado naquele capítulo.

3.1 TRABALHOS RELEVANTES SOBRE O SEGMENTO ESTÁTICO

Esta seção apresenta trabalhos relevantes da literatura relacionada ao escalonamento e a análise de tempo de resposta do segmento Estático do FlexRay.

O trabalho apresentado em (GRENIER et al., 2008) aborda o problema do empacotamento de sinais em quadros que serão transmitidos no segmento Estático. No trabalho, são propostos dois algoritmos para empacotamento de sinais em quadros, sendo que o número de *slots* estáticos alocados é um parâmetro a ser minimizado para maximizar a utilização da banda. A proposta considera o caso onde os processos que geram os sinais não estão sincronizados com o ciclo FlexRay, e tem as premissas de que os sinais serão transmitidos e recebidos através de uma arquitetura AUTOSAR (AUTOSAR, 2012), e que o tamanho do ciclo FlexRay e do segmento Estático são conhecidos *a priori*, não sendo passíveis de otimização. O padrão AUTOSAR foi discutido no Capítulo 2, Seção 2.2, página 53.

Em (DING et al., 2008) é proposta uma abordagem que utiliza

algoritmos genéticos para encontrar escalonamentos viáveis para o segmento Estático. O trabalho considera que o período das tarefas e dos fluxos de mensagens do sistema são múltiplos inteiros do ciclo FlexRay.

Apesar do principal foco do trabalho apresentado em (POP et al., 2008) ser o escalonamento do Segmento Dinâmico do FlexRay, o trabalho apresenta três propostas para o escalonamento do ST. A primeira simplesmente considera que é alocado um *slot* estático por nodo, e então tenta definir o tamanho do FC com base nos requisitos do DN. A segunda proposta é uma heurística gulosa que explora diferentes alternativas para o tamanho do ST, considerando as restrições impostas pela norma com relação ao número de *slots* estáticos em cada FC. Por fim, a terceira proposta usa técnicas de *simulated annealing* para definir um escalonamento ótimo para os fluxos de mensagens, em uma formulação que considera os tamanhos do FC, do ST e do DN como parâmetros.

Lukasiewycz et al. (2009) aborda o problema de escalonar o segmento Estático obedecendo às restrições impostas pelo AUTOSAR. O trabalho propõe o uso de heurísticas gulosas para transformar o problema do escalonamento em um *Bin Packing Problem* (BPP). O BPP é um problema de otimização bastante conhecido que consiste em definir o número mínimo de itens que são necessários para que uma caixa com uma capacidade mínima conhecida seja considerada cheia. Na proposta, o BPP é resolvido com o uso de uma formulação para programação linear inteira (*Integer Linear Programming*, ILP) cujo objetivo é minimizar o número de *slots* estáticos alocados. O método proposto considera que os tamanhos do ciclo FlexRay e de cada segmento são predefinidos.

O problema de escalonamento do segmento Estático também é abordado em (SCHMIDT; SCHMIDT, 2009). No trabalho, inicialmente é feita uma discussão sobre como definir o tamanho FC_{bus} de um ciclo FlexRay. De acordo com os autores, todos os sinais do sistema devem ser escalonados em múltiplos inteiros de ciclo FlexRay, sendo demonstrado que é favorável escolher o tamanho do FC como sendo o Máximo Divisor Comum (MDC) dos períodos ou intervalos mínimos entre chegadas (*Minimum Interarrival Time*, MIT) dos sinais alocados no ST e no DN. O trabalho divide o problema do escalonamento do ST em dois subproblemas. O primeiro subproblema é o empacotamento de sinais periódicos em quadros para o segmento Estático, e para este problema é proposta uma formulação ILP cujo objetivo é maximizar a alocação de sinais de um dado sistema em mensagens estáticas. O segundo problema é a definição de um escalonamento de quadros que minimize o

jitter das mensagens e o número de *slots* alocados, sendo propostas no trabalho duas soluções baseadas em ILP, uma considerando o caso de escalonamento sem *jitter* e outra considerando o caso onde o *jitter* deve ser o mínimo possível.

Uma extensão de (SCHMIDT; SCHMIDT, 2009) é apresentada em (SCHMIDT; SCHMIDT, 2010a), onde é proposto o uso da repetição de ciclos como um parâmetro para capturar a relação entre o *jitter* e o número de identificadores estáticos alocados. O trabalho é baseado no fato de que o FlexRay permite multiplexação, o que significa que as mensagens geradas por um mesmo fluxo não precisam ser escalonadas em todos os ciclos FlexRay, mas apenas em ciclos pré-determinados que são identificados por um ciclo base e uma repetição (maiores informações sobre o ciclo base e a repetição são dadas na Seção 2.1.1.2 deste capítulo). Os parâmetros do ciclo base e da repetição são utilizados em uma formulação ILP cujo objetivo é atribuir uma repetição apropriada para cada fluxo de mensagens enquanto minimiza o *jitter* das mensagens e o número de identificadores alocados. A formulação considera que são conhecidos *a priori* os valores de FC_{bus}, ST_{bus}, DN_{bus} e *gdStaticSlot*, bem como a forma como os sinais são empacotados em quadros.

O trabalho descrito em (PARK; SUNWOO, 2011) apresenta um método que utiliza técnicas de otimização para determinar tamanhos ótimos para o ST, para o DN e para o FC. O trabalho considera cenários onde as tarefas executadas em cada nodo não estão sincronizadas com o ciclo FlexRay (no chamado método de escalonamento assíncrono), e também considera que o tempo de transmissão no pior caso (*worst-case transmission time*, *wctt*) de fluxos de mensagens do ST e do DN são parâmetros para o problema de otimização. De forma complementar, no artigo é proposto um método simples para determinar o *wcrt* de mensagens transmitidas no DN.

Zeng et al. (2011) propõem um *framework* baseado em programação linear inteira mista (*Mixed Integer Linear Programming*, MILP) para definir o empacotamento de sinais em quadros, fazer a atribuição de $FrameIDs$ estáticos para os fluxos de mensagens, definir o escalonamento de mensagens e definir a sincronização entre os sinais do sistema e as tarefas considerando uma arquitetura AUTOSAR. A abordagem proposta considera que o número de *slots* estáticos alocados é um parâmetro a ser otimizado, assumindo que os valores de FC_{bus}, ST_{bus}, DN_{bus} e $gdStaticSlots$ são informados. O trabalho apresenta uma discussão sobre os dois possíveis padrões de sincronização entre tarefas e sinais (escalonamentos síncronos e assíncronos) e sua relação com o

AUTOSAR. Também são feitas várias considerações sobre as possíveis definições para uma rede FlexRay e suas relações com o AUTOSAR.

Em (HANZáLEK et al., 2011) são propostos métodos heurísticos para os problemas de empacotamento de sinais em quadros e escalonamento do segmento Estático do FlexRay quando respeitadas as restrições impostas pelo AUTOSAR. O trabalho considera que o empacotamento de sinais é feito nos componentes padronizados do AUTOSAR que são responsáveis pela comunicação de rede, e não a nível de aplicação, como considerado em trabalhos anteriores.

O Trabalho de Conclusão de Curso em (VAZ,) descreve a implementação de uma ferramenta para automatizar a configuração de sistemas FlexRay. No entanto, o trabalho é basicamente a implementação das técnicas propostas em (SCHMIDT; SCHMIDT, 2010a) e (SCHMIDT; SCHMIDT, 2010b).

Kang et al. (2012) propõem dois algoritmos para o empacotamento de sinais em quadros do ST. Estes algoritmos utilizam técnicas de otimização para minimizar o uso da largura de banda. Ambos os algoritmos assumem que os sinais podem ter diferentes períodos, e que sinais com diferentes períodos podem ser empacotados em um mesmo quadro, desde que estes sinais sejam oriundos do mesmo nodo de rede. A proposta assume que todos os sinais devem ser empacotados em quadros cujos períodos são múltiplos inteiros de FC_{bus}.

Dvorak e Hanzalek (2015) apresentam uma heurística para escalonar um sistema FlexRay onde, através do uso dois dois canais de comunicação disponibilizados pelo FlexRay, o uso de *slots* estáticos é minimizado. A exemplo de outros trabalhos relacionados, a heurística proposta faz uso de uma formulação ILP. Segundo o conhecimento do autor desta Tese, a proposta apresentada em (DVORAK; HANZALEK, 2015) é um dos primeiros trabalhos a explorar o uso dos dois canais de comunicação para fins de aumento de largura de banda.

Pode-se destacar a existência de alguns aspectos em comum nos trabalhos relatados. Parte deles considera que os períodos dos sinais ou fluxos de mensagens são definidos como múltiplos inteiros do ciclo FlexRay (por exemplo, (DING et al., 2008; SCHMIDT; SCHMIDT, 2009, 2010a; ZENG et al., 2011; KANG et al., 2012)). Esta suposição não é baseada em restrições impostas pela especificação do FlexRay, e pode restringir as possibilidades de projeto e também aumentar o *jitter* das mensagens.

De forma similar, algumas das abordagens propostas assumem que os tamanhos do ciclo, do DN e do ST são conhecidos *a priori* e não estão sujeitos a otimizações (GRENIER et al., 2008; LUKASIEWYCZ et al.,

2009; ZENG et al., 2011).

Existem também trabalhos como, por exemplo, (GRENIER et al., 2008; SCHMIDT; SCHMIDT, 2009, 2010a; ZENG et al., 2011), que assumem a existência de sincronização entre as tarefas das ECUs, sinais ou fluxos de mensagens e o ciclo FlexRay. Entretanto, uma solução que considere escalonamento assíncrono pode ter um alto interesse prático, especialmente quando se considera o mapeamento de fluxos de mensagens projetados para CAN em *frames* FlexRay (PARK; SUNWOO, 2011; ZENG et al., 2011).

Outra característica em comum nos trabalhos destacados acima é que muitos deles utilizam técnicas de otimização (por exemplo, ILP or MILP) para definir conjuntos ótimos de parâmetros (ótimos no sentido de que os parâmetros resultantes são ótimos para o conjunto de sinais ou fluxos de mensagens) (DING et al., 2008; LUKASIEWYCZ et al., 2009; SCHMIDT; SCHMIDT, 2009, 2010a; ZENG et al., 2011). Mas é reconhecido que métodos baseados em otimização têm um alto custo de computação, o que pode ser inaceitável na prática mesmo quando no caso em que as definições do sistema são feitas *off-line* (POP et al., 2008).

Finalmente, existe uma característica comum a todos os trabalhos reportados acima: eles assumem que os sinais ou fluxos de mensagens das ECUs são associados a *slots* estáticos e repetições de ciclo únicos, e esta associação não é alterada durante a execução do sistema.

Além dos trabalhos citados acima, a literatura apresenta diversas outras propostas, muitas das quais podem ser encontradas através de uma busca em mecanismos de indexação como o IEEE Xplore (IEEE, 2015), relacionadas ao escalonamento do segmento Estático do FlexRay. Entretanto, tais propostas, em geral, são variações dos métodos listados neste capítulo e utilizam técnicas de otimização para escalonar o ST considerando a associação de sinais ou fluxos de mensagens a *slots* e repetições de ciclos exclusivas.

3.2 TRABALHOS RELEVANTES SOBRE O SEGMENTO DINÂMICO

Nesta seção serão resumidos os trabalhos da literatura sobre escalonamento e análise de tempo de resposta do segmento Dinâmico do FlexRay considerados relevantes para esta Tese. No final da seção, serão feitas considerações sobre os trabalhos listados.

Cena e Valenzano (2006) apresentam uma análise das principais características do FTDMA. Apesar do foco do trabalho ser o protocolo Byteflight, algumas conclusões podem ser aplicadas ao segmento

Dinâmico do FlexRay, pois ambos utilizam um mecanismo MAC semelhante. No trabalho são apresentadas discussões sobre o desempenho do Byteflight, sendo propostas equações para calcular, por exemplo, o número máximo de mensagens que podem ser transmitidas em um barramento que utilize um mecanismo MAC baseado no FTDMA. Uma das principais conclusões do trabalho é que o FTDMA troca eficiência por flexibilidade, o que pode ser um problema para aplicações práticas que utilizem o DN devido ao máximo número de $FrameIDs$ do segmento (2048, segundo a especificação do FlexRay (FLEXRAY, 2015)). Entretanto, o método não pode ser aplicado diretamente ao FlexRay por não considerar a influência do ST (OUEDRAOGO; KUMAR, 2014).

No trabalho apresentado em (POP et al., 2008), os autores propõem a transformação do problema de definir o *wctt* de fluxos de mensagens que transmitem no DN do FlexRay em um BPP no qual se considera que os MS e as mensagens dinâmicas que podem ser transmitidas em um ciclo são itens, um segmento Dinâmico equivale a uma caixa, $pLatestTx$ é o mínimo requerido para que uma caixa seja considerada cheia. Pop et al. (2008) propõe dois métodos para definir o *wctt* de mensagens esporádicas, um que utiliza uma formulação ILP e outro baseado em uma heurística que assume uma simplificação no MAC do DN, sendo que ambas as propostas são baseadas em métodos existentes para resolução do BPP. Entretanto, o trabalho assume que $pLatestTx$ é definido para cada fluxo de mensagem, em vez de ser definido para cada nodo, o que pode levar a resultados otimistas para determinados conjuntos de fluxos.

Em (SCHMIDT; SCHMIDT, 2010b) o método proposto em (POP et al., 2008) é questionado, sendo proposta uma nova formulação ILP que aperfeiçoa as limitações do método anterior. De acordo com os autores do novo método, a proposta apresentada em (POP et al., 2008) considera o uso de um intervalo de tempo para a análise, e este intervalo deve ser cuidadosamente escolhido, requer um grande número de variáveis, resultando em uma complexa formulação ILP, e não reforça o fato de que ciclos consecutivos devem ser preenchidos com mensagens de alta prioridade, podendo levar a um *wctt* otimista para algumas mensagens dinâmicas. A nova formulação proposta corrige estas limitações, e, como complemento ao método para análise do tempo de resposta, (SCHMIDT; SCHMIDT, 2010b) apresenta a proposta de um algoritmo para atribuir identificadores aos fluxos de mensagens do segmento Dinâmico. É importante ressaltar que esse método considera um modelo de sistema onde as mensagens são geradas de forma esporádica.

Em (ZENG et al., 2010) o método em (POP et al., 2008) também

é questionado. De forma semelhante a (SCHMIDT; SCHMIDT, 2010b), é proposto um novo método para análise do tempo de resposta de mensagens e também um método para atribuição de identificadores aos fluxos de mensagens. A nova proposta utiliza uma combinação de método heurístico e formulação ILP, e considera um modelo de sistema onde as mensagens são esporádicas com *jitter*.

Tanasa et al. (2012) abordam a análise de escalonabilidade e tempo de resposta quando é utilizada multiplexação de *slots* no DN, isto é, quando dois ou mais fluxos de mensagens de um nodo compartilham um $FrameID$ do segmento Dinâmico. Nesse caso, cada fluxo de mensagem também deve estar associado i) a um ciclo base (ou ciclo inicial) no qual o fluxo pode transmitir e ii) a uma taxa de repetição que indica o intervalo mínimo (em inteiros de FC) entre duas transmissões do fluxo. O trabalho demonstra que definir o número de FC completos utilizados por mensagens com maior prioridade que uma mensagem qualquer sob análise quando é utilizada multiplexação de *slots* dinâmicos é um problema similar a um caso especial do BPP chamado *bin covering problem with conflicts*. Por fim, os autores propõem duas técnicas de análise de tempo de resposta, uma para o caso sem multiplexação de *slots* e outra para o caso onde multiplexação é utilizada. Ambos os métodos propostos utilizam uma formulação para ILP.

Ouedraogo e Kumar (2014) apresentam outra proposta para obter o *wctt* de fluxos escalonados no DN. No trabalho, os autores argumentam que a proposta apresentada por Pop et al. (2008) é pessimista por utilizar um método onde a formulação ILP é executada de forma iterativa (apesar do método proposto em (POP et al., 2008) apresentar o já citado problema em relação ao valor de $pLatestTx$ de cada fluxo). Os autores também argumentam que o método proposto em (SCHMIDT; SCHMIDT, 2010b) pode levar a *wctts* otimistas devido às restrições impostas sobre os tempos entre as transmissões de mensagens.

Em (HAGIESCU et al., 2007) é proposto um método que utiliza o *Real-Time Calculus* (RTC) (THIELE et al., 2000) para calcular um limite superior para o *wcrt* de mensagens dinâmicas. De acordo com o trabalho, os modelos RTC existentes não podem ser aplicados diretamente ao DN devido às restrições impostas pelo MAC do segmento, sendo proposto um novo modelo que utiliza transformações algorítmicas na curva de serviço inferior. A curva de serviço resultante da transformação fornece um limite inferior para o serviço fornecido pelo DN, e pode ser utilizada para computar as propriedades temporais das mensagens transmitidas no segmento.

A abordagem proposta em (HAGIESCU et al., 2007) é questionada

diretamente em (CHOKSHI; BHADURI, 2010), onde é demonstrado que o método anterior tem falhas que podem levar ao cálculo de um *wctt* otimista. Os autores propõem um novo modelo para analisar o tempo de resposta de mensagens transmitidas no DN, também utilizando transformações algorítmicas nas curvas utilizadas em um modelo RTC. O novo modelo proposto é baseado no fato de que o tempo de resposta é composto por um tempo de espera e um tempo de comunicação.

Kim e Park (2009) apresentam uma abordagem que considera distribuições probabilísticas para escalonar o DN. Na proposta, a probabilidade de atraso de um quadro e a distribuição dos MS não utilizados são utilizados como métricas para definir o escalonamento do DN. O trabalho, entretanto, não especifica quais características temporais dos fluxos devem consideradas quando a proposta é aplicada.

Em (TANASA et al., 2013) também é considerado um modelo probabilístico do DN. Na proposta, é utilizada uma formulação para Programação Linear Inteira Mista (*Mixed-Integer Linear Programming*, MILP) que computa a relação de perda de *deadlines* (*Deadline Miss Ratio*) de mensagens. Essa relação é, posteriormente, utilizada como métrica para avaliar o escalonamento de fluxos de mensagens no DN.

Pop et al. (2008) demonstram que obter o *wcrt* de fluxos de mensagens com características de *hard real-time* escalonados no DN é um problema *NP-Hard* semelhante ao *bin packing problem* (BPP). Por este motivo, muitos dos trabalhos que consideram modelos de sistema onde as mensagens do segmento Dinâmico possuem características de *hard real-time* utilizam ferramentas para problemas de otimização como, por exemplo, ILP ou MILP (POP et al., 2008; SCHMIDT; SCHMIDT, 2010b; ZENG et al., 2010; TANASA et al., 2012; OUEDRAOGO; KUMAR, 2014). Segundo o conhecimento dos autores desta tese, a literatura apresenta diversas outras propostas que seguem esta linha de pesquisa. Entretanto, os trabalhos mais recentes são meramente variações das formulações ILP ou MILP já existentes.

Hagiescu et al. (2007) e Chokshi e Bhaduri (2010) seguem uma linha de pesquisa que utiliza RTC para modelar o DN. Entretanto, (BONET et al., 2011) chega à conclusão de que o RTC não é adequado para modelar o DN devido às particularidades deste segmento do FlexRay, especialmente quando considerado um modelo de sistema onde os fluxos de mensagens são esporádicos ou aperiódicos.

Finalmente, uma terceira linha de pesquisa considera modelos probabilísticos para o escalonamento ou análise de tempo de transmissão de fluxos de mensagens que transmitem no DN. Segundo o conhecimento dos autores desta, apenas (KIM; PARK, 2009) e (TANASA et al.,

2013) seguem esta linha, mas o foco destes trabalhos é a obtenção da taxa de perda de *deadlines* (*deadline miss ratio*) dos fluxos de mensagens.

3.3 SUMÁRIO

Neste capítulo foram apresentados e discutidos trabalhos da literatura sobre a análise de escalonamento e de tempo de resposta no protocolo FlexRay.

Os conceitos apresentados neste capítulo, bem como os conceitos apresentados no Capítulo 2, foram utilizados nas propostas que compõem a contribuição científica desta tese, como será visto nos próximos capítulos deste documento.

4 ARQUITETURA DE REFERÊNCIA PARA ANÁLISE DE REDES HÍBRIDAS

Neste capítulo é descrito o modelo de sistema e estabelecidas as notações que serão utilizadas nos demais capítulos deste documento. O modelo de sistema descrito pode ser utilizado na análise temporal de sistemas veiculares compostos por redes heterogêneas onde barramentos que utilizam protocolos diferentes são interligados através de *gateways*.

Seguindo o conceito de blocos padronizados presente no padrão AUTOSAR (AUTOSAR, 2012), o modelo é composto por componentes padronizados que possuem características temporais como intervalo mínimo entre chegadas (MIT), tempo de execução ou transmissão, e *deadline*. O modelo utiliza quatro componentes básicos, que são interconectados para formar um sistema. Tais componentes são os nodos de rede, as tarefas, os fluxos de mensagens e as atividades. Cada componente, bem como sua representação e definição formal, será apresentado nas seções que compõem este capítulo.

O modelo considera que cada nodo de rede possui seu próprio escalonador e implementa sua própria estratégia de escalonamento. Também considera que não existe sincronização entre os diversos componentes do sistema, resultando em um **modelo de sistema com escalonamento assíncrono**. Zeng et al. (2011) define que, no contexto de sistemas automotivos, um modelo de sistema com escalonamento assíncrono é aquele onde não existem sincronizações entre o Ciclo FlexRay e a geração de mensagens ou a ativação de uma tarefa.

Para facilitar o entendimento, a descrição do modelo apresenta um exemplo no qual segmentos de rede que utilizam o protocolo FlexRay estão interligados a segmentos que utilizam o protocolo Controller Area Network (CAN), sendo esta interligação feita através de um tipo especial de nodo chamado *gateway*.

O restante deste capítulo está organizado como se segue. Na Seção 4.1 é inicialmente feita uma breve descrição sobre o protocolo CAN, já que o mesmo é utilizado nos exemplos apresentados neste capítulo. Na sequência, são apresentados os diversos componentes que compõem a arquitetura (Seções 4.2 a 4.5), e um exemplo típico (Seção 4.6). Por fim, na Seção 4.7 são feitas considerações sobre o capítulo.

4.1 CONTROLLER AREA NETWORK

O Protocolo CAN - *Controller Area Network*, é um barramento de rede projetado para fornecer comunicação simples, robusta e eficiente em aplicações automotivas. Desde a introdução do protocolo na década de 1980, o CAN se tornou um padrão de fato na indústria automotiva. Além disso, através dos anos o CAN passou a ser adotado em diferentes campos de aplicação, e estimava-se que mais de 400 milhões de nodos CAN seriam vendidos a cada ano (NAVET; SIMONOT-LION, 2008; ZURAWSKI, 2005). O CAN é definido pelo *CAN Specification Version 2.0* (Bosch, 1991) e pela norma ISO Standard 11898 (ISO11898-1, 2003).

A popularidade do CAN, aliada a limitações no protocolo, levou ao desenvolvimento de novos protocolos baseados no mesmo, como, por exemplo, o *Time-Triggered CAN* (TT-CAN) (LEEN; HEFFERNAN, 2002), o *Flexible Time-Triggered CAN* (FTT-CAN) (ALMEIDA et al., 1999) ou o *CAN with Flexible Data-rate CAN with Flexible Data-rate* (CAN-FD) (Bosch, 2015). Apesar desta diversidade, este trabalho ficará restrito à versão clássica do CAN, definida pela norma ISO 11898 (ISO11898-1, 2003). A popularidade do CAN também se reflete na quantidade de trabalhos de pesquisa relacionados. Uma simples busca com o termo "Controller Area Network" em um mecanismo de indexação como o IEEE Xplore (IEEE, 2015) resulta em mais de 3.000 entradas apontando para a literatura existente.

A especificação do CAN define um barramento serial digital assíncrono com taxas de transmissão de até 1Mb/s. O protocolo utiliza um controle de acesso ao meio (MAC) baseado em *Carrier Sense Multiple Access/Non Destructive Bitwise Arbitration* (CSMA/BA). Nesta técnica, o protocolo CAN faz com que os nodos aguardem até o barramento estar livre para transmitir. Se dois ou mais nodos iniciarem a transmissão ao mesmo tempo, um nodo pode determinar se está transmitindo a mensagem de prioridade mais alta ou se deve parar a transmissão e aguardar através do monitoramento de cada bit transmitido na rede.

A especificação do CAN também define dois formatos de quadros (formato *standard* e estendido), sendo que a principal diferença entre eles está no tamanho do campo identificador (*ID field*). A especificação também define quatro tipos de quadro (*data*, *remote*, *error* e *overload*). O quadro tipo *data* é utilizado com *payload* real. O quadro *remote* é semelhante ao tipo *data*, mas não carrega dados e é utilizado para requisitar informações de outros nodos da rede. Quadros tipo *error* são utilizados para notificar os nodos sobre erros na rede, e os quadros

tipo *overload* podem ser utilizados para sinalizar que as operações da rede devem retardadas devido à sobrecarga em nodos lentos (DAVIS et al., 2007; ZURAWSKI, 2005).

Para garantir um nível de sincronização satisfatório entre os nodos da rede, o CAN se baseia em uma técnica chamada *bit stuffing*. Neste método, sempre que forem transmitidos cinco bits com o mesmo valor, o nodo transmissor insere um bit adicional com o valor complementar. Estes bits adicionais são removidos pelo nodo receptor. Os bits adicionais aumentam o tempo de transmissão da mensagem e devem ser considerados no cálculo de seu *wctt* (DAVIS et al., 2007).

Considera-se que a análise de escalonamento e tempo de resposta no CAN iniciou com o trabalho apresentado em (TINDELL et al., 1995). Naquele trabalho, os autores demonstram que o *wcrt* de mensagens CAN pode ser calculado, permitindo assim que a indústria projete sistemas CAN com altos níveis de utilização da rede (DAVIS et al., 2007). Mas apesar do trabalho ter sido citado em mais de 400 outros trabalhos, foi descoberto mais tarde que a análise original possui um problema, e uma proposta de correção para o método foi apresentada em (DAVIS et al., 2007).

O método proposto em (DAVIS et al., 2007) utiliza o conjunto de equações para escalonamento com prioridade fixa não-preemptiva (GEORGE et al., 1996) e o conceito de *busy period* proposto em (LEHOCZKY, 1990) para derivar um correto *wcrt* de mensagens CAN. O novo método e as equações relevantes são resumidos a seguir.

De acordo com Davis et al. (2007) o tempo de transmissão $R_{h,j}^{CAN}$ de mensagens geradas por um fluxo de mensagens CAN $S_{h,j}^{CAN}$, no pior caso, é composto por três elementos: *release jitter máximo* $J_{h,j}^{CAN}$, tempo de transmissão $C_{h,j}^{CAN}$ e atraso $w_{h,j}^{CAN}$ (Equação 4.1).

$$R_{h,j}^{CAN} = J_{h,j}^{CAN} + C_{h,j}^{CAN} + w_{h,j}^{CAN} \tag{4.1}$$

O tempo de transmissão $C_{h,j}^{CAN}$ de uma mensagem $m_{h,j}^{CAN,l}$ CAN contendo $lm_{h,j}^{CAN}$ *bytes* de dados e identificador com 11 bits (formato CAN padrão) é dado pela Equação 4.2, onde τ_{bit}^{CAN} é o tempo de transmissão de um bit em uma rede CAN.

$$C_{h,j}^{CAN} = \left(55 + 10lm_{h,j}^{CAN}\right)\tau_{bit}^{CAN} \tag{4.2}$$

De forma similar, o tempo de transmissão máximo para um qua-

dro no formato estendido (identificador de 29 bits) é dado por

$$C_{h,j}^{CAN} = \left(80 + 10lm_{h,j}^{CAN}\right)\tau_{bit} \tag{4.3}$$

O atraso $w_{h,j}^{CAN}$ é composto por dois elementos: bloqueio $B_{h,j}^{CAN}$ causado por mensagens geradas por fluxos com prioridades mais baixa que $S_{h,j}^{CAN}$ e interferência causada por fluxos com prioridade mais alta que $S_{h,j}^{CAN}$. O valor de B_m é dado pela Equação 4.4 onde $lp(m_{h,j}^{CAN,l})$ é o conjunto contendo os fluxos de mensagens com prioridade mais baixa que $S_{h,j}^{CAN}$.

$$B_{h,j}^{CAN} = \max_{x \in lp(S_{h,j}^{CAN})}(C_x^{CAN}) \tag{4.4}$$

Se existe mais de uma mensagem gerada por $S_{h,j}^{CAN}$ durante um *busy period* de $S_{h,j}^{CAN}$, então é necessário determinar o *wcrt* de cada mensagem para que seja possível determinar o maior *wcrt* dentre elas. Para isso, primeiro é necessário determinar o tamanho Δ_t do *busy period* de $S_{h,j}^{CAN}$ utilizando a Equação 4.5, que inicia com $\Delta_t^0 = C_{h,j}^{CAN}$ e termina quando $\Delta_t^{y+1} = \Delta_t^y$ (onde y é um índice que armazena o valor da iteração atual). O número Q_m de mensagens geradas por $S_{h,j}^{CAN}$ em um *busy period* é dado pela Equação 4.6.

$$\Delta_t^{y+1} = B_{h,j}^{CAN} + \sum_{\forall x \in hep(S_{h,j}^{CAN})} \left\lceil \frac{\Delta_t^y + J_x^{CAN}}{P_x^{CAN}} \right\rceil C_x \tag{4.5}$$

$$Q_m = \left\lceil \frac{\Delta_t + J_{h,j}^{CAN}}{P_{h,j}^{CAN}} \right\rceil \tag{4.6}$$

Nas Equações 4.7, 4.8 e 4.9, é utilizado o índice q para representar uma mensagem gerada por $S_{h,j}^{CAN}$. A primeira mensagem tem $q = 0$ e a última tem $q = Q_m - 1$. O tempo de transmissão no pior caso de uma mensagem q é dado pela Equação 4.7. A Equação 4.8 é utilizada para calcular $w_{h,j}^{CAN}$, iniciando com $w_{h,j}^{CAN^0}(q) = B_{h,j}^{CAN} + qC_{h,j}^{CAN}$ e finalizando quando o sistema converge em $w_{h,j}^{CAN^{y+1}}(q) = w_{h,j}^{CAN^y}(q)$, ou finalizando se $J_{h,j}^{CAN} + w_{h,j}^{CAN^{y+1}}(q) - qP_{h,j}^{CAN} + C_{h,j}^{CAN} > D_{h,j}^{CAN}$ (caso em que a mensagem perde seu *deadline*). Para valores de $q > 0$, um valor inicial eficiente é dado por $w_{h,j}^{CAN^0}(q) = w_{h,j}^{CAN}(q-1) + C_{h,j}^{CAN}$.

$$R_{h,j}^{CAN}(q) = J_{h,j}^{CAN} + w_{h,j}^{CAN}(q) - qP_{h,j}^{CAN} + C_{h,j}^{CAN} \tag{4.7}$$

$$w_{h,j}^{CAN^{y+1}}(q) = B_{h,j}^{CAN} + qC_{h,j}^{CAN} + \sum_{\forall x \in hp(S_{h,j}^{CAN})} \left\lceil \frac{w_{h,j}^{CAN^{y}} + J_{x}^{CAN} + \tau_{bit}}{P_{x}^{CAN}} \right\rceil C_{x}^{CAN} \tag{4.8}$$

O *wcrt* de $S_{h,j}^{CAN}$ é o maior dentre todos os *wcrt* das mensagens em Q_m (Equação 4.9).

$$R_{h,j}^{CAN} = \max_{q=0..Q_m-1}(R_{h,j}^{CAN}(q)) \tag{4.9}$$

Um algoritmo para definir atribuições de prioridade para processos de um sistema é apresentado em (BURNS; WELLINGS, 2001). No algoritmo, um procedimento testa se um dado processo é viável com uma determinada prioridade, e pode ser estendido para atribuir prioridades para fluxos de mensagens CAN, se forem utilizadas as equações reproduzidas acima como teste de escalonabilidade.

Como comentado no início desta seção, existem, na literatura, milhares de propostas relacionadas ao CAN. Resumir e analisar todas as propostas relacionadas à análise de escalonamento e tempo de resposta no protocolo CAN é uma tarefa grandiosa e potencialmente desnecessária, devido à quantidade de diferentes abordagens propostas para os mesmos problemas. Por isso, esta seção irá se limitar aos aspectos relacionados ao CAN que apresentados nesta seção. Informações detalhadas sobre o protocolo e suas características podem ser encontradas, entre outros, em (Bosch, 1991; ISO11898-1, 2003).

4.2 MODELO DE REDE

Nas propostas apresentadas nesta Tese, considera-se uma topologia de rede composta por $|\mathbb{N}|$ nodos de rede. O conjunto $\mathbb{N}$ de nodos é definido como

$$\mathbb{N} = \{N_1, N_2, ...N_{|\mathbb{N}|}\}. \tag{4.10}$$

Os nodos de rede são conectados a um ou mais segmentos de rede que utilizam ou o protocolo FlexRay ou o protocolo CAN. Um exemplo da topologia considerada é apresentado na Figura 12.

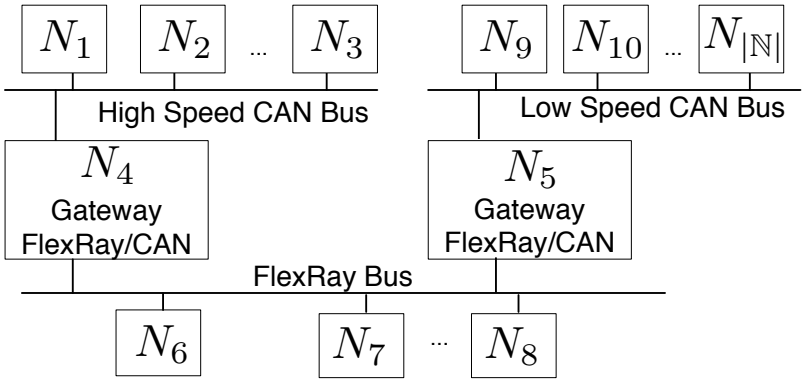

Figura 12 – Exemplo de Topologia de Rede

4.3 NODO DE REDE

Cada nodo de rede N_h, onde $h = 1, 2, ..., |\mathbb{N}|$, é composto por uma unidade central de processamento (*Central Processing Unit*, CPU) e um ou mais Controladores de Comunicação (CC). A CPU e os CCs são interconectados através de uma *interface host*-controlador (*Controller-Host Interface*, CHI). Um CC implementa as funções e serviços necessários para a comunicação no segmento de rede ao qual o nodo está interligado, e seu funcionamento é independente da CPU. Um CC FlexRay segue a especificação FlexRay descrita em (FLEXRAY, 2015) e, de forma similar, um CC CAN segue a especificação do CAN descrita em (Bosch, 1991).

A CPU de um nodo executa uma ou mais tarefas. Uma tarefa pode ser responsável, por exemplo, pelo controle de um atuador ou pela realização dos cálculos de leis de controle. O número de tarefas em um nodo é dado por $|\mathbb{T}_h|$, e $\mathbb{T}_h$ representa o conjunto de tarefas do nodo N_h, dado por:

$$\mathbb{T}_h = \Gamma_{h,1}, \Gamma_{h,2}, ..., \Gamma_{h,|\mathbb{T}_h|}. \tag{4.11}$$

Cada tarefa $\Gamma_{h,i}$ é caracterizada pela tupla $(C^{\Gamma}_{h,i}, P^{\Gamma}_{h,i}, D^{\Gamma}_{h,i})$ onde:

- $h = 1, 2, ..., |\mathbb{N}|$ denota o número do nodo;
- $i = 1, 2, ..., |\mathbb{T}_h|$ denota o número da tarefa no nodo N_h;
- $C^{\Gamma}_{h,i}$ é o tempo de execução da tarefa no pior caso;
- $P^{\Gamma}_{h,i}$ é o intervalo mínimo entre duas ativações consecutivas de $\Gamma_{h,i}$;
- $D^{\Gamma}_{h,i}$ é o *deadline* relativo de $\Gamma_{h,i}$.

A utilização $U(N_h)$ de um nodo N_h é dada por:

$$U(N_h) = \sum_{i=1}^{|\mathbb{T}_h|} \frac{C^{\Gamma}_{h,i}}{P^{\Gamma}_{h,i}}. \tag{4.12}$$

Cada tarefa tem uma sequência infinita de ativações. A k-ésima ativação de uma tarefa $\Gamma_{h,i}$ é representada por $\gamma^{k}_{h,i}$ onde k é o número da ativação da tarefa.

Se uma tarefa envia dados para outro nodo de rede, ela é associada com um ou mais fluxos de mensagens. Os fluxos de mensagens serão formalizados na Seção 4.4.

Nodos conectados a barramentos diferentes podem trocar mensagens entre si. Este trabalho considera que a troca de mensagens entre nodos conectados a barramentos diferentes é gerenciada por um tipo especial de nodo que contém uma classe de tarefas cuja função é receber uma mensagem gerada por um fluxo de mensagens do nodo de origem, extrair o *payload* desta mensagem, empacotar os dados em um quadro no formato necessário para a rede de destino e enviar este quadro para o CC de saída. Tais tarefas são chamadas de tarefas de *gatewaying* e, por conveniência, um nodo que contém tais tarefas é chamado de *gateway*. Um nodo do tipo gateway pode conter uma ou mais tarefas de *gatewaying*. No exemplo ilustrado na Figura 12, os nodos N_4 e N_5 são nodos do tipo *gateway*.

Na Figura 13 é apresentado um exemplo de sistema. Neste sistema, existe um nodo ligado a um barramento FlexRay, um nodo CAN ligado a um barramento CAN e um nodo do tipo *gateway*. Tanto o nodo FlexRay quanto o nodo CAN possuem um conjunto de tarefas e fluxos de mensagens. Já o *gateway* contém tarefas de *gatewaying* para gerenciar possíveis trocas de mensagens entre os nodos ligados aos diferentes barramentos.

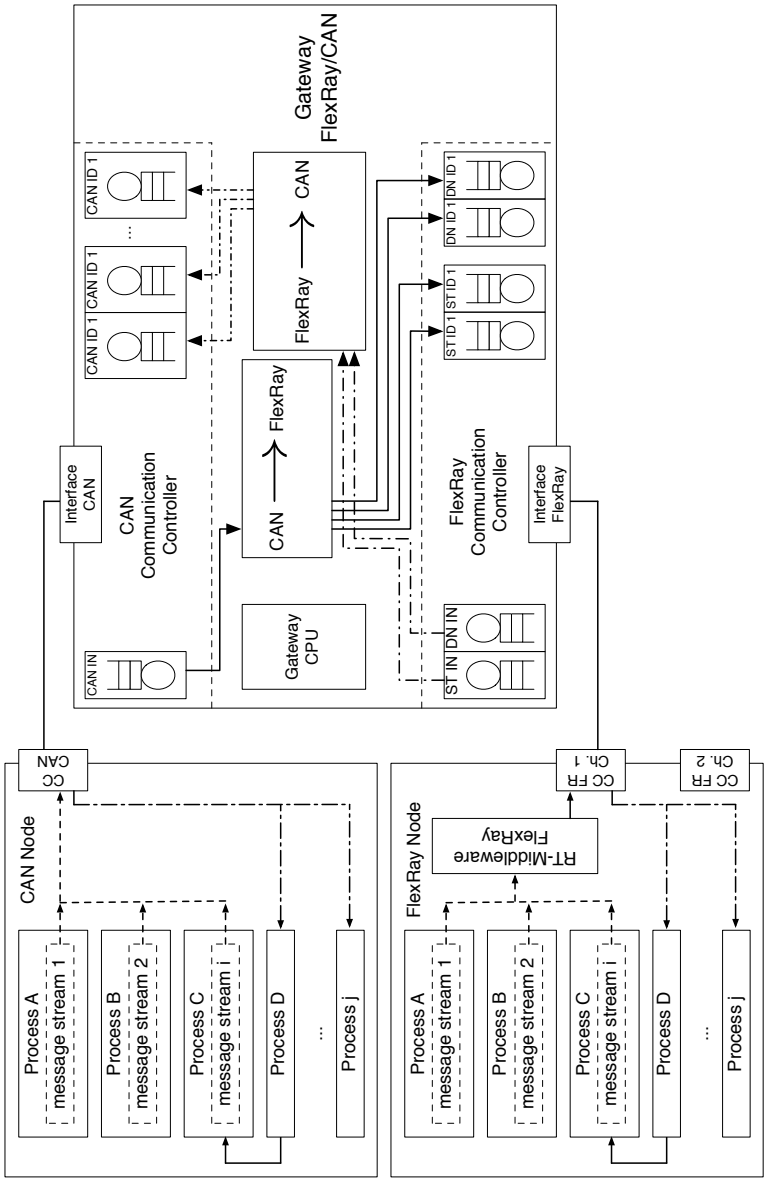

Figura 13 – Detalhe dos nodos de rede e do *gateway*

4.4 FLUXOS DE MENSAGENS

Nesta Tese são considerados sistemas automotivos distribuídos intra-veiculares nos quais a comunicação entre processos é realizada através de trocas de mensagens. Essa troca de mensagens é modelada por um conjunto de fluxos de mensagens. Um fluxo de mensagens pode estar associado com o segmento Estático de uma rede FlexRay, com o segmento Dinâmico de uma rede FlexRay ou com uma rede CAN.

O número de fluxos de mensagens de um nodo H_h que transmitem no ST é dado por $|\mathbb{S}_h^{ST}|$. O conjunto dos fluxos de mensagens do nodo H_h que transmitem no ST é representado por $\mathbb{S}_h^{ST}$:

$$\mathbb{S}_h^{ST} = S_{h,1}^{ST}, S_{h,2}^{ST}, ..., S_{h,|\mathbb{S}_h^{ST}|}^{ST}. \tag{4.13}$$

Cada fluxo de mensagem $S_{h,j}^{ST}$ em $\mathbb{S}_h^{ST}$ é caracterizado por uma tupla $(C_{h,j}^{ST}, P_{h,j}^{ST}, D_{h,j}^{ST})$ onde:

- $h = 1, 2, ..., |\mathbb{N}|$ denota o número do nodo;
- $j = 1, 2, ..., |\mathbb{S}_h^{ST}|$ é o identificador do fluxo de mensagens no nodo;
- $C_{h,j}^{ST}$ é o tempo máximo requerido para transmitir uma mensagem gerada pelo fluxo $S_{h,j}^{ST}$. É considerado que $C_{h,j}^{ST}$ inclui tanto o *payload* quanto o cabeçalho e cauda necessários para o quadro que encapsula a mensagem. $C_{h,j}^{ST}$ é calculado utilizando-se as equações apresentadas no Capítulo 2, Seção 2.1.1.3, página 50;
- $P_{h,j}^{ST}$ é o tempo mínimo entre duas gerações consecutivas de mensagens pelo fluxo;
- $D_{h,j}^{ST}$ é o *deadline* relativo do fluxo.

Um fluxo de mensagens $S_{h,j}^{ST}$ gera uma sequência infinita de mensagens. A l-ésima mensagem gerada por $S_{h,j}^{ST}$ é representada por $m_{h,j}^{ST,l}$ $(l = 1, 2, ..., +\infty)$. Uma mensagem herda as características temporais do fluxo que a gerou.

O conjunto $\mathbb{S}^{ST}$ de fluxos de mensagens que utilizam o ST do FlexRay em um sistema é denotado por:

$$\mathbb{S}^{ST} = \{\mathbb{S}_1^{ST} \cup \mathbb{S}_2^{ST} \cup ... \cup \mathbb{S}_{|\mathbb{N}|}^{ST}\}. \tag{4.14}$$

De forma semelhante ao ST, o número de fluxos de mensagens de um nodo H_h que transmitem no DN do FlexRay é dado por $|\mathbb{S}_h^{DN}|$.

O conjunto dos fluxos de mensagens do nodo H_h que transmitem no DN é representado por $\mathbb{S}_h^{DN}$:

$$\mathbb{S}_h^{DN} = S_{h,1}^{DN}, S_{h,2}^{DN}, ..., S_{h,|\mathbb{S}_h^{DN}|}^{DN}. \tag{4.15}$$

Um fluxo de mensagem periódico ou esporádico $S_{h,j}^{DN}$ em $\mathbb{S}_h^{DN}$ é caracterizado por uma tupla $(C_{h,j}^{DN}, P_{h,j}^{DN}, D_{h,j}^{DN})$ onde:

- $h = 1, 2, ..., |\mathbb{N}|$ é o número do nodo ao qual o fluxo de mensagens pertence;
- $j = 1, 2, ..., |\mathbb{S}_h^{DN}|$ é o identificador do fluxo de mensagens no nodo;
- $C_{h,j}^{DN}$ é o tempo máximo requerido para transmitir uma mensagem gerada pelo fluxo $S_{h,j}^{DN}$. É considerado que $C_{h,j}^{DN}$ inclui tanto o *payload* quanto o cabeçalho e cauda necessários para o quadro que encapsula a mensagem. $C_{h,j}^{DN}$ é calculado utilizando-se das equações apresentadas no Capítulo 2, Seção 2.1.1.3, página 50;
- $P_{h,j}^{DN}$ é o tempo mínimo entre duas gerações consecutivas de mensagens pelo fluxo;
- $D_{h,j}^{DN}$ é o *deadline* relativo do fluxo.

Considera-se que um fluxo de mensagens aperiódico $S_{h,j}^{DN}$ não possui um intervalo mínimo entre chegadas, e pode ou não estar associado a um *deadline* relativo $D_{h,j}^{DN}$.

Um fluxo de mensagens $S_{h,j}^{DN}$ gera uma sequência infinita de mensagens. A l-ésima mensagem gerada por $S_{h,j}^{DN}$ é representada por $m_{h,j}^{DN,l}$ $(l = 1, 2, ..., +\infty)$. Uma mensagem herda as características temporais do fluxo que a gerou.

O conjunto $\mathbb{S}^{DN}$ de fluxos de mensagens que utilizam o DN do FlexRay em um sistema é denotado por:

$$\mathbb{S}^{DN} = \{\mathbb{S}_1^{DN} \cup \mathbb{S}_2^{DN} \cup ... \cup \mathbb{S}_{|\mathbb{N}|}^{DN}\}. \tag{4.16}$$

Por fim, o número de fluxos de mensagens de um nodo H_h que transmitem em um barramento CAN é dado por $|\mathbb{S}_h^{CAN}|$. O conjunto dos fluxos de mensagens de H_h que transmitem em um barramento CAN é representado por $\mathbb{S}_h^{CAN}$:

$$\mathbb{S}_h^{CAN} = S_{h,1}^{CAN}, S_{h,2}^{CAN}, ..., S_{h,|\mathbb{S}_h^{CAN}|}^{DN}. \tag{4.17}$$

Um fluxo de mensagem periódico ou esporádico $S_{h,j}^{CAN}$ em $\mathbb{S}_h^{CAN}$ é caracterizado por uma tupla $(C_{h,j}^{CAN}, P_{h,j}^{CAN}, D_{h,j}^{CAN})$ onde:

- $h = 1, 2, ..., |\mathbb{N}|$ é o número do nodo ou do gateway ao qual o fluxo de mensagens pertence;
- $j = 1, 2, ..., |\mathbb{S}_h^{CAN}|$ é o identificador do fluxo de mensagens no nodo;
- $C_{h,j}^{CAN}$ é o tempo máximo requerido para transmitir uma mensagem gerada pelo fluxo $S_{h,j}^{CAN}$. É considerado que $C_{h,j}^{CAN}$ inclui tanto o *payload* quanto o cabeçalho e cauda necessários para o quadro que encapsula a mensagem. $C_{h,j}^{CAN}$ é calculado utilizando-se das equações apresentadas na Seção 4.1;
- $P_{h,j}^{CAN}$ é o tempo mínimo entre duas gerações consecutivas de mensagens pelo fluxo;
- $D_{h,j}^{CAN}$ é o *deadline* relativo do fluxo.

Um fluxo de mensagens $S_{h,j}^{CAN}$ gera uma sequência infinita de mensagens. A l-ésima mensagem gerada por $S_{h,j}^{CAN}$ é representada por $m_{h,j}^{CAN,l}$ $(l = 1, 2, ..., +\infty)$. Uma mensagem herda as características temporais do fluxo que a gerou.

O conjunto $\mathbb{S}^{CAN}$ de fluxos de mensagens que utilizam o protocolo CAN em um sistema é denotado por:

$$\mathbb{S}^{CAN} = \{\mathbb{S}_1^{CAN} \cup \mathbb{S}_2^{CAN} \cup ... \cup \mathbb{S}_{|\mathbb{N}|}^{CAN}\}. \tag{4.18}$$

O conjunto $\mathbb{S}$ de todos os fluxos de mensagens de um sistema é denotado por

$$\mathbb{S} = \mathbb{S}^{ST} \cup \mathbb{S}^{DN} \cup \mathbb{S}^{CAN}. \tag{4.19}$$

4.5 ATIVIDADE

Para esta Tese, interessam os sistemas intra-veiculares nos quais tarefas que executam funções específicas trocam dados com outras tarefas que executam funções distintas. Um exemplo de tal sistema é o caso onde uma tarefa A, que é responsável por ler dados de um sensor, envia dados para outra tarefa B através de uma rede CAN. A tarefa B realiza os cálculos referentes a uma malha de controle e envia o resultado deste cálculo, através de uma rede FlexRay, para uma tarefa C que, por sua vez, controla um atuador.

Tais sistemas possuem uma tarefa de origem e uma tarefa de destino. Neste trabalho chamamos o caminho entre uma tarefa de origem e uma tarefa de destino de **atividade**. Seguindo a notação proposta em (MARINCA et al., 2004; ZENG et al., 2011), uma atividade é representada como um grafo direcional onde os vértices são as tarefas e as arestas são os fluxos de mensagens que realizam a comunicação entre duas tarefas consecutivas da atividade. Um exemplo de atividade com o respectivo grafo que a representa é ilustrado na Figura 14.

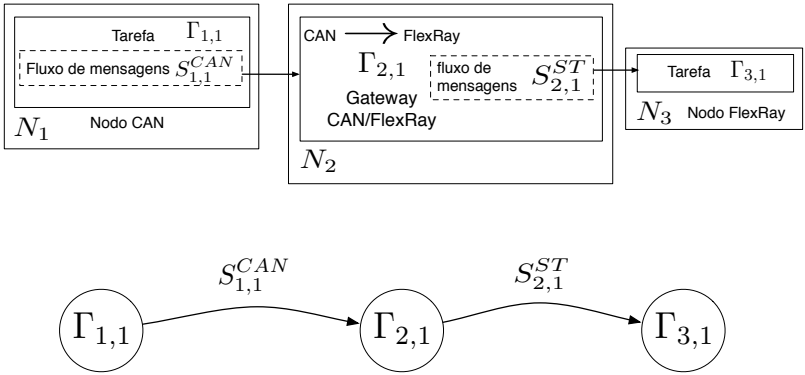

Figura 14 – Exemplo de representação de atividade

Considera-se que um sistema pode ser dividido em um número $|c|$ de atividades. Considera-se que, de forma complementar à representação em forma de grafo, uma atividade também pode ser representada como um conjunto φ_c, onde $c = 1, 2, ..., |c|$ é o número da atividade. Os elementos de φ_c são ordenados de acordo com a sua ordem dentro da atividade. Por exemplo, o conjunto φ_c que representa o grafo ilustrado na Figura 14 é definido como

$$\varphi_c = \{\Gamma_{1,1}, S^{CAN}_{1,1}, \Gamma_{2,1}, S^{ST}_{2,1}, \Gamma_{3,1}\}. \tag{4.20}$$

Para uma atividade φ_c, $\mathbb{T}^{\varphi}_c$ representa o conjunto de tarefas desta atividade. O número de tarefas em $\mathbb{T}^{\varphi}_c$ é dado por $|\mathbb{T}^{\varphi}_c|$. Uma atividade φ_c deve conter ao menos uma tarefa.

O conjunto de fluxos de mensagens na atividade φ_c é representado por $\mathbb{S}^{\varphi}_c$. O número de fluxos de mensagens na atividade φ_c é dado por $|\mathbb{S}^{\varphi}_c|$.

O número total de elementos em φ_c é dado por $|\varphi_c|$:

$$|\varphi_c| = |\mathbb{T}_c^{\varphi}| + |\mathbb{S}_c^{\varphi}|. \tag{4.21}$$

Cada atividade φ_c possui um requisito temporal fim-a-fim D_c^{φ}. Neste trabalho não será discutido como é feita a definição de D_c^{φ}.

Por fim, considera-se que a conexão entre duas tarefas de um mesmo nodo não possui custo, sendo o tempo de computação necessário para a troca de dados entre as mesmas considerado como parte do tempo de execução da tarefa de origem.

4.6 UM EXEMPLO TÍPICO

Nesta seção será apresentado um exemplo que ilustra como um sistema pode ser mapeado para um conjunto de atividades.

Infelizmente é difícil obter exemplos de sistemas baseados em veículos reais devido a restrições comerciais e industriais. Por este motivo, este exemplo é baseado nos exemplos para redes FlexRay e CAN que estão incluídos nos *softwares* CANoe e CANalyzer da Vector (VECTOR, 2012).

O sistema exemplo é ilustrado na Figura 15. Este sistema contém um conjunto de nodos e tarefas que estão listados na Tabela 1. O sistema também contém dois barramentos de rede, um utilizando o protocolo CAN e outro utilizando FlexRay. A interconexão entre estes barramentos é realizada por um nodo *gateway* que contém uma tarefa de *gatewaying* cuja função é receber uma mensagem vinda do barramento CAN, extrair os dados do *payload*, encapsular estes dados em um quadro FlexRay e enviar este quadro no barramento FlexRay.

As tarefas alocadas em um nodo podem interagir com tarefas alocadas em outros nodos através de mensagens trocadas via rede. Os relacionamentos entre tarefas alocadas em diferentes nodos e o fluxo de mensagens que gerencia a troca de dados entre estas tarefas são listados na Tabela 2. Note que quando a troca de dados é realizada através do nodo *gateway*, o relacionamento listado na Tabela 2 apresenta o caminho através do mesmo.

Com base nos relacionamentos apresentados na Tabela 2, as atividades correspondentes são construídas seguindo a explicação apresentada na Seção 4.5. Os grafos relativos às atividades resultantes são ilustrados nas Figuras 16 até 26.

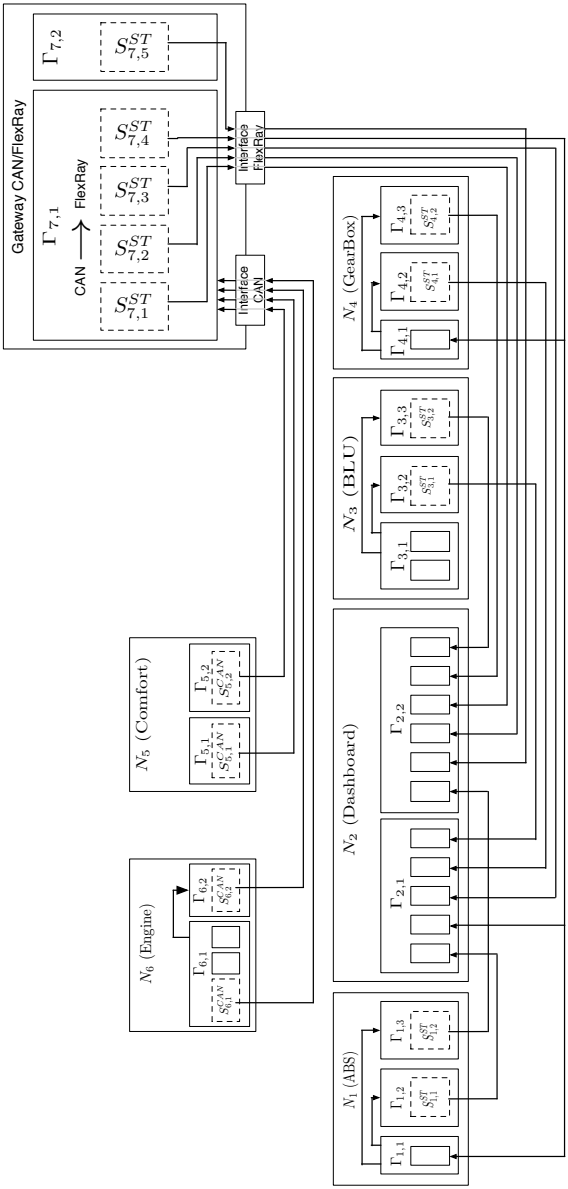

Figura 15 – Um exemplo típico

Nodo	Tarefa	Descrição da tarefa
N_1 (ABS)	$\Gamma_{1,1}$	Controle do ABS
	$\Gamma_{1,2}$	Diagnóstico
	$\Gamma_{1,3}$	*Status*
N_2 (*Dashboard*)	$\Gamma_{2,1}$	*Dashboard*
	$\Gamma_{2,2}$	Diagnóstico
N_3 (BLU)	$\Gamma_{3,1}$	*Controle das luzes traseiras*
	$\Gamma_{3,2}$	Diagnóstico
N_4 (*Gearbox*)	$\Gamma_{4,1}$	Controle do sistema de marchas
	$\Gamma_{4,2}$	Diagnóstico
N_5 (*Comfort*)	$\Gamma_{5,1}$	Controle do sistema de conforto
	$\Gamma_{5,2}$	Diagnóstico
N_6 (*Engine*)	$\Gamma_{6,1}$	Controle do Motor
	$\Gamma_{6,2}$	Diagnóstico
N_7 (*gateway*)	$\Gamma_{7,1}$	Conversão do CAN para o FlexRay
	$\Gamma_{7,2}$	Diagnóstico

Tabela 1 – Nodos e respectivas tarefas

Tarefa de origem	Tarefa de Destino	Caminho
$\Gamma_{1,2}$	$\Gamma_{2,1}$	$S^{ST}_{1,1}$
$\Gamma_{1,3}$	$\Gamma_{2,2}$	$S^{ST}_{1,2}$
$\Gamma_{3,2}$	$\Gamma_{2,1}$	$S^{ST}_{3,1}$
$\Gamma_{3,3}$	$\Gamma_{2,2}$	$S^{ST}_{3,2}$
$\Gamma_{4,2}$	$\Gamma_{2,1}$	$S^{ST}_{4,1}$
$\Gamma_{4,3}$	$\Gamma_{2,2}$	$S^{ST}_{4,2}$
$\Gamma_{7,2}$	$\Gamma_{2,2}$	$S^{ST}_{7,5}$
$\Gamma_{5,1}$	$\Gamma_{2,1}$	$S^{CAN}_{5,1} \rightarrow \Gamma_{7,1} \rightarrow S^{ST}_{7,3}$
$\Gamma_{5,2}$	$\Gamma_{2,2}$	$S^{CAN}_{5,2} \rightarrow \Gamma_{7,1} \rightarrow S^{ST}_{7,1}$
$\Gamma_{6,1}$	$\Gamma_{2,1}$	$S^{CAN}_{6,1} \rightarrow \Gamma_{7,1} \rightarrow S^{ST}_{7,4}$
$\Gamma_{6,2}$	$\Gamma_{2,2}$	$S^{CAN}_{6,2} \rightarrow \Gamma_{7,1} \rightarrow S^{ST}_{7,2}$

Tabela 2 – Relação entre tarefas

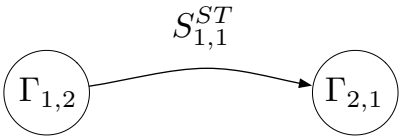

Figura 16 – Atividade φ_1

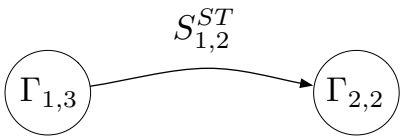

Figura 17 – Atividade φ_2

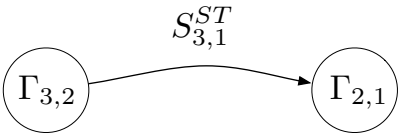

Figura 18 – Atividade φ_3

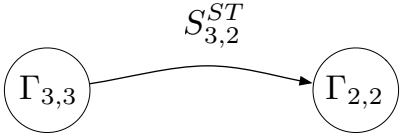

Figura 19 – Atividade φ_4

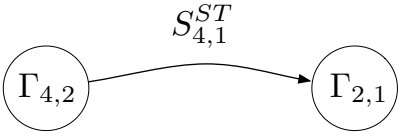

Figura 20 – Atividade φ_5

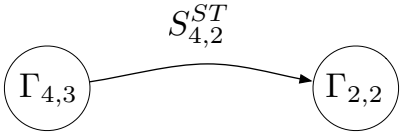

Figura 21 – Atividade φ_6

Figura 22 – Atividade φ_7

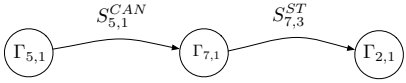

Figura 23 – Atividade φ_8

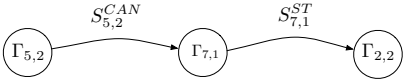

Figura 24 – Atividade φ_9

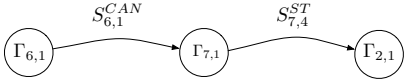

Figura 25 – Atividade φ_{10}

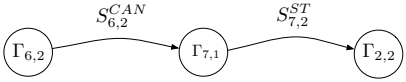

Figura 26 – Atividade φ_{11}

4.7 SUMÁRIO

Neste capítulo foi descrito o modelo de sistema que será utilizado na propostas que serão apresentadas nesta Tese, sendo também estabelecidas as notações que serão utilizadas nos demais capítulos deste documento.

Foram formalizadas as notações para nodos de rede, fluxos de mensagens e tarefas. Também foi apresentado o conceito de atividade, utilizado para representar o caminho entre uma tarefa de origem e uma tarefa de destino.

O uso do modelo de sistema apresentado foi ilustrado através de um exemplo completo.

5 CONTRIBUIÇÕES RELACIONADAS AO SEGMENTO ESTÁTICO DO FLEXRAY

Neste capítulo serão apresentadas contribuições desta Tese para o estado da arte relacionado ao Segmento Estático do FlexRay.

O Segmento Estático (ST) é projetado para transmitir mensagens geradas periodicamente por fluxos, ou seja, o ST é o componente *time-triggered* do FlexRay. Seu projeto tem um papel importante no projeto do sistema como um todo. É uma tarefa que envolve, entre outros, a definição do número de *slots* estáticos, do tamanho do ST e do número de *slots* alocados a cada nodo. Para que seja realizado um projeto adequado do ST, deve-se escolher um modelo de escalonamento que atenda às exigências do sistema que está sendo considerado.

Zeng et al. (2011) definem um modelo de escalonamento síncrono (*Synchronous Scheduling model*), onde a execução das tarefas do sistema está sincronizadas com o Ciclo FlexRay de tal forma que uma tarefa deve terminar antes do início do *slot* que transmite o sinal de saída daquela tarefa. No Capítulo 3 deste documento foi apresentado um levantamento de trabalhos considerados relevantes para esta tese. Como salientado naquele capítulo, uma característica comum a muitos dos trabalhos existentes na literatura é a adoção de um modelo de escalonamento síncrono, considerando que existe um forte sincronismo entre tarefas, sinais ou fluxos de mensagens e o Ciclo FlexRay, e que os períodos dos fluxos de mensagens que encapsulam os sinais em mensagens são múltiplos inteiros de FC_{bus}, impondo, deste modo, uma sincronização geral no sistema. Mas segundo Zeng et al. (2011), abordagens que considerem um escalonamento assíncrono, isto é, um modelo de escalonamento onde não existem sincronizações entre o Ciclo FlexRay e a geração de mensagens ou a ativação de uma tarefa, podem ser de alto interesse prático quando considerada a migração de sistemas CAN para o FlexRay.

Outra característica comum a muitos dos trabalhos da literatura é que muitos apresentam propostas onde os fluxos de mensagens são associados com *slots* estáticos e repetições de ciclos exclusivos, sendo que essa associação em geral é definida utilizando-se alguma técnica de otimização como, por exemplo, ILP. Entretanto, a especificação do FlexRay define apenas que a associação deve ser exclusiva entre *slots* estáticos e nodos do sistema.

A seguir será apresentada uma abordagem diferente para o projeto do segmento Estático do FlexRay. Na abordagem proposta é re-

movida a forte sincronização entre tarefas e sinais no nível de camada de aplicação e o ciclo FlexRay, usualmente assumida pelos trabalhos da literatura. Ao se relaxar esta restrição, torna-se possível escalonar, no ST, as mensagens associadas a aplicações distribuídas cujos períodos não são múltiplos inteiros de FC_{bus}. Como consequência, o número de sistemas que são escalonáveis pode ser melhorado significativamente.

A abordagem é composta por dois métodos de alocação de *slots* estáticos que permitem a implementação, no ST, de um modelo de escalonamento assíncrono (*Asynchronous Scheduling model*), onde o número de *slots* estáticos alocados para um nodo está diretamente relacionado aos requisitos temporais dos fluxos de mensagens deste nodo. Basicamente, o número de *slots* estáticos alocados para um nodo é definido utilizando técnicas de análise de tempo de resposta (*Response Time Analysis*, RTA) tradicionais (AUDSLEY et al., 1993).

O primeiro método, chamado Método de Alocação Proporcional, considera que existe em cada ciclo FlexRay um instante de tempo chamado *freeze instant* onde um *middleware* de tempo real implementado em cada nodo N_h define quais mensagens geradas pelos fluxos de mensagens de N_h serão enviadas para a rede utilizando os *slots* estáticos alocados para N_h. Já o segundo método, chamado Método de Alocação Proporcional Adaptativo é uma extensão do primeiro método, e considera que existe um *freeze instant* por *slot* estático, em vez de um único *freeze instant* por FC (como ocorre no primeiro método).

No restante deste capítulo serão apresentados os modelos de sistema e de rede considerados na proposta (Seção 5.1), seguidos da apresentação do Método de Alocação Proporcional (Seção 5.2) e do Método de Alocação Proporcional Adaptativo (Seção 5.3). A apresentação de cada método inclui a descrição de um componente de software necessário para o mesmo, a descrição de um método de alocação de *slots* estáticos e um exemplo de uso completo. No caso do Método de Alocação Proporcional, é explicado como o mesmo pode ser integrado com o padrão AUTOSAR (AUTOSAR, 2012). Já a apresentação do Método de Alocação Proporcional Adaptativo inclui comparação com uma das técnicas do estado da arte sobre escalonamento no ST. Por fim, na Seção 5.4 serão apresentadas as considerações finais sobre o capítulo.

A proposta apresentada neste capítulo é original, mas versões preliminares foram publicadas em (LANGE et al., 2012) e (LANGE et al., 2015).

5.1 MODELO DE SISTEMA E REDE

As propostas apresentadas neste capítulo utilizam o modelo de sistema apresentado no Capítulo 4. Para facilitar o entendimento do texto, os pontos daquele capítulo que são relevantes para esta proposta são resumidos a seguir.

São considerados sistemas automotivos distribuídos compostos por $|\mathbb{N}|$ nodos FlexRay interconectados por um canal de comunicação FlexRay (Figura 27). A exemplo dos demais trabalhos da literatura sobre o FlexRay (discutidos no Capítulo 3), esta proposta não considera o uso do segundo canal de comunicação, mas este poderia ser utilizado para fins de redundância na rede.

Cada nodo FlexRay é composto por uma unidade central de processamento (CPU) e um controlador de comunicação (CC). A CPU e o CC são interconectados por uma *interface host*-controlador (CHI). O CC trabalha de forma independente da CPU, e implementa os serviços de comunicação FlexRay. O CHI segue a especificação do FlexRay descrita em (FLEXRAY, 2015).

Nos sistemas considerados, a comunicação é realizada através da troca de mensagens. Esta troca de mensagens é modelada por um conjunto $mathbbS^{ST}$ composto por $|\mathbb{S}^{ST}|$ fluxos de mensagens síncronos que transmitem no segmento Estático de uma rede FlexRay.

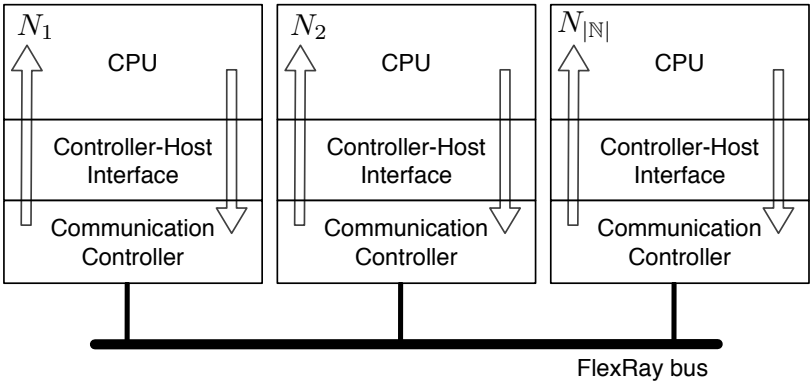

Figura 27 – Topologia de Rede

Cada nodo N_h possui um número $|\mathbb{S}_h^{ST}|$ de fluxos de mensagens síncronos. O conjunto de fluxos de mensagens síncronos do nodo H_h é representado por $\mathbb{S}_h^{ST}$.

Cada fluxo de mensagem $S_{h,j}^{ST}$ em $\mathbb{S}^{ST}$ é caracterizado por uma tupla $(C_{h,j}^{ST}, P_{h,j}^{ST}, D_{h,j}^{ST})$ onde:

- $h = 1, 2, ..., |\mathbb{N}|$ denota o número do nodo;
- $j = 1, 2, ..., |\mathbb{S}_h^{ST}|$ é o identificador do fluxo de mensagens no nodo;
- $C_{h,j}^{ST}$ é o tempo máximo requerido para transmitir uma mensagem gerada pelo fluxo $S_{h,j}^{ST}$. É considerado que $C_{h,j}^{ST}$ inclui tanto o *payload* quanto o cabeçalho e cauda necessários para o quadro que encapsula a mensagem. $C_{h,j}^{ST}$ é calculado utilizando-se das equações apresentadas no Capítulo 2, Seção 2.1.1.3, página 50;
- $P_{h,j}^{ST}$ é o tempo mínimo entre duas gerações consecutivas de mensagens pelo fluxo;
- $D_{h,j}^{ST}$ é o *deadline* relativo do fluxo.

Um fluxo de mensagens gera uma sequência infinita de mensagens. A l-ésima mensagem gerada por um fluxo $S_{h,j}^{ST}$ é representada por $m_{h,j}^{ST,l}$ $(l = 1, 2, ..., +\infty)$. Uma mensagem herda as características temporais do fluxo que a gerou.

Para cada nodo FlexRay N_h $(h = 1, 2, ..., |\mathbb{N}|)$ é alocado um número inteiro H_h de *slots* estáticos. O Método de Alocação Proporcional (*Proportional Allocation Scheme*) proposto na Seção 5.2.3 tornará possível a definição do número H_h de *slots* estáticos a serem alocados para cada nodo de acordo com os requisitos de comunicação deste nodo.

Como descrito no Capítulo 2, Seção 2.1.1.2, página 40, o ST é dividido em *slots* de tamanho fixo e idêntico *gdStaticSlot*, sendo que este tamanho é definido durante a fase de projeto do sistema. Nesta proposta, não será abordado o problema de agrupar sinais em quadros (*signal packing*). Assume-se simplesmente que os sinais foram agrupados em quadros utilizando-se uma das técnicas propostas na literatura (por exemplo, a técnica proposta em (GRENIER et al., 2008)). Como *gdStaticSlot* tem o mesmo tamanho para todos os *slots* estáticos, este deve ser definido de forma a acomodar inclusive a maior mensagem do sistema:

$$gdStaticSlot = C_{max}^{ST} \tag{5.1}$$

onde C_{max}^{ST} é o tempo [máximo] de transmissão da maior mensagem gerada por um fluxo em $\mathbb{S}^{ST}$.

Sem perda da generalização, neste trabalho considera-se que o tamanho da maior mensagem é unitário, isto é:

$$C_{max}^{ST} = 1 \tag{5.2}$$

e consequentemente

$$gdStaticSlot = 1. \tag{5.3}$$

No restante desta proposta, assume-se que todos os parâmetros do sistema, incluindo FC_{bus}, ST_{bus} e qualquer outro parâmetro relacionado aos fluxos de mensagens, são dados em inteiros de *slots* estáticos. Esta suposição está de acordo com outras propostas da literatura como (POP et al., 2008) e (GRENIER et al., 2008).

Assume-se, também, que a geração das mensagens na camada de aplicações não está sincronizada com o FC.

Finalmente, considera-se que $hp(S^{ST}_{h,j})$ e $lp(S^{ST}_{h,j})$ são os conjuntos de fluxos com prioridade, respectivamente, maior e menor que o fluxo $S^{ST}_{h,j}$ do nodo h:

$$hp(S^{ST}_{h,j}) = \{S^{ST}_{h,j+1}, S^{ST}_{h,j+2}, ..., S^{ST}_{h,|\mathbb{S}^{ST}_h|}\} \tag{5.4}$$

$$lp(S^{ST}_{h,j}) = \{S_{S^{ST}_{h,1}}, S^{ST}_{h,2}, ..., S^{ST}_{h,j-1}\}. \tag{5.5}$$

5.2 MÉTODO DE ALOCAÇÃO PROPORCIONAL

Nesta seção é apresentado o Método de Alocação Proporcional de *Slots* Estáticos.

No método proposto, é removida a forte sincronização entre tarefas e sinais no nível de camada de aplicação e o ciclo FlexRay usualmente assumida pelos trabalhos da literatura. Em vez de associar fluxos de mensagens com identificadores de *slots* estáticos únicos, propõe-se considerar um modelo de escalonamento assíncrono, isto é, um modelo de escalonamento onde não existem sincronizações entre o FC e a geração de mensagens ou a ativação de tarefas, e apenas alocar *slots* para os nodos, utilizando posteriormente um *middleware* de tempo real específico (implementado na CPU de cada nodo) para arbitrar, durante a execução do sistema, quais mensagens serão enviadas para o barramento durante o ciclo FlexRay atual. A definição de quais mensagens serão enviadas é realizada em um instante de tempo chamado *freeze instant*, que é sincronizado com o ciclo FlexRay.

A proposta utiliza técnicas de análise de tempo de resposta (*Response Time Analysis*, RTA) tradicionais (AUDSLEY et al., 1993) para definir, com base nos requisitos temporais dos fluxos de mensagens de um nodo, quantos *slots* estáticos serão alocados para este nodo.

No restante desta seção será apresentado o *middleware* de tempo

real considerado, seguido da proposta de um método para a alocação de *slots* estáticos e de um exemplo ilustrativo (Seções 5.2.2 a 5.2.4). Por fim, na Seção 5.2.5 é apresentada uma proposta para a integração do método proposto com o padrão AUTOSAR (AUTOSAR, 2012).

5.2.1 RT-Middleware

Nesta proposta, ao invés de fixar a associação entre os fluxos de mensagens e os *slots* estáticos, será considerado que os *slots* estáticos são apenas alocados para os nodos e, em cada nodo, um *middleware* de tempo real específico irá arbitrar em tempo de execução qual mensagem será transmitida em cada *slot* de um FC. Aqui, o *middleware* é denominado *RT-Middleware*.

O *RT-Middleware* é implementado na CPU de cada nodo, e possui um *dispatcher* e um *buffer* para cada fluxo de mensagens. Considera-se que o *dispatcher* trabalha com prioridades fixas para o escalonamento da transmissão das mensagens (*Fixed-Priority Scheduling*). Uma mensagem gerada na camada de aplicação é colocada no *buffer* associado ao seu fluxo, e aguarda até ser despachada.

Posteriormente, em um instante chamado *instante de freeze* (ou *freezing instant*) f_h^g, sendo $h = 1, 2, ..., n$ e o índice g denotando o g-ésimo *freeze instant*, o *dispatcher* remove H_h mensagens dos *buffers* locais e as move para o *buffer* de entrada do CHI. O *freeze instant* é repetido periodicamente e está sincronizado com o FC. O *freeze instant* é definido para cada nodo h durante a fase de projeto, e o intervalo de tempo δ_{f_h} entre o *freeze instant* de um nodo N_h e o início do primeiro *slot* estático associado a este nodo deve ser suficiente para acomodar o processo de *dispatching*, incluindo a geração dos quadros FlexRay a serem transmitidos. Assume-se aqui que existe apenas um *freeze instant* por nodo, e o mesmo ocorre sempre no mesmo ponto do FC. Portanto, o intervalo de tempo Δ_{f_h} entre dois *freeze instants* consecutivos é $\Delta_{f_h} = FC_{bus}$.

Nesta proposta, define-se que o *dispatcher* seleciona as mensagens a serem transmitidas de acordo com uma estratégia baseada em escalonamento *Rate Monotonic* (RM): quanto menor o período do fluxo de mensagens, maior sua prioridade (BURNS; WELLINGS, 2001).

Os princípios básicos do *RT-Middleware* proposto são ilustrados na Figura 28. Na figura 29 é ilustrado um exemplo de FC com os instantes de *freeze* f_h^g e f_h^{g+1}, e o intervalo de tempo Δ_{f_g} decorrido entre eles.

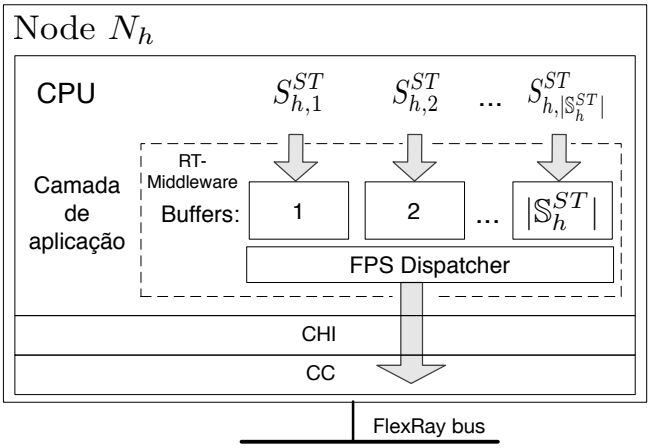

Figura 28 – *Middleware* de Tempo Real

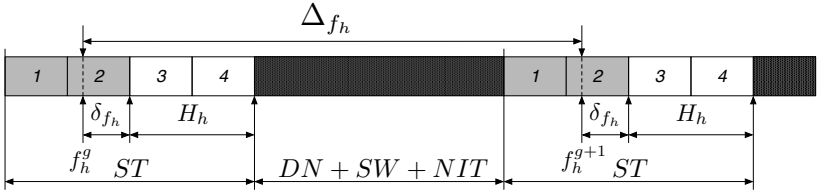

Figura 29 – *Freeze instants* f^g_h e f^{g+1}_h e intervalo de tempo Δ_{f_h}

A utilização deste tipo de *middleware* requer a alocação prévia de um número adequado (H_h) de *slots* estáticos para o nodo N_h. Na próxima Seção será proposto e discutido um método para definir esta alocação.

5.2.2 Descrição do Método de Alocação Proporcional

O método de alocação de *slots* estáticos desempenha um papel importante na garantia dos *deadlines* das mensagens periódicas. Nesta seção será apresentada formalmente a definição de método de alocação de *slots*, sendo discutidos também seus requisitos e métricas de desempenho.

Definimos um *método de alocação de slots estáticos* como um

algoritmo que produz como saída o número de *slots* estáticos que devem ser alocados para um nodo FlexRay para que os *deadlines* das mensagens geradas por seus fluxos sejam respeitados. A entrada deste algoritmo são os parâmetros do sistema e o conjunto de fluxos de mensagens.

Um método de alocação específico será proposto e analisado na Seção 5.2.3. A seguir os requisitos que qualquer método de alocação deve satisfazer serão apresentados e discutidos. Esses requisitos podem ser modelados por uma *restrição de protocolo* (*protocol constraint*) e por uma *restrição de deadline* (*deadline constraint*), como sugerido anteriormente em (AGRAWAL et al., 1994; MALCOLM, 1994) para resolver o problema de alocação de banda para o Timed Token Protocol (GROW, 1984). O Timed Token Protocol é o fundamento de diversos protocolos de comunicação para tempo real que utilizam uma arbitragem baseada na troca de *tokens* como, por exemplo, o PROFIBUS Industrial Communication Protocol (TOVAR; VASQUES, 1998, 1999).

5.2.2.1 Restrição de Protocolo

De acordo com a especificação do FlexRay, o ST é composto por um número inteiro $gNumberOfStaticSlots$ de *slots* estáticos. E como já definido, nesta proposta assume-se que todos os parâmetros do sistema são dados em múltiplos inteiros de *slots* estáticos, portanto o tamanho ST_{bus} do ST é dado por:

$$ST_{bus} = gNumberOfStaticSlots. \tag{5.6}$$

A especificação do FlexRay define que cada nodo do sistema deve estar associado a, pelo menos, um *slot* estático, e que o número de *slots* estáticos é limitado a 1023, que é o maior identificador ($FrameID$) que pode ser atribuído para um *slot*.

Schmidt et al. (2010) sugere que existe uma dependência cíclica entre o tamanho do DN e do ST. Nesta proposta, considera-se que ST_{bus} tem o tamanho necessário para acomodar os *slots* estáticos alocados para cada nodo somado a uma possível reserva para expansões, ficando o restante do ciclo disponível para o DN e para os segmentos de controle. Desta forma,

$$ST_{bus} = \sum_{h=1}^{|\mathbb{N}|} H_h + \rho,\ \rho \geq 0 \tag{5.7}$$

onde H_h é a alocação de *slots* estáticos para o nodo N_h e ρ representa

a possível alocação para futuras expansões.

O FC deve ter um tamanho FC_{bus} tal que permita que todas as mensagens periódicas cumpram seus *deadlines*. Isso significa que a definição de FC_{bus} deve considerar as características do conjunto de fluxo de mensagens $\mathbb{S}^{ST}$, como exemplificado a seguir.

Considere um sistema onde $S_{h,j}^{ST}$ é o fluxo de mensagem j que pertence ao nodo N_h. $S_{h,j}^{ST}$ tem um período $P_{h,j}^{ST}$, um *deadline* $D_{h,j}^{ST}$ que é igual ao seu período $P_{h,j}^{ST}$ e um tamanho $C_{h,j}^{ST}$ que é unitário e igual ao tamanho de um *slot* estático. $S_{h,j}^{ST}$ está associado ao *slot* estático com $FrameID = 1$. A chegada da $l-$ésima mensagem gerada por $S_{h,j}^{ST}$ é representada por $m_{h,j}^{ST,l}$ $(l = 1, 2, ..., +\infty)$. $m_{h,j}^{ST,l}$ herda o *deadline* de $S_{h,j}^{ST}$.

Considere que a mensagem $m_{h,j}^{ST,1}$ chega imediatamente após o início do seu *slot* estático no primeiro FC, como representado na Figura 30, onde FC_{bus} tem tamanho definido como sendo igual a $P_{h,j}^{ST}$.

De acordo com Pop et al. (2008), o CC decide se uma mensagem gerada por $S_{h,j}^{ST}$ será enviada para o barramento no início do *slot* associado. Se a mensagem é gerada imediatamente após o início do *slot*, ela terá que aguardar um FC completo até o início do próximo *slot* para ser transmitida. No exemplo ilustrado, $m_{h,j}^{ST,1}$ irá iniciar sua transmissão no primeiro *slot* do segundo FC. Como $m_{h,j}^{ST,1}$ tem tamanho unitário $(C_{h,j}^{ST} = 1)$, ela perderá seu *deadline* porque não existe tempo suficiente para concluir a transmissão antes de $D_{h,j}^{ST}$.

O exemplo acima mostra que o *deadline* de um fluxo de mensagens $S_{h,j}^{ST} \in \mathbb{S}^{ST}$ deve ser ao menos $C_{h,j}^{ST}$ maior do que o tamanho do FC: $D_{h,j}^{ST} \geq FC_{bus} + C_{h,j}^{ST}$. Considerando-se que i) o tamanho do ciclo FlexRay deve permitir a transmissão de quadros com diferentes *deadlines*, que ii) no pior caso o tamanho de um quadro do ST é igual a *gdStaticSlot* e que iii) $P_{h,j}^{ST} = D_{h,j}^{ST}$, chega-se à seguinte restrição imposta sobre o tamanho do ciclo FlexRay:

$$FC_{bus} \leq P_{min}^{ST} - gdStaticSlot \tag{5.8}$$

onde P_{min}^{ST} é o menor período dentre os fluxos de mensagens em $\mathbb{S}^{ST}$.

Considere agora o conceito de *freeze instant* apresentado na Seção 5.2.1. Neste caso, a decisão sobre a transmissão de $m_{h,j}^{ST,1}$ no ciclo atual será tomada no *freeze instant*. $m_{h,j}^{ST,1}$ deve chegar antes do *freeze instant* para que possa utilizar os *slots* alocados para o nodo h, incorrendo em um atraso adicional que deve ser considerado no tempo

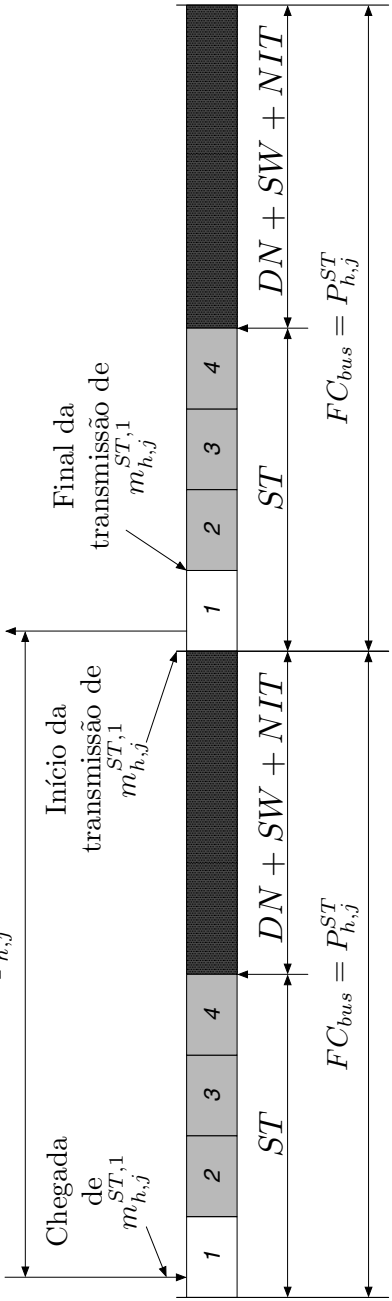

Figura 30 – Tamanho do Ciclo FlexRay

de transmissão de $m_{h,j}^{ST,1}$. Portanto, o período de um fluxo de mensagens $S_{h,j}^{ST} \in \mathbb{S}^{ST}$ deve ser no mínimo tão grande quanto o tamanho do FC (em inteiros de *gdStaticSlot*) acrescido do atraso introduzido pelo *freeze instant* do seu respectivo nodo:

$$P_{h,j}^{ST} \geq FC_{bus} + gdStaticSlot + \delta_{f_h}. \tag{5.9}$$

Isto leva à seguinte restrição imposta sobre o tamanho do ciclo FlexRay:

$$FC_{bus} \leq P_{min}^{ST} - (gdStaticSlot + \delta_{f_{max}}) \tag{5.10}$$

onde $\delta_{f_{max}}$ é o maior δ_f dos nodos do sistema.

No estudo apresentado em (SCHMIDT; SCHMIDT, 2009), os autores concluem que é favorável utilizar o maior valor possível para o tamanho do FC. De fato, definir um FC maior que o estritamente necessário para os fluxos de mensagens periódicas permite a reserva de *slots* estáticos para expansões futuras, e também permite alguma flexibilidade na definição do tamanho do DN.

Como descrito no Capítulo 2, um ciclo FlexRay é composto por um segmento Estático com tamanho ST_{bus}, um segmento Dinâmico com tamanho DN_{bus}, e dois segmentos de controle, NIT e SW. Assim sendo, o tamanho FC_{bus} do FC é dado por:

$$FC_{bus} = ST_{bus} + DN_{bus} + \theta \tag{5.11}$$

onde θ representa o tamanho dos segmentos de controle.

A especificação do FlexRay define que o DN não é obrigatório, e portanto

$$DN_{bus} \geq 0. \tag{5.12}$$

Das Equações 5.7, 5.11 e 5.12, segue-se que

$$\begin{aligned} FC_{bus} &= \sum_{h=1}^{|\mathbb{N}|} H_h + DN_{bus} + \theta \\ FC_{bus} &\geq \sum_{h=1}^{|\mathbb{N}|} H_h + \theta. \end{aligned} \tag{5.13}$$

Considerando-se agora ambas as restrições impostas sobre o tamanho do ciclo FlexRay (Equações 5.10 e 5.13), pode-se definir que a seguinte equação representa a *restrição de protocolo* que deve ser res-

peitada por qualquer método de alocação de *slots* estáticos H_h:

$$\sum_{h=1}^{|\mathbb{N}|} H_h + \theta \leq FC_{bus} \leq P_{min}^{ST} - (gdStaticSlot + \delta_{f_{max}}). \tag{5.14}$$

Como consequência, a restrição de protocolo impõe tanto um limite inferior como um limite superior para o tamanho do FC. Estas imposições garantem que existe um intervalo de tempo suficientemente grande para acomodar o número de *slots* estáticos alocados, e também garantem que tal intervalo de tempo está de acordo com as demandas de resposta do sistema.

5.2.2.2 Restrição de Deadline

Um método de alocação de *slots* estáticos deve garantir que o sistema é sempre escalonável. Em outras palavras, um método de alocação de *slots* estáticos deve garantir que todos os fluxos de mensagens periódicas consigam sempre transferir suas mensagens antes de seus *deadlines*.

Os tamanhos fixos do ciclo FlexRay, segmentos e *slots* estáticos permitem a estimativa da demanda gerada pelo conjunto de fluxos de mensagens de um nodo N_h e, como consequência, também é possível determinar o número de *slots* estáticos que devem estar disponíveis para N_h em um determinado intervalo de tempo de forma a transferir adequadamente todas as mensagens síncronas. Uma restrição de *deadline* pode, portanto, ser expressa como uma restrição que deve ser satisfeita com o fim de garantir que todas as mensagens periódicas geradas recebam ao menos um *slot* estático para transmissão antes da expiração de seus próprios *deadlines*.

Definindo-se que $R_{h,j}^{ST}$ é o tempo de transmissão no pior caso de uma mensagem gerada por $S_{h,j}^{ST}$ durante um intervalo de tempo $t_0 \in [0, +\infty[$, uma restrição de *deadline* pode ser definida como se segue:

$$R_{h,j}^{ST} \leq D_{h,j}^{ST},\ \forall S_{h,j}^{ST} \in \mathbb{S}^{ST} \tag{5.15}$$

Nesta proposta, considera-se que as mensagens geradas pelos fluxos serão encaminhadas para o barramento em um determinado instante (ou seja, no *freeze instant*) de acordo com uma política de escalonamento RM não preemptivo (em inglês, *non-preemptive RM*). Deste modo, o $R_{h,j}^{ST,l}$ de uma mensagem $m_{h,j}^{ST,l}$ gerada pelo fluxo $S_{h,j}^{ST}$ pode

ser calculado através de um conjunto de equações para sistemas de prioridade fixa (FPS) baseado nas equações propostas em (ANDERSSON; TOVAR, 2009; LIU; LAYLAND, 1973).

Inicialmente, deve-se reforçar que, como definido na Seção 5.1, todos os valores e parâmetros do sistema são expressos em números inteiros de *slots* estáticos, e tanto o tamanho das mensagens quanto dos *slots* estáticos é unitário.

Sem perda de generalidade, considere um FC com H_h *slots* estáticos alocados para o nodo N_h. O pior cenário para uma mensagem $m_{h,j}^{ST,l}$ gerada pelo fluxo $S_{h,j}^{ST}$ do nodo N_h é ser gerada imediatamente após um *freeze instant* f_h^0 pois, neste caso, ela será obrigada a aguardar um FC completo até o próximo *freeze instant* f_h^1 para competir pelos *slots* estáticos do nodo N_h. Este atraso inicial é denotado por:

$$f_h^1 = f_h^0 + FC_{bus}. \tag{5.16}$$

No pior caso, mensagens geradas por todos os fluxos em $hp(S_{h,j}^{ST})$ chegam no mesmo instante em que $m_{h,j}^{ST,l}$. Neste caso, no *freeze instant* f_h^1 as mensagens com prioridade mais alta que $m_{h,j}^{ST,l}$ geram uma demanda $\Theta_{h,j}^1$ de *slots* estáticos dada por

$$\Theta_{h,j}^1 = \sum_{d=1}^{|hp(S_{h,j}^{ST})|} 1 \tag{5.17}$$

onde $|hp(S_{h,j}^{ST})|$ é o número de elementos em $hp(S_{h,j}^{ST})$.

Como em cada FC existe um número H_h de *slots* estáticos alocados para o nodo N_h, a demanda $\Theta_{h,j}^1$ requer um número inteiro $\eta_{h,j}^1$ de FCs dado por:

$$\eta_{h,j}^1 = \lfloor \frac{\Theta_{h,j}^1}{H_h} \rfloor \tag{5.18}$$

restando um número $\iota_{h,j}^1$ de mensagens para serem transferidas dado por:

$$\iota_{h,j}^1 = \Theta_{h,j}^1 - \eta_{h,j}^1 \times H_h. \tag{5.19}$$

A transferência de $m_{h,j}^{ST,l}$ pode então ocorrer a partir do *freeze instant* f_h^2 dado $f_h^2 = f_h^1 + \eta_{h,j}^1 \times FC_{bus}$. Entretanto, até f_h^2 pode haver a chegada de novas mensagens geradas pelos fluxos em $hp(S_{h,j}^{ST})$, e a demanda gerada por estas novas mensagens também deve ser considerada. A demanda total gerada pelos fluxos de mensagens em $hp(S_{h,j}^{ST})$

até f_h^2 é dada por:

$$\Theta_{h,j}^2 = \sum_{d=1}^{|hp(S_{h,j}^{ST})|} \lceil \frac{f_h^2 - f_h^0}{P_{h,d}^{ST}} \rceil$$
$$= \sum_{d=1}^{|hp(S_{h,j}^{ST})|} \lceil \frac{(\eta_{h,j}^1 + 1) \times FC_{bus}}{P_{h,d}^{ST}} \rceil \quad (5.20)$$

requerendo um número inteiro de FCs dado por

$$\eta_{h,j}^2 = \lfloor \frac{\Theta_{h,j}^2}{H_h} \rfloor \quad (5.21)$$

restando um número $\iota_{h,j}^2$ de mensagens para serem transferidas no último FC:

$$\iota_{h,j}^2 = \Theta_{h,j}^2 - \eta_{h,j}^2 \times H_h. \quad (5.22)$$

É possível obter iterativamente o *freeze instant* f_h^g no qual $m_{h,j}^{ST,l}$ finalmente será selecionada para transmissão através do seguinte conjunto de equações:

$$\Theta_{h,j}^1 = \sum_{d=1}^{|hp(S_{h,j}^{ST})|} 1 \quad (5.23)$$

$$\eta_{h,j}^{\omega} = \lfloor \frac{\Theta_{h,j}^{\omega}}{H_h} \rfloor, \omega \geq 1 \quad (5.24)$$

$$\Theta_{h,j}^{\omega} = \sum_{x=1}^{|hp(S_{h,j}^{ST})|} \lceil \frac{(\eta_{h,j}^{\omega-1} + 1) \times FC_{bus}}{P_{h,d}^{ST}} \rceil, \omega > 1 \quad (5.25)$$

A iteração é interrompida quando $\Theta_{h,j}^{\omega+1} = \Theta_{h,j}^{\omega}$, sendo $\omega > 1$ (ω é um índice para iteração), ou quando $\eta_{h,j}^{\omega} \times FC_{bus} > D_{h,j}^{ST}$ (caso em que $m_{h,j}^{ST,l}$ irá perder seu *deadline*).

Por fim, o *wcrt* $R_{h,j}^{ST}$ para a transmissão de mensagens geradas por $S_{h,j}^{ST}$ é dado pela soma dos seguintes termos:

- Intervalo de tempo entre a chegada de $m_{h,j}^{ST,l}$ e o primeiro *freeze instant* f_h^0, no pior caso: $\Delta_{f_h} = FC_{bus}$;
- Número de FCs completos até o instante de *freeze* f_h^g no qual $m_{h,j}^{ST,l}$ é selecionada para ser transmitida: $\eta_{h,j}^{\omega} \times FC_{bus}$;

- Intervalo de tempo entre o *freeze instant* e o início da alocação do nodo N_h o último FC, que é igual a δ_{f_h};

- Número de *slots* estáticos utilizados para a transmissão de mensagens de prioridade mais alta no último FC ($\iota^{\omega}_{h,j} = \Theta^{\omega}_{h,j} - \eta^{\omega}_{h,j} \times H_j$);

- Um *slot* estático para a transmissão da própria mensagem $m^{ST,l}_{h,j}$.

O que resulta em um $R^{ST}_{h,j}$ dado por:

$$R^{ST}_{h,j} = FC_{bus} + \eta^{\omega}_{h,j} \times FC_{bus} + \delta_{f_h} + \iota^{\omega}_{h,j} + 1. \tag{5.26}$$

O problema do escalonamento de mensagens com prioridade fixa não-preemptivo é complexo e foi abordado em (ANDERSSON; TOVAR, 2009). Na abordagem de escalonamento proposta aqui, existem entretanto algumas simplificações. Considera-se que o tamanho das mensagens é unitário e igual a um *slot* estático e, como será explicado a seguir, devido ao *freeze instant*, mensagens com prioridade mais baixa que $m^{ST,l}_{h,j}$ não podem bloquear mensagens com prioridade mais alta do mesmo nodo, e portanto podem ser ignoradas na análise do tempo de resposta.

Em um dado *freeze instant*, se $m^{ST,l}_{h,j}$ já chegou, ela tem uma preferência de transmissão sobre as mensagens com prioridade mais baixa. Se $m^{ST,l}_{h,j}$ ainda não chegou, ela não será transmitida no ciclo corrente, não importando se mensagens de prioridade mais baixa chegaram ou não. Isso significa que, no contexto desta proposta, é possível ignorar completamente as mensagens com prioridade mais baixa que $m^{ST,l}_{h,j}$, o que simplifica consideravelmente a análise.

Para ilustrar a utilização da fórmula para a restrição de *deadline* exposta acima (Equação 5.26), considere o seguinte exemplo. Em um dado sistema, um nodo N_h possui três fluxos de mensagens $S^{ST}_{h,1}$, $S^{ST}_{h,2}$ e $S^{ST}_{h,3}$ com períodos $P^{ST}_{h,1} = 12$, $P^{ST}_{h,2} = 15$ e $P^{ST}_{h,3} = 35$, respectivamente. O ciclo FlexRay tem tamanho $FC_{bus} = 10$, sendo o tamanho do segmento Estático dado por $ST_{bus} = 4$. N_h tem uma alocação de dois *slots* estáticos (ou seja, $H_h = 2$), com os identificadores $FrameID = 1$ e $FrameID = 2$. Como o *RT-Middleware* irá despachar as mensagens de acordo com uma ordem RM, $S^{ST}_{h,1}$ tem a maior prioridade, e $S^{ST}_{h,3}$ a menor. Para simplificar o exemplo, define-se que $\delta_{f_h} = 1$.

O comportamento do sistema é ilustrado na Figura 31. Imediatamente após o *freeze instant* f^0_h ocorre a chegada simultânea das mensagens $m^{ST,1}_{h,1}$, $m^{ST,1}_{h,2}$ e $m^{ST,1}_{h,3}$. Como o *freeze instant* já ocorreu,

os *slots* estáticos alocados para N_h no primeiro ciclo ficarão vazios. No *freeze instant* seguinte (f_h^1) a fila no *RT-Middleware* contém as mensagens $m_{h,1}^{ST,1}$, $m_{h,2}^{ST,1}$ e $m_{h,3}^{ST,1}$. Devido às suas prioridades relativas, $m_{h,1}^{ST,1}$ e $m_{h,2}^{ST,1}$ são selecionadas para serem transferidas nos H_h *slots* estáticos do segundo FC, o que leva aos tempos de resposta $R_{h,1}^{ST} = 12$ e $R_{h,2}^{ST} = 13$, respectivamente. A mensagem $m_{h,3}^{ST,1}$ permanece na fila. Imediatamente após o instante de tempo 11 há a chegada da mensagem $m_{h,1}^{ST,2}$, e imediatamente após o instante 14 chega a mensagem $m_{h,2}^{ST,2}$. Em f_h^2 a fila do *RT-Middleware* contém $m_{h,1}^{ST,2}$, $m_{h,2}^{ST,2}$ e $m_{h,3}^{ST,1}$, sendo selecionadas para transferência $m_{h,1}^{ST,2}$ e $m_{h,2}^{ST,2}$. Imediatamente após o instante 23, chega $m_{h,1}^{ST,3}$. Como a nova mensagem gerada por $S_{h,2}^{ST}$ chega imediatamente APÓS o *freeze instant* f_h^3, em f_h^3 a fila do *RT-Middleware* contém apenas $m_{h,1}^{ST,3}$ e $m_{h,3}^{ST,1}$, sendo a mensagem $m_{h,3}^{ST,1}$ finalmente selecionada para ser transmitida com $R_{h,3}^{ST} = 33$.

Os valores para $R_{h,1}^{ST}$, $R_{h,2}^{ST}$ e $R_{h,3}^{ST}$ podem ser confirmados pela equação para a restrição de *deadline* (Equação 5.15).

Como $hp(S_{h,1}^{ST})$ é um conjunto vazio, tem-se que para $S_{h,1}^{ST}$, $\Theta_{h,1}^{\omega} = 0$, $\eta_{h,1}^{\omega} = 0$ e $\iota_{h,1}^{\omega} = 0$. Da Equação 5.26 tem-se que:

$$R_{h,1}^{ST} = 10 + 0 + 1 + 0 + 1 = 12. \tag{5.27}$$

Como $R_{h,1}^{ST} = 12 \leq D_{h,1}^{ST}$, o *deadline* de $S_{h,1}^{ST}$ é respeitado.
Para $S_{h,2}^{ST}$ tem-se:

$$\begin{aligned}
\Theta_{h,2}^{1} &= \sum_{d=1}^{|hp(S_{h,2}^{ST})|} 1 = 1 \\
\Theta_{h,2}^{2} &= \lceil \frac{(\lfloor \frac{1}{2} \rfloor + 1) \times 10}{12} \rceil = 1 \\
\Theta_{h,2}^{3} &= \lceil \frac{(\lfloor \frac{1}{2} \rfloor + 1) \times 10}{12} \rceil = 1 \\
\Theta_{h,2}^{\omega} &= \Theta_{h,2}^{3} = \Theta_{h,2}^{2} = 1 \\
\eta_{h,2}^{\omega} &= \lfloor \frac{1}{2} \rfloor = 0 \\
\iota_{h,2}^{\omega} &= \Theta_{h,2}^{\omega} - \eta_{h,2}^{\omega} \times H_h = 1 - 0 = 1 \\
R_{h,2}^{ST} &= 10 + 0 + 1 + 1 + 1 = 13.
\end{aligned} \tag{5.28}$$

Novamente, o *deadline* de $S_{h,2}^{ST}$ é respeitado ($S_{h,2}^{ST} = 13 \leq D_{h,2}^{ST}$).

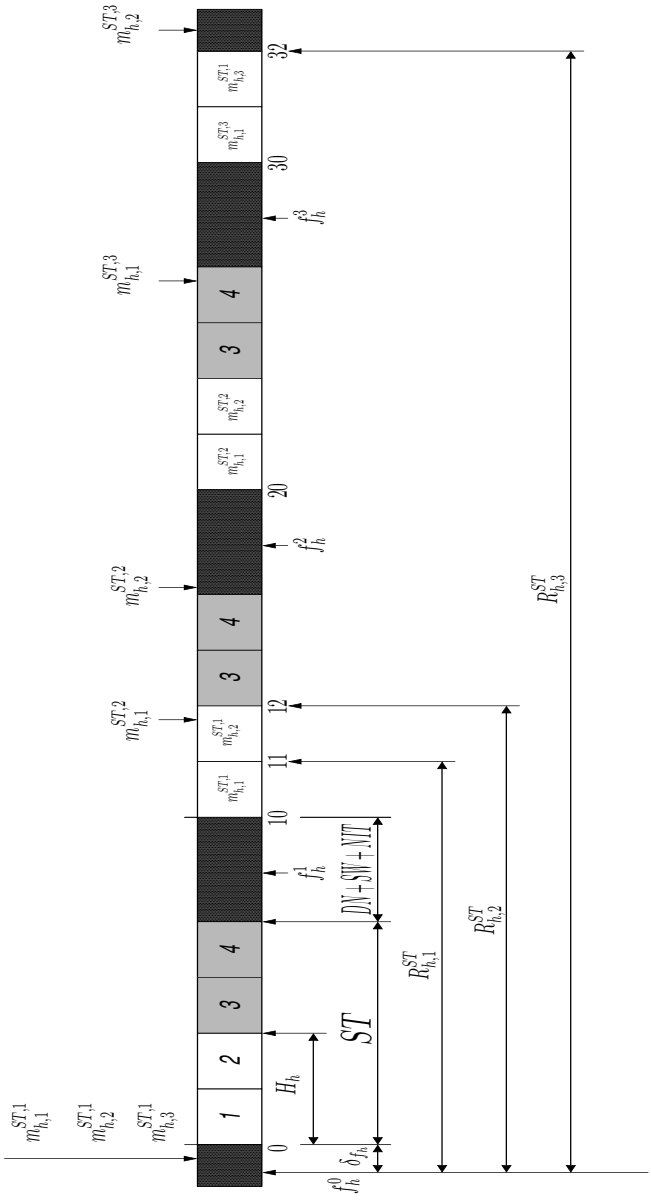

Figura 31 – Exemplo para a restrição de *deadline*

Finalmente, para $S_{h,3}^{ST}$ tem-se:

$$\begin{aligned}
\Theta_{h,3}^{1} &= \sum_{d=1}^{|hp(S_{h,3}^{ST})|} 1 = 2 \\
\Theta_{h,3}^{2} &= \lceil \frac{(\lfloor \frac{1}{2} \rfloor + 1) \times 10}{12} \rceil + \lceil \frac{(\lfloor \frac{1}{2} \rfloor + 1) \times 10}{15} \rceil = 2 \\
\Theta_{h,3}^{3} &= \lceil \frac{(\lfloor \frac{2}{2} \rfloor + 1) \times 10}{12} \rceil + \lceil \frac{(\lfloor \frac{2}{2} \rfloor + 1) \times 10}{15} \rceil = 4 \\
\Theta_{h,3}^{4} &= \lceil \frac{(\lfloor \frac{4}{2} \rfloor + 1) \times 10}{12} \rceil + \lceil \frac{(\lfloor \frac{4}{2} \rfloor + 1) \times 10}{15} \rceil = 5 \\
\Theta_{h,3}^{5} &= \lceil \frac{(\lfloor \frac{4}{2} \rfloor + 1) \times 10}{12} \rceil + \lceil \frac{(\lfloor \frac{4}{2} \rfloor + 1) \times 10}{15} \rceil = 5 \\
\Theta_{h,3}^{\omega} &= \Theta_{h,3}^{4} = \Theta_{h,3}^{5} = 5 \\
\eta_{h,3}^{\omega} &= \lfloor \frac{5}{2} \rfloor = 2 \\
\iota_{h,3}^{\omega} &= \Theta_{h,3}^{\omega} - \eta_{h,3}^{\omega} \times H_h = 5 - 4 = 1 \\
R_{h,3}^{ST} &= 10 + (2 \times 10) + 1 + 1 + 1 = 33.
\end{aligned} \tag{5.29}$$

O *deadline* de $S_{h,3}^{ST}$ também é respeitado, o que significa que a restrição de *deadline* é respeitada no exemplo acima.

Este resultado está de acordo com os valores observados na Figura 31.

5.2.3 Heurística Para Definição da Alocação H_j de *Slots* Estáticos

Por fim, existe a necessidade de definir a alocação H_j de *slots* estáticos para cada nodo N_j. Nesta seção, propõe-se o uso de uma heurística adequada para derivar uma alocação que respeita os requisitos impostos tanto pela restrição de protocolo (Equação 5.14) quanto pela restrição de *deadline* (Equação 5.15). Esta heurística é chamada Método de Alocação Proporcional (*Proportional Allocation Scheme*, PAS).

Na heurística para definição da alocação de *slots* estáticos proposta, o número H_h de *slots* estáticos alocados para o nodo N_h é dado por:

$$H_h = \lceil \sum_{j=1}^{|\mathbb{S}_h^{ST}|} \frac{FC_{bus}}{P_{h,j}^{ST}} \rceil. \tag{5.30}$$

A racionalização por trás da equação proposta é a seguinte: considere que uma mensagem do fluxo $S_{h,j}^{ST}$ é fragmentada e que os fragmentos resultantes estão espalhados entre os *slots* estáticos durante o período $P_{h,j}^{ST}$. $FC_{bus}/P_{h,j}^{ST}$ representa então o percentual de utilização do *slot* durante cada FC. Considere agora o conjunto de fluxos de mensagens $\mathbb{S}_h^{ST}$ relacionado ao nodo N_h. A Equação 5.30 representa o número de *slots* requeridos para servir todos os fluxos de mensagens do nodo N_h. Assim, a equação representa o número de *slots* estáticos que devem ser alocados para N_h de forma a garantir que o *deadline* de todas as mensagens geradas no nodo sejam respeitados.

Para definir o segmento Estático de um sistema FlexRay, pode-se utilizar então o Algoritmo 1.

Algoritmo 1 Algoritmo para o Método de Alocação Proporcional

1: 1 - Definir um FC_{bus} com a maior duração possível (Equação 5.14)
2: **for** h:= 1 to $|\mathbb{N}|$ **do**
3: Define a alocação H_h (Equação 5.30)
4: **end for**
5: 2 - Testa se a restrição de protocolo (Equação 5.14) é respeitada
6: **for** h:= 1 to $|\mathbb{N}|$ **do**
7: Testa se a restrição de *deadline* (Equação 5.15) é respeitada para o nodo N_h
8: **end for**

5.2.4 Exemplo: Uso do Método de Alocação Proporcional

Esta seção apresenta um exemplo que ilustra o uso do método de alocação de *slots* estáticos proposto.

Considere um sistema composto por 3 nodos, com $\delta_{f_{max}} = 1$. Cada nodo tem um conjunto de fluxos de mensagens cujos períodos são apresentados na Tabela 3, e o tamanho dos segmentos de controle é $\theta = 1$. É importante salientar que todos os valores são dados em inteiros de *slots* estáticos.

Como exposto na Seção 5.2.3, o Algoritmo 1 deve ser seguido para que o segmento Estático de um sistema FlexRay seja definido. Isto leva aos seguintes passos:

1) Definir o tamanho FC_{bus} como a maior duração possível.

De acordo com o exposto na Seção 5.2.2.1, é favorável usar o maior valor possível para o tamanho do FC, dado pela Equação 5.10:

nodo N_1	$P^{ST}_{1,1}$	$P^{ST}_{1,2}$	$P^{ST}_{1,3}$	$P^{ST}_{1,4}$	
	12	15	29	50	
nodo N_2	$P^{ST}_{2,1}$	$P^{ST}_{2,2}$	$P^{ST}_{2,3}$		
	23	33	100		
nodo N_3	$P^{ST}_{3,1}$	$P^{ST}_{3,2}$	$P^{ST}_{3,3}$	$P^{ST}_{3,4}$	$P^{ST}_{3,5}$
	12	23	29	37	44

Tabela 3 – Conjunto de fluxos de mensagens

$$\begin{aligned} FC_{bus} &\leq P^{ST}_{min} - (gdStaticSlot + \delta_{f_{max}}) \\ P_{min} &= 12 \\ \delta_{f_{max}} &= 1 \\ FC_{bus} &\leq 12 - (1+1) \\ FC_{bus} &= 10. \end{aligned} \tag{5.31}$$

2) Para todos os nodos do sistema, calcular a alocação H_h de *slots* estáticos.

O segundo passo do algoritmo utiliza o Método de Alocação Proporcional (Equação 5.30) para calcular o número de *slots* estáticos alocados para cada nodo do sistema. Os cálculos relacionados são reproduzidos abaixo.

Para o nodo N_1, a alocação H_1 é:

$$H_1 = \lceil \frac{10}{12} + \frac{10}{15} + \frac{10}{29} + \frac{10}{50} \rceil = 3. \tag{5.32}$$

Para o nodo N_2, a alocação H_2 é:

$$H_2 = \lceil \frac{10}{23} + \frac{10}{33} + \frac{10}{100} \rceil = 1. \tag{5.33}$$

Finalmente, para o nodo N_3, a alocação H_3 é:

$$H_3 = \lceil \frac{10}{12} + \frac{10}{23} + \frac{10}{29} + \frac{10}{37} + \frac{10}{44} \rceil = 3. \tag{5.34}$$

A Tabela 4 resume a alocação para cada nodo.

3) Testar se a restrição de protocolo é respeitada.

Como explicado na Seção 5.2.2, qualquer método de alocação de *slots* estáticos deve satisfazer a restrição de protocolo. Como a alocação

$\boldsymbol{H_1}$	3
$\boldsymbol{H_2}$	1
$\boldsymbol{H_3}$	3

Tabela 4 – Alocação para os nodos do Sistema

H_h para cada nodo N_h foi calculada no passo anterior, a restrição de protocolo pode ser verificada com a utilização da Equação 5.14:

$$\begin{aligned} \sum_{h=1}^{|\mathbb{N}|} H_h + \theta \quad &\leq FC_{bus} \quad \leq P_{min} - (1 + \delta_{f_{max}}) \\ 3 + 1 + 3 + 1 \quad &\leq 10 \quad \leq 12 - 2 \\ 8 \quad &\leq 10 \quad \leq 10. \end{aligned} \tag{5.35}$$

Sendo que a restrição de protocolo é respeitada para o sistema FlexRay deste exemplo.

4) Para todos os nodos do sistema, verificar se a restrição de *deadline* é respeitada.

Utiliza-se as informações da alocação para verificar se a restrição de *deadline* é respeitada. Um resumo dos cálculos relacionados é apresentado a seguir.

Para o nodo N_1:

$$\begin{aligned} R_{1,1}^{ST} &= 12 \leq D_{1,1}^{ST} \\ R_{1,2}^{ST} &= 13 \leq D_{1,2}^{ST} \\ R_{1,3}^{ST} &= 14 \leq D_{1,3}^{ST} \\ R_{1,4}^{ST} &= 24 \leq D_{1,4}^{ST}. \end{aligned} \tag{5.36}$$

Como os *deadlines* são respeitados para todos os fluxos de mensagens, a restrição de *deadline* é respeitada para o nodo N_1.

Para o nodo N_2:

$$\begin{aligned} R_{2,1}^{ST} &= 12 \leq D_{2,1}^{ST} \\ R_{2,2}^{ST} &= 22 \leq D_{2,2}^{ST} \\ R_{2,3}^{ST} &= 62 \leq D_{2,3}^{ST}. \end{aligned} \tag{5.37}$$

Como os *deadlines* são respeitados para todos os fluxos de mensagens, a restrição de *deadline* é respeitada para o nodo N_2.

Finalmente, para o nodo N_3:

$$\begin{aligned} R_{3,1}^{ST} &= 12 \leq D_{3,1}^{ST} \\ R_{3,2}^{ST} &= 13 \leq D_{3,2}^{ST} \\ R_{3,3}^{ST} &= 14 \leq D_{3,3}^{ST} \\ R_{3,4}^{ST} &= 23 \leq D_{3,4}^{ST} \\ R_{3,5}^{ST} &= 24 \leq D_{3,5}^{ST}. \end{aligned} \tag{5.38}$$

Como os *deadlines* são respeitados para todos os fluxos de mensagens, a restrição de *deadline* também é respeitada para o nodo N_3.

A Tabela 5 resume, para cada fluxo de mensagens do sistema, o tempo de transmissão no pior caso calculado, o *deadline* e se a restrição de *deadline* é respeitada.

	$\boldsymbol{R_{1,j}^{ST}}$	$\boldsymbol{D_{1,j}^{ST}}$	$\boldsymbol{R_{1,j}^{ST} \leq D_{1,j}^{ST}}$
nodo N_1	$R_{1,1}^{ST} = 12$	$D_{1,1}^{ST} = 12$	$R_{1,1}^{ST} \leq D_{1,1}^{ST}$ ok
	$R_{1,2}^{ST} = 13$	$D_{1,2}^{ST} = 15$	$R_{1,2}^{ST} \leq D_{1,2}^{ST}$ ok
	$R_{1,3}^{ST} = 14$	$D_{1,3}^{ST} = 29$	$R_{1,3}^{ST} \leq D_{1,3}^{ST}$ ok
	$R_{1,4}^{ST} = 24$	$D_{1,4}^{ST} = 50$	$R_{1,4}^{ST} \leq D_{1,4}^{ST}$ ok
	$\boldsymbol{R_{2,j}^{ST}}$	$\boldsymbol{D_{2,j}^{ST}}$	$\boldsymbol{R_{2,j}^{ST} \leq D_{2,j}^{ST}}$
nodo N_2	$R_{2,1}^{ST} = 12$	$D_{2,1}^{ST} = 23$	$R_{2,1}^{ST} \leq D_{2,1}^{ST}$ ok
	$R_{2,2}^{ST} = 22$	$D_{2,2}^{ST} = 23$	$R_{2,2}^{ST} \leq D_{2,2}^{ST}$ ok
	$R_{2,3}^{ST} = 62$	$D_{2,3}^{ST} = 100$	$R_{2,3}^{ST} \leq D_{2,3}^{ST}$ ok
	$\boldsymbol{R_{3,j}^{ST}}$	$\boldsymbol{D_{3,j}^{ST}}$	$\boldsymbol{R_{3,j}^{ST} \leq D_{3,j}^{ST}}$
nodo N_3	$R_{3,1}^{ST} = 12$	$D_{3,1}^{ST} = 23$	$R_{3,1}^{ST} \leq D_{3,1}^{ST}$ ok
	$R_{3,2}^{ST} = 22$	$D_{3,2}^{ST} = 23$	$R_{3,2}^{ST} \leq D_{3,2}^{ST}$ ok
	$R_{3,3}^{ST} = 62$	$D_{3,3}^{ST} = 100$	$R_{3,3}^{ST} \leq D_{3,3}^{ST}$ ok
	$R_{3,4}^{ST} = 22$	$D_{3,4}^{ST} = 23$	$R_{3,4}^{ST} \leq D_{3,4}^{ST}$ ok
	$R_{3,5}^{ST} = 62$	$D_{3,5}^{ST} = 100$	$R_{3,5}^{ST} \leq D_{3,5}^{ST}$ ok

Tabela 5 – Resumo dos valores para a restrição de deadline no Exemplo 1

Como tanto a restrição de protocolo quanto a restrição de *deadline* são respeitadas, o Método de Alocação Proporcional gerou uma alocação de *slots* estáticos que garante que o sistema utilizado no exemplo é escalonável.

Este exemplo claramente mostra como o método de alocação de *slots* estáticos proposto, em conjunto com as restrições de protocolo e *deadline*, é utilizado para gerar um conjunto de parâmetros viáveis

para um dado sistema FlexRay.

Na próxima seção será proposta uma abordagem que permite a integração do método de alocação de *slots* estáticos proposto com o padrão AUTOSAR.

5.2.5 Integração do Método de Alocação Proposto com o AUTOSAR

O padrão AUTOSAR (AUTOSAR, 2012), discutido brevemente no Capítulo 2, Seção 2.2, página 53, impõe diversas restrições no projeto de um sistema FlexRay, entre elas uma associação entre os fluxos de mensagens, *slots* estáticos, ciclo FlexRay base e repetições de ciclo que não pode ser alterada em um sistema que está no estado de execução.

A utilização das técnicas propostas neste capítulo dentro de um ambiente AUTOSAR permite a flexibilização de certas características do sistema, como a decisão sobre quais sinais serão transmitidos durante o ST durante a execução do sistema. A técnica proposta torna possível o projeto de sistemas onde não existe uma sincronização entre as tarefas, sinais ou fluxos de mensagens e o ciclo FlexRay, e também permite que sejam considerados sistemas nos quais os períodos dos fluxos de mensagens não são múltiplos inteiros do ciclo FlexRay.

As técnicas propostas neste capítulo dependem do uso de um *middleware* de tempo real que define em tempo de execução qual mensagem periódica será transmitida nos *slots* estáticos associados ao nodo N_j. Entretanto, o AUTOSAR define regras rígidas em relação à associação de sinais/mensagens a *Protocol Data Units* (PDUs) e posteriormente a quadros e *slots* do FlexRay. Como o *RT-Middleware* implementa uma associação dinâmica entre as mensagens geradas nos fluxos de cada nodo e os *slots* estáticos associados a cada nodo, a especificação do AUTOSAR prejudica sua implementação abaixo da camada de aplicação. Entretanto, o *middleware* pode ser facilmente implementado como um *Software Component* (SW-C) na camada de aplicação do AUTOSAR, como será descrito nesta seção.

O *middleware* de tempo real a ser integrado com o AUTOSAR (nomeado aqui como RTM-A) nada mais é que o *RT-Middleware* descrito na Seção 5.2.1. No *RT-Middleware*, as mensagens associadas ao ST que são geradas no nodo N_j são colocadas em *buffers* à espera do despacho para a camada inferior. Em um *freeze instant* que é sincronizado com um FlexRay Job do AUTOSAR, um *dispatcher* move mensagens do *buffer* para a camada inferior de acordo com uma política

FPS.

Devido às restrições impostas pelo AUTOSAR em relação à construção de PDUs, o nodo emissor deve empacotar uma mensagem por um de seus fluxos em um tipo especial de PDU (denominado $ISignalIPDU$ nesta proposta) antes de enviá-la para a camada do RTE. O $ISignalIPDU$ é criado por um componente de software chamado $ISignalIPDUBuilder$, e será posteriormente desempacotado no nodo receptor, sendo que a mensagem que o PDU contém será então encaminhada para a aplicação destino por outro componente de software chamado $ISignalIPDUFilter$. A arquitetura proposta para o RTM-A é representada na Figura 32.

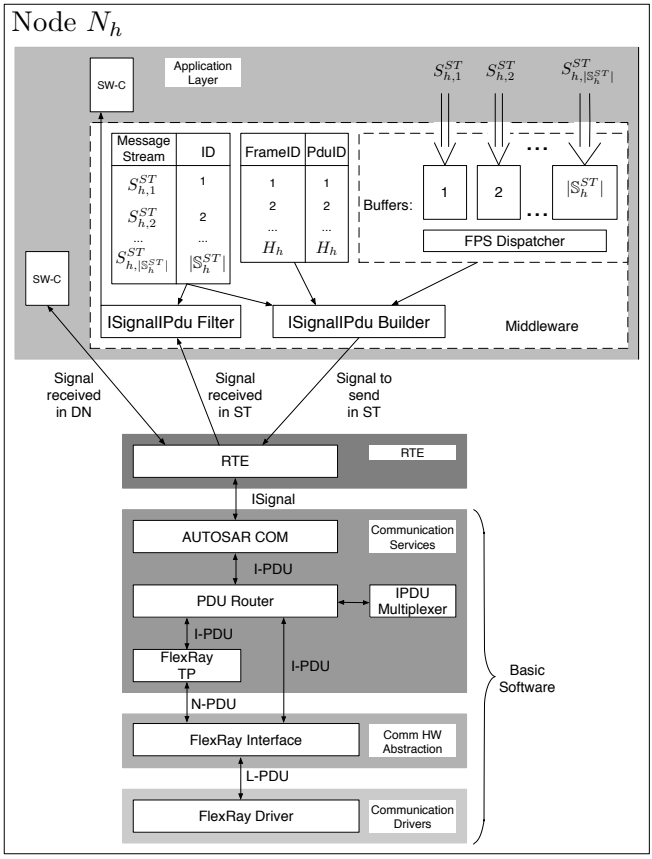

Figura 32 – Arquitetura proposta para o RTM-A

O *payload* do $ISignalIPDU$ é composto por a) um segmento ID contendo um identificador que permite ao $ISignalIPDU$ Filter conhecer a aplicação destino e b) a própria mensagem a ser enviada. O $ISignalIPDU$ é gerado pelo componente $ISignalIPDU$ *Builder*, que é parte integrante do *RT-Middleware*. O $ISignalIPDU$ *Builder* é representado na Figura 33.

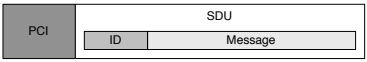

Figura 33 – $ISignalIPDU$ gerado pelo $ISignalIPDU$ Builder

O campo identificador do $ISignalIPDU$ no qual a mensagem será enviada contém as informações necessárias para que o mesmo seja roteado até o *slot* estático desejado. Para que o $ISignalIPDU$ seja construído com o identificador correto, o $ISignalIPDUBuilder$ deverá consultar uma tabela interna que indica o *slot* estático atual, a alocação do nodo e o identificador que representa o *slot* estático desejado.

Com esta arquitetura para o RTM-A, fica claro como o método de alocação de *slots* estáticos proposto neste capítulo pode ser utilizado em uma implementação baseada no AUTOSAR.

Na próxima seção será apresentada uma extensão do Método de Alocação Proporcional.

5.3 MÉTODO DE ALOCAÇÃO PROPORCIONAL ADAPTATIVO

Nesta seção é apresentada uma extensão do método proposto na Seção 5.2. Naquele método, pressupunha-se a existência de um único *freeze instant* por ciclo FlexRay. No método proposto nesta Seção, assume-se que existe um *freeze instant* **para cada *slot* estático**, resultando em um valor menos pessimista para a restrição de *deadline*.

A existência de um *freeze instant* por *slot* estático tem impacto direto no modo como é definida a alocação H_h para um nodo N_h. Torna-se necessário um novo conjunto de equações para definir H_h, não sendo mais possível utilizar o Método de Alocação Proporcional proposto na Seção 5.2.3.

Nesta seção será apresentado um novo método para a alocação de *slots* estáticos chamado Método de Alocação Proporcional Adaptativo. Inicialmente (Seções 5.3.1 a 5.3.2.3) será discutido o impacto da existência de um *freeze instant* sobre as restrições de protocolo e

deadline, bem como apresentado um novo conjunto de equações para a restrição de *deadline* e um exemplo de uso. A seguir (Seção 5.3.3), é proposto um algoritmo para a definição da alocação de *slots* estáticos e, por fim, na Seção 5.3.4 o Método de Alocação Proporcional Adaptativo é experimentalmente comparado com uma proposta do estado da arte relacionado ao escalonamento no ST.

5.3.1 *Middleware* de Tempo Real

Assim como na proposta apresentada na Seção 5.2, a proposta apresentada neste capítulo utiliza um *middleware* de tempo real (*Real-Time Middleware*, RTM) que irá arbitrar, em tempo de execução, qual mensagem será transmitida em cada *slot* estático de um FC. Este *middleware* de tempo real tem características semelhantes às características do RTM apresentado na Seção 5.2.1: Em um *freeze instant* $f^g_{STS_e}$, sendo STS_e ($e = 1, 2, ..., gNumberOfStaticSlots$) um *slot* estático e o índice g denotando o g-ésimo instante de *freeze*, o *dispatcher* do RTM irá remover uma das mensagens de seus *buffers* locais, empacotar esta mensagem em um quadro FlexRay e mover o quadro para o *buffer* de entrada do CHI. O *freeze instant* é sincronizado com o ciclo FlexRay, e o intervalo de tempo $\delta_{f_{STS_e}}$ entre o instante de *freeze* $f^g_{STS_e}$ e o *slot* estático STS_e correspondente deve ser suficiente para acomodar todo o processo de *dispatch* a nível de CPU, incluindo a geração do quadro FlexRay a ser transmitido.

Diferentemente do método proposto na Seção 5.2.1 onde era considerada a existência de um único *freeze instant* por alocação H_h, o Método de Alocação Proporcional Adaptativo considera que existe um *freeze instant* **por *slot* estático**. Na Figura 34 é ilustrado um exemplo de sistema que contém um *freeze instant* por *slot* estático. No exemplo, os instantes de *freeze* $f^g_{STS_3}$, $f^{g+1}_{STS_3}$, $f^g_{STS_4}$ e $f^{g+1}_{STS_4}$ estão associados, respectivamente, com os *slots* estáticos 3 e 4 que foram alocados para o nodo N_h (o nodo N_h tem, portanto, uma alocação de *slots* estáticos $H_h = 2$). Note que o primeiro FC (FC1) contém os *freeze instants* $f^g_{STS_3}$ e $f^g_{STS_4}$, e que no segundo FC (FC2) o índice dos *freeze instants* foi incrementado.

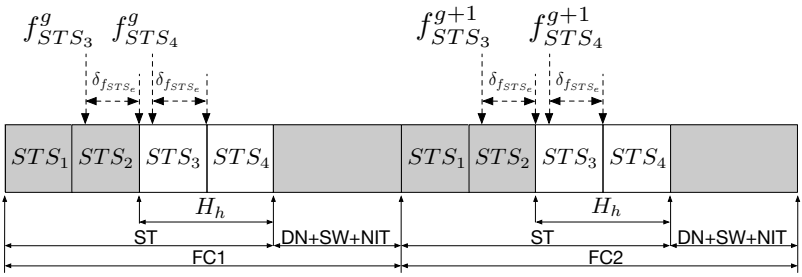

Figura 34 – *freeze instants* $f^{g}_{STS_3}$, $f^{g+1}_{STS_3}$, $f^{g}_{STS_4}$ e $f^{g+1}_{STS_4}$ associados com os *slots* estáticos STS_3 e STS_4, alocados para o nodo N_h

5.3.2 Restrições de Protocolo e *Deadline*

Na Seção 5.2 foram apresentadas duas restrições, chamadas restrições de protocolo e de *deadline*, que o Método de Alocação Proporcional deve respeitar para que seja possível definir uma alocação H_h de *slots* estáticos para um nodo N_h. Mas as equações relacionadas a tais restrições consideravam a existência de um único *freeze instant* por FC.

Antes que se possa afirmar que as equações anteriormente apresentadas para ambas as restrições de protocolo e de *deadline* podem ser utilizadas diretamente com o método proposto nesta seção, deve-se analisar o impacto que a existência de um *freeze instant* tem sobre as mesmas. A seguir, será inicialmente analisado tal impacto sobre a restrição de protocolo, e na sequência, sobre a restrição de *deadline*.

5.3.2.1 Restrição de Protocolo

Na Seção 5.2.2.1 foi apresentada a seguinte restrição de protocolo:

$$\sum_{h=1}^{|\mathbb{N}|} H_h + \theta \leq FC_{bus} \leq P^{ST}_{min} - (gdStaticSlot + \delta_{f_{max}}). \qquad (5.39)$$

Tal equação impõe um limite superior para o tamanho do Ciclo FlexRay dado por $P^{ST}_{min} - (gdStaticSlot + \delta_{f_{max}})$, onde P^{ST}_{min} é o menor período dentre os períodos dos fluxos de mensagens escalonados no

ST e $\delta_{f_{max}}$ é o maior $\delta_{f_{STS_e}}$ no sistema. No Método de Alocação Proporcional, δ_{f_h} representa o intervalo de tempo entre o início de um *freeze instant* f_h^g e o início da alocação H_h do nodo N_h.

De forma resumida, a primeira parte da Equação 5.39 significa que o total de *slots* estáticos alocados para os nodos de um sistema, acrescido do intervalo de tempo necessário para os segmentos de controle do FC (dado por θ) não pode ser maior que FC_{bus}.

A terceira parte da Equação 5.39 implica que o tamanho FC_{bus} do FC deve ser tal que uma mensagem $m_{h,j}^{ST,l}$ gerada por $S_{h,j}^{ST}$ não perca seu *deadline*, mesmo i) que $S_{h,j}^{ST}$ possua o menor período do sistema, ii) que $m_{h,j}^{ST,l}$ seja gerada imediatamente após o último *freeze instant* associado com a alocação H_h de N_h e iii) que seja considerado o atraso induzido por $gdStaticSlot + \delta_{f_{STS_e}}$ (tempo necessário para a transmissão de $m_{h,j}^{ST,l}$, no pior caso, acrescido de $\delta_{f_{STS_e}}$).

Se for considerado um sistema onde um nodo N_h tem uma alocação $H_h = 1$ e possui um fluxo de mensagens $S_{h,j}^{ST}$, é fácil observar que a restrição de protocolo apresentada anteriormente se mantém: se FC_{bus} for maior que $P_{min}^{ST} - (gdStaticSlot + \delta_{f_{max}})$, as mensagens geradas por $S_{h,j}^{ST}$ perderão seu *deadline*, independente do fato de haver um *freeze instant* por *slot* estático.

5.3.2.2 Restrição de *Deadline*

Uma restrição de *deadline* pode ser expressa como uma restrição que deve ser respeitada para que seja possível garantir que todas as mensagens geradas por *slots* estáticos serão transferidas antes de seus respectivos *deadlines*. Na Seção 5.2.2.2 foi definida uma restrição de *deadline* adequada para o Método de Alocação Proporcional. Entretanto, como aquele método considera a existência de um único *freeze instant* por FC, tal restrição precisa ser modificada para atender aos requisitos do Método de Alocação Proporcional Adaptativo proposto nesta seção. Mas assim como ocorre para a restrição de *deadline* proposta na Seção 5.2.2.2, o *wcrt* $R_{h,j}^{ST}$ de uma mensagem $m_{h,j}^{ST,l}$ gerada pelo fluxo $S_{h,j}^{ST}$ pode ser obtido utilizando-se uma abordagem semelhante ao método para análise de sistemas de prioridade fixa (*fixed priority systems*, FPS) proposto em (AUDSLEY et al., 1993) e (ANDERSSON; TOVAR, 2009).

Inicialmente considere o seguinte exemplo onde um sistema FlexRay possui três fluxos de mensagens periódicos. Será analisada uma mensagem $m_{h,3}^{ST,1}$ gerada pelo fluxo $S_{h,3}^{ST}$ (Figura 35). É importante

lembrar que os valores e parâmetros do sistema são expressos em múltiplos inteiros de *gdStaticSlot*, que as mensagens são enviadas para o CC durante os *freeze instants*, que existe um *freeze instant* por *slot* estático e que são alocados H_h *slots* estáticos contíguos para cada nodo N_h.

O pior cenário possível para $m_{h,3}^{ST,1}$ é ser gerada simultaneamente com mensagens de todos os outros fluxos em $hp(S_{h,3}^{ST})$, e imediatamente após o *freeze instant* do último *slot* estático alocado para o nodo N_h. Neste caso, $m_{h,3}^{ST,1}$ sofrerá a maior interferência possível de $hp(S_{h,3}^{ST})$, sendo que todas as mensagens serão forçadas a aguardar até o primeiro *freeze instant* do primeiro *slot* estático da alocação H_h do FC seguinte (Figura 35).

O fato de que o segundo *slot* estático do primeiro FC não pode ser utilizado representa um atraso inicial $B_{h,3}^{ST}$ equivalente a um *slot* estático:

$$B_{h,3}^{ST} = gdStaticSlot. \tag{5.40}$$

No primeiro *freeze instant* da alocação no segundo FC (representado como $f_{STS_1}^{g+1}$ na Figura 35), o *middleware* de tempo real irá selecionar a mensagem que possui a maior prioridade ($m_{h,1}^{ST,1}$) dos seus *buffers*, irá empacotar esta mensagem em um quadro, enviar o quadro para o CC e remover a mensagem dos seus *buffers*. No segundo *freeze instant* do segundo FC (f_{STS_2} na Figura 35) o *dispatcher* irá novamente selecionar nos seus *buffers* a mensagem com maior prioridade (que agora é $m_{h,2}^{ST,1}$), empacotar, enviar a mesma para o CC e a remover do *buffer*. Este processo é repetido até que não existam mais mensagens geradas pelos fluxos em $hp(S_{h,3}^{ST})$, quando então $m_{h,3}^{ST,1}$ será finalmente selecionada para ser transferida.

Entretanto, até o *freeze instant* f_{STS_e} no qual $m_{h,3}^{ST,1}$ pode ser enviada, existirão *slots* do ST e segmentos do FC nos quais N_h não poderá transmitir mensagens. Além disso, poderão haver novas gerações de mensagens por parte dos fluxos em $hp(S_{h,3}^{ST})$ (por exemplo, $m_{h,1}^{ST,2}$ na Figura 35). Estas interferências devem ser consideradas quando é calculado o tempo de transmissão no pior caso de $m_{h,3}^{ST,1}$.

A parcela do FC que não pode ser utilizada pelo nodo N_h pode ser modelada como um fluxo de mensagem de alta prioridade $S_{h,0}^{ST}$ cujo período $P_{h,0}^{ST} = FC_{bus}$ e cujo tempo de transmissão é $C_{h,0}^{ST} = FC_{bus} - H_h$.

Observando-se o exposto acima e seguindo o raciocínio proposto em (AUDSLEY et al., 1993) para escalonar FPSs, chega-se à conclusão

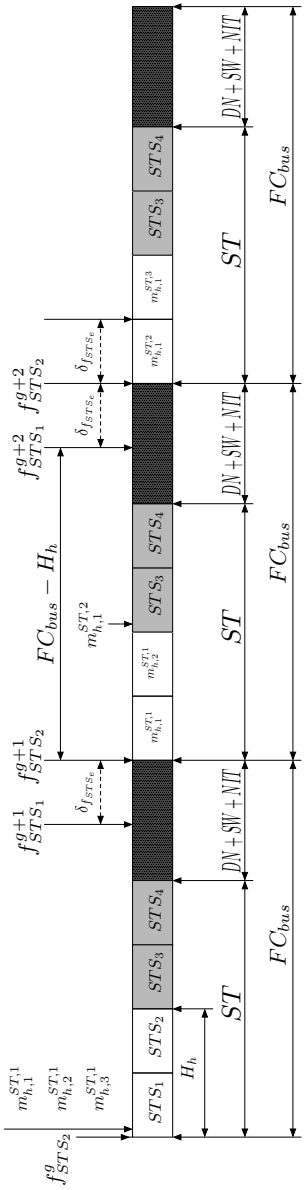

Figura 35 – Tamanho do Ciclo FlexRay: Exemplo 2

que é possível obter iterativamente o *freeze instant* f_{STS_e} no qual uma mensagem $m_{h,j}^{ST,l}$ qualquer será finalmente selecionada para ser transferida no barramento através da recursão descrita pela Equação 5.42, para a qual o valor inicial de $\Theta_{h,j}^{\omega}$ é dado pela Equação 5.41.

$$\Theta_{h,j}^{1} = B_{h,j}^{ST} + C_{h,0}^{ST} + \sum_{d=1}^{|hp(S_{h,j}^{ST})|} C_{h,d}^{ST} \tag{5.41}$$

$$\Theta_{h,j}^{\omega} = B_{h,j}^{ST} + \left\lceil \frac{\Theta_{h,j}^{\omega-1}}{P_{h,0}^{ST}} \right\rceil \times C_{h,0}^{ST} + \sum_{d=1}^{|hp(S_{h,j}^{ST})|} \left\lceil \frac{\Theta_{h,j}^{\omega-1}}{P_{h,d}^{ST}} \right\rceil, \omega > 1. \tag{5.42}$$

Como discutido no Capítulo 2, Seção 2.1, página 37, *slots* do ST possuem tamanho fixo *gdStaticSlot*, e a transmissão de um quadro pode iniciar apenas após o início do *slot* estático associado com o $FrameID$ daquele quadro, mesmo se o *slot* anterior foi apenas parcialmente ocupado. Portanto, o tempo de transmissão dos fluxos em $hp(S_{h,j}^{ST})$ pode ser considerado igual a *gdStaticSlot*: $\forall S_{h,d}^{ST} \in hp(S_{h,j}^{ST})$, $C_{h,d}^{ST} = gdStaticSlot$.

Sendo que $B_{h,j}^{ST}$ e os tempos de transmissão dos fluxos em $\mathbb{S}^{ST}$ são dados em inteiros de *gdStaticSlot*, as Equações 5.43 e 5.44 podem ser reescritas como:

$$\Theta_{h,j}^{1} = gdStaticSlot + C_{h,0}^{ST} + \sum_{d=1}^{|hp(S_{h,j}^{ST})|} gdStaticSlot \tag{5.43}$$

$$\Theta_{h,j}^{\omega} = gdStaticSlot + \left\lceil \frac{\Theta_{h,j}^{\omega-1}}{P_{h,0}^{ST}} \right\rceil \times C_{h,0}^{ST} + \sum_{d=1}^{|hp(S_{h,j}^{ST})|} \left\lceil \frac{\Theta_{h,j}^{\omega-1}}{P_{h,d}^{ST}} \right\rceil gdStaticSlot, \omega > 1. \tag{5.44}$$

A iteração termina quando $\Theta_{h,j}^{\omega+1} = \Theta_{h,j}^{\omega}$, ou quando $\Theta_{h,j}^{\omega} > D_{h,j}^{ST}$, sendo $\omega > 1$ (caso no qual $m_{h,j}^{ST,l}$ irá perder seu *deadline*).

Por fim, o tempo de transmissão no pior caso $R_{h,j}^{ST}$ para a trans-

missão de mensagens geradas pelo fluxo $S_{h,j}^{ST}$ é dado pela soma dos seguintes termos:

- O intervalo de tempo entre o *freeze instant* e o início do *slot* estático associado (*freeze interval)*, que é igual a $\delta_{f_{STS_e}}$;
- A interferência $\Theta_{h,j}^{\omega}$ devido a mensagens com prioridade mais alta e segmentos do FC nos quais transmissões de mensagens geradas no nodo N_h não são permitidas;
- O intervalo de tempo necessário para a transmissão do quadro FlexRay que contém $m_{h,j}^{ST,l}$.

A **restrição de deadline** resultante é dada por:

$$R_{h,j}^{ST} = \delta_{f_{STS_e}} + \Theta_{h,j}^{\omega} + C_{h,j}^{ST} \leq D_{h,j}^{ST}. \tag{5.45}$$

O problema do escalonamento de mensagens em sistemas de prioridade fixa não-preemptivos é complexo e foi abordado em (DAVIS et al., 2007). Na abordagem proposta nesta seção assume-se, entretanto, algumas simplificações. É considerado que os tempos de transmissões de mensagens geradas pelos streams em $hp(S_{h,j}^{ST})$ são iguais a um *slot* estático. Adicionalmente, devido ao conceito do *freeze instant*, mensagens com prioridade menor que $m_{h,j}^{ST,l}$ não conseguem bloquear mensagens do mesmo nodo que possuem prioridades mais altas, e portanto podem ser ignoradas na análise do tempo de resposta. O raciocínio que suporta esta afirmação é o seguinte: em um *freeze instant*, se $m_{h,j}^{ST,l}$ já chegou, ela tem uma prioridade mais alta que mensagens geradas por fluxos com prioridades mais baixas e será portanto transmitida. Se $m_{h,j}^{ST,l}$ não chegou, ela não será transmitida no ciclo atual, não importando se mensagens com prioridades mais baixas já chegaram ou não. Isto significa que, no contexto desta proposta, é possível ignorar o bloqueio causado por mensagens com prioridades mais baixas que $m_{h,j}^{ST,l}$, o que simplifica consideravelmente a análise.

5.3.2.3 Exemplo Ilustrativo

Em um dado sistema, um nodo N_h possui três fluxos de mensagens $S_{h,1}^{ST}$, $S_{h,2}^{ST}$ e $S_{h,3}^{ST}$. Por questões de simplicidade, todos os parâmetros serão expressos em inteiros de *gdStaticSlot*.

Os períodos de $S_{h,1}^{ST}$, $S_{h,2}^{ST}$ e $S_{h,3}^{ST}$ são, respectivamente, $P_{h,1}^{ST} = 12$, $P_{h,2}^{ST} = 21$ e $P_{h,3}^{ST} = 35$. Os quadros que contém as mensagens geradas

por $S_{h,1}^{ST}$, $S_{h,2}^{ST}$ e $S_{h,3}^{ST}$ têm tamanho unitário ($C_{h,1}^{ST} = C_{h,2}^{ST} = C_{h,3}^{ST} = 1$). O ciclo FlexRay tem tamanho $FC_{bus} = 10$, onde o ST tem tamanho $ST_{bus} = 4$ *slots*. Dois *slots* estáticos são alocados para N_h, i.e., $H_h = 2$, com $FrameID = 1$ and $FrameID = 2$. Como o *middleware* de tempo real irá despachar as mensagens de acordo com uma sequência RM, $S_{h,1}^{ST}$ tem a maior prioridade e $S_{h,3}^{ST}$ a menor. Para simplificar o exemplo, é definido que $\delta_{f_{STS_e}} = 1$.

Na Figura 36 é representado o comportamento do sistema. Existe a chegada simultânea das mensagens $m_{h,1}^{ST,1}$, $m_{h,2}^{ST,1}$ e $m_{h,3}^{ST,1}$ imediatamente após o *freeze instant* $f_{STS_2}^1$. Como no primeiro FC o *freeze instant* do último *slot* estático alocado para N_h já ocorreu, o mesmo permanece vazio naquele ciclo. No *freeze instant* seguinte ($f_{STS_1}^2$) os buffers do RTM contêm as mensagens $m_{h,1}^{ST,1}$, $m_{h,2}^{ST,1}$ e $m_{h,3}^{ST,1}$. Devido à sua prioridade relativa, no *freeze instant* $f_{STS_1}^2$ a mensagem $m_{h,1}^{ST,1}$ será selecionada para ser transferida no *slot* estático que possui $FrameID = 1$ no segundo FC, com um tempo de resposta no pior caso $R_{h,1}^{ST} = 11$. No *freeze instant* $f_{STS_2}^2$, a mensagem com a prioridade mais alta nos *buffers* do RTM é $m_{h,2}^{ST,1}$, e portanto a mesma será selecionada para ser transmitida no *slot* com $FrameID = 2$ no segundo FC, com um tempo de resposta no pior caso $R_{h,2}^{ST} = 12$. Imediatamente após o instante de tempo $t = 22$ ocorre a geração da mensagem $m_{h,1}^{ST,2}$, e os buffers do RTM passam a conter $m_{h,1}^{ST,2}$ e $m_{h,3}^{ST,1}$. Devido à sua prioridade, no *freeze instant* $f_{STS_1}^3$ $m_{h,1}^{ST,2}$ é selecionada para ser transmitida, com um tempo de resposta igual a 9. Como $R_{h,1}^{ST} = 11 \geq 9$, o valor de $R_{h,1}^{ST}$ se mantém. Finalmente, no *freeze instant* $f_{STS_2}^3$ a mensagem $m_{h,3}^{ST,1}$ é selecionada para ser transferida com um tempo de resposta no pior caso $R_{h,3}^{ST} = 22$.

Os valores para $R_{h,1}^{ST}$, $R_{h,2}^{ST}$ e $R_{h,3}^{ST}$ podem ser confirmados pela restrição de *deadline* (Equação 5.45), como se segue. Inicialmente, a porção do FC que não pode ser utilizada por N_h é modelada como sendo o fluxo de mensagens $S_{h,0}^{ST}$ com $P_{h,0}^{ST} = FC_{bus} = 10$ e $C_{h,0}^{ST} = FC_{bus} - H_h = 10 - 2 = 8$.

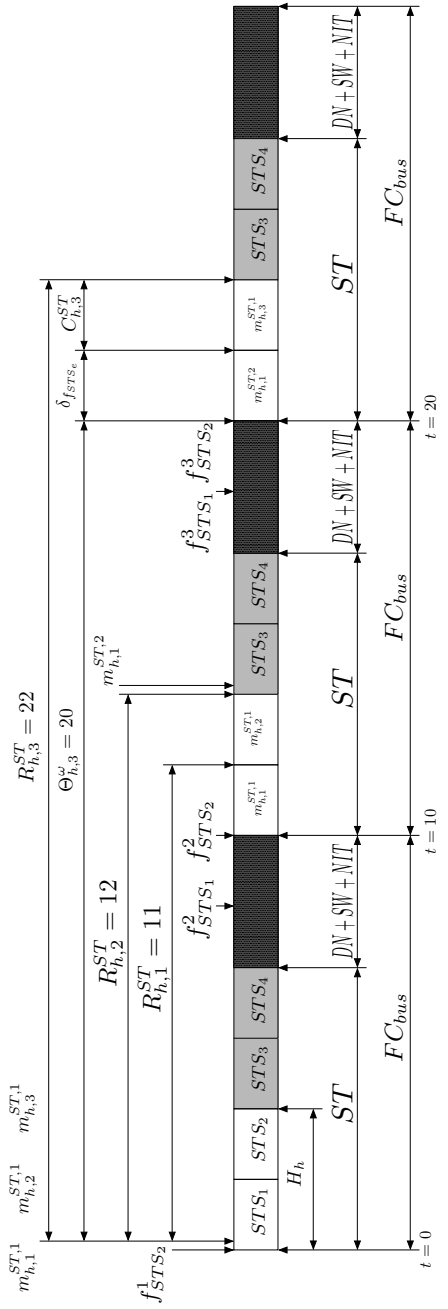

Figura 36 – Exemplo de restrição de *deadline*

Para $S_{h,1}^{ST}$ tem-se que:

$$\Theta_{h,1}^{1} = 1 + C_{h,0}^{ST} + \sum_{d=1}^{|hp(S_{h,1}^{ST})|} C_{h,d}^{ST} = 1 + 8 + 0 = 9$$
$$\Theta_{h,1}^{2} = 1 + \left\lceil \frac{9}{10} \right\rceil \times 8 = 9$$
$$R_{h,1}^{ST} = 1 + 9 + 1 = 11 \leq D_{h,1}^{ST}. \tag{5.46}$$

Da Equação 5.45, tem-se que o *deadline* de $S_{h,1}^{ST}$ é respeitado. Para $S_{h,2}^{ST}$ tem-se que:

$$\Theta_{h,2}^{1} = 1 + C_{h,0}^{ST} + \sum_{d=1}^{|hp(S_{h,2}^{ST})|} C_{h,d}^{ST} = 1 + 8 + 1 = 10$$
$$\Theta_{h,2}^{2} = 1 + \left\lceil \frac{10}{10} \right\rceil \times 8 + \left\lceil \frac{10}{12} \right\rceil \times 1 = 1 + 8 + 1 = 10$$
$$R_{h,2}^{ST} = 1 + 10 + 1 = 12 \leq D_{h,2}^{ST}. \tag{5.47}$$

Novamente, o *deadline* de $S_{h,2}^{ST}$ é respeitado ($R_{h,2}^{ST} = 12 \leq D_{h,2}^{ST}$). Por fim, tem-se para $S_{h,3}^{ST}$ que:

$$\Theta_{h,3}^{1} = 1 + C_{h,0}^{ST} + \sum_{d=1}^{|hp(S_{h,3}^{ST})|} C_{h,d}^{ST} = 1 + 8 + 1 + 1 = 11$$
$$\Theta_{h,3}^{2} = 1 + \left\lceil \frac{11}{10} \right\rceil \times 8 + \left\lceil \frac{11}{12} \right\rceil \times 1 + \left\lceil \frac{11}{23} \right\rceil \times 1 = 1 + 16 + 1 + 1 = 19$$
$$\Theta_{h,3}^{3} = 1 + \left\lceil \frac{19}{10} \right\rceil \times 8 + \left\lceil \frac{19}{12} \right\rceil \times 1 + \left\lceil \frac{19}{23} \right\rceil \times 1 = 1 + 16 + 2 + 1 = 20$$
$$\Theta_{h,3}^{4} = 1 + \left\lceil \frac{20}{10} \right\rceil \times 8 + \left\lceil \frac{20}{12} \right\rceil \times 1 + \left\lceil \frac{20}{23} \right\rceil \times 1 = 1 + 16 + 2 + 1 = 20$$
$$R_{h,3}^{ST} = 1 + 20 + 1 = 22 \leq D_{h,3}^{ST}. \tag{5.48}$$

O *deadline* de $S_{h,3}^{ST}$ também é respeitado, o que significa que a restrição de *deadline* é respeitada. Este resultado é consistente com os valores que podem ser observados na Figura 36.

5.3.3 Heurística Para Definição da Alocação de *Slots* Estáticos

Finalmente, existe a necessidade de se definir uma alocação de *slots* adequada para cada nodo, garantindo assim que os requisitos temporais de um sistema FlexRay são respeitados.

Nesta seção será proposta uma heurística para derivar a alocação H_h de um nodo chamada *Adaptive Proportional Allocation Scheme* (APAS). A heurística APAS preenche os requisitos de ambas as restrições de protocolo (Equação 5.39) e de *deadline* (Equação 5.45).

O algoritmo APAS (Algoritmo 2) é uma melhoria do Algoritmo PAS proposto na Seção 5.2.3.

O APAS inicia definindo um valor inicial para FC_{bus} utilizando a Equação 5.10 (passo 1, linha 2) e uma alocação de *slots* estáticos H_h para cada nodo N_h utilizando a Equação 5.49 (passo 2, linhas 4 a 6).

$$H_h = \left\lceil \sum_{j=1}^{|\mathbb{S}_h^{ST}|} \frac{FC_{bus}}{P_{h,j}^{ST}} \right\rceil . \tag{5.49}$$

A racionalização por trás da Equação 5.49 é a seguinte: considere que uma mensagem do fluxo $S_{h,j}^{ST}$ é fragmentada e que os fragmentos resultantes estão espalhados entre os *slots* estáticos durante o período $P_{h,j}^{ST}$. $FC_{bus}/P_{h,j}^{ST}$ representa então o percentual de utilização do *slot* durante cada FC. Considere agora o conjunto de fluxos de mensagens $\mathbb{S}_h^{ST}$ relacionado ao nodo N_h. A Equação 5.49 representa o número de *slots* requeridos para servir todos os fluxos de mensagens do nodo N_h. Assim, a equação representa o número de *slots* estáticos que devem ser alocados para N_h de forma a garantir que o *deadline* de todas as mensagens geradas no nodo sejam respeitados.

No passo 3, compreendido entre as linhas 8 e 17, a restrição de *deadline* (Equação 5.45) é verificada para cada nodo do sistema. Se a restrição de *deadline* não é respeitada para um dado nodo, a alocação H_h deste nodo é incrementada de um *slot* estático (linha 10), o laço é executado outra vez (linha 15) e a restrição de *deadline* é verificada novamente (linha 8). Este processo é repetido até que a restrição de *deadline* seja respeitada para o nodo, ou até que H_h exceda o tamanho do ST. No caso da alocação H_h exceder ST_{bus} (linhas 11 a 14), é definido um novo valor para FC_{bus}, e o algoritmo é executado novamente a partir da linha 5.49 (linha 13). A racionalização por trás da redução do tamanho do FC é: com um FC menor, o intervalo entre uma alocação

Algoritmo 2 Adaptive Proportional Allocation Scheme

Entrada: $\mathbb{S}$

1: /* Passo 1 */
2: Definir um valor inicial para FC_{bus} com a maior duração possível usando a Equação 5.10
3: /* Passo 2 */
4: **for** h:= 1 to $|\mathbb{N}|$ **do**
5: Definir uma alocação de *slots* estáticos H_h inicial para o nodo N_h utilizando a Equação 5.49
6: **end for**
7: /* Passo 3 */
8: **for** h:= 1 to $|\mathbb{N}|$ **do**
9: **if** A restrição de *deadline* (Equação 5.45) não é respeitada para o nodo N_h **then**
10: Aumentar a alocação de *slots* para o nodo h: $H_h = H_h + 1$
11: **if** $H_h > FC_{bus}$ **then**
12: Diminuir FC_{bus}: $FC_{bus} = FC_{bus} - 1$
13: Ir para linha 4
14: **end if**
15: Ir para linha 8
16: **end if**
17: **end for**
18: /* Passo 4 */
19: **if** A restrição de protocolo (Equação 5.39) não é respeitada **then**
20: Diminuir FC_{bus}: $FC_{bus} = FC_{bus} - 1$
21: **if** $FC_{bus} < |\mathbb{N}|$ **then**
22: **Retornar:** falha
23: **end if**
24: Ir para linha 4
25: **end if**
26: /* Passo 5 */
27: **Retornar:** sucesso

H_j e a alocação seguinte é menor, oferecendo assim mais oportunidades para a transmissão das mensagens geradas pelos fluxos do nodo H_h.

No passo 4 (linhas 19 a 25) é verificada a restrição de protocolo (Equação 5.39). Se a restrição não é respeitada, é definido um novo valor para FC_{bus} reduzindo-se o mesmo em um *slot*, sendo então o algoritmo executado novamente a partir do passo 2 (linha 4). O laço é repetido até que seja encontrado um sistema escalonável, caso no qual o algoritmo termina, retornando sucesso (passo 5, linha 27). Entretanto, se o novo valor de FC_{bus} for menor que $|\mathbb{N}|$ (caso em que não existirá no mínimo um *slot* estático para cada nodo), o algoritmo é encerrado retornando a indicação de que o sistema não é escalonável, já que o limite inferior que a especificação do FlexRay impõe sobre FC_{bus} não é respeitado (linhas 21 e 22).

5.3.3.1 Exemplo Ilustrativo

Considere um sistema FlexRay composto por 3 nodos, sendo $\delta_{f_{max}} = 1$. Cada nodo possui um conjunto de fluxos de mensagens cujos períodos são apresentados na Tabela 6. A soma dos tamanhos dos segmentos de controle é $\theta = 1$. Considere também que, para cada fluxo do sistema, $D^{ST}_{h,j} = P^{ST}_{h,j}$. Todos os valores são dados em múltiplos inteiros de $gdStaticSlot$.

Nodo N_1	$\boldsymbol{P^{ST}_{1,1}}$	$\boldsymbol{P^{ST}_{1,2}}$	$\boldsymbol{P^{ST}_{1,3}}$	$\boldsymbol{P^{ST}_{1,4}}$	
	12	15	29	50	
Nodo N_2	$\boldsymbol{P^{ST}_{2,1}}$	$\boldsymbol{P^{ST}_{2,2}}$	$\boldsymbol{P^{ST}_{2,3}}$		
	23	33	100		
Nodo N_3	$\boldsymbol{P^{ST}_{3,1}}$	$\boldsymbol{P^{ST}_{3,2}}$	$\boldsymbol{P^{ST}_{3,3}}$	$\boldsymbol{P^{ST}_{3,4}}$	$\boldsymbol{P^{ST}_{3,5}}$
	12	23	29	37	44

Tabela 6 – Conjunto de Fluxos de Mensagens para o Exemplo Ilustrativo

De acordo com o Algoritmo 2, o segmento Estático pode ser definido como se segue:

1) Definir um valor inicial para FC_{bus} com a maior duração possível (Equação 5.10):

$$\begin{aligned} FC_{bus} &\leq P^{ST}_{min} - (1 + \delta_{f_{max}}) \\ FC_{bus} &\leq 12 - (1 + 1) = 10. \end{aligned} \tag{5.50}$$

2) Para todos os nodos N_h no sistema, definir uma alocação H_h inicial utilizando a Equação 5.49:

$$H_1 = \left\lceil \frac{10}{12} + \frac{10}{15} + \frac{10}{29} + \frac{10}{50} \right\rceil = 3 \tag{5.51}$$

$$H_2 = \left\lceil \frac{10}{23} + \frac{10}{33} + \frac{10}{100} \right\rceil = 1 \tag{5.52}$$

$$H_3 = \left\lceil \frac{10}{12} + \frac{10}{23} + \frac{10}{29} + \frac{10}{37} + \frac{10}{44} \right\rceil = 3. \tag{5.53}$$

3) Para todos os nodos do sistema, verificar se a restrição de *deadline* é respeitada (Equação 5.45). Por questões de simplicidade, não serão repetidos os cálculos que levam ao tempo de transmissão no pior caso de cada fluxo de mensagens. Na Tabela 7 são resumidos estes resultados para cada fluxo de mensagens do sistema. A tabela também apresenta o *deadline* de cada fluxo e se a restrição de deadline foi respeitada ou não.

	$\boldsymbol{R^{ST}_{1,j}}$	$\boldsymbol{D^{ST}_{1,j}}$	$\boldsymbol{R^{ST}_{1,j} \leq D^{ST}_{1,j}}$
Nodo N_1	$R^{ST}_{1,1} = 12$	$D^{ST}_{1,1} = 12$	$R^{ST}_{1,1} \leq D^{ST}_{1,1}$ ok
	$R^{ST}_{1,2} = 13$	$D^{ST}_{1,2} = 15$	$R^{ST}_{1,2} \leq D^{ST}_{1,2}$ ok
	$R^{ST}_{1,3} = 14$	$D^{ST}_{1,3} = 29$	$R^{ST}_{1,3} \leq D^{ST}_{1,3}$ ok
	$R^{ST}_{1,4} = 24$	$D^{ST}_{1,4} = 50$	$R^{ST}_{1,4} \leq D^{ST}_{1,4}$ ok
	$\boldsymbol{R^{ST}_{2,j}}$	$\boldsymbol{D^{ST}_{2,j}}$	$\boldsymbol{R^{ST}_{2,j} \leq D^{ST}_{2,j}}$
Nodo N_2	$R^{ST}_{2,1} = 12$	$D^{ST}_{2,1} = 23$	$R^{ST}_{2,1} \leq D^{ST}_{2,1}$ ok
	$R^{ST}_{2,2} = 22$	$D^{ST}_{2,2} = 23$	$R^{ST}_{2,2} \leq D^{ST}_{2,2}$ ok
	$R^{ST}_{2,3} = 62$	$D^{ST}_{2,3} = 100$	$R^{ST}_{2,3} \leq D^{ST}_{2,3}$ ok
	$\boldsymbol{R^{ST}_{3,j}}$	$\boldsymbol{D^{ST}_{3,j}}$	$\boldsymbol{R^{ST}_{3,j} \leq D^{ST}_{3,j}}$
Nodo N_3	$R^{ST}_{3,1} = 12$	$D^{ST}_{3,1} = 23$	$R^{ST}_{3,1} \leq D^{ST}_{3,1}$ ok
	$R^{ST}_{3,2} = 22$	$D^{ST}_{3,2} = 23$	$R^{ST}_{3,2} \leq D^{ST}_{3,2}$ ok
	$R^{ST}_{3,3} = 62$	$D^{ST}_{3,3} = 100$	$R^{ST}_{3,3} \leq D^{ST}_{3,3}$ ok
	$R^{ST}_{3,4} = 22$	$D^{ST}_{3,4} = 23$	$R^{ST}_{3,4} \leq D^{ST}_{3,4}$ ok
	$R^{ST}_{3,5} = 62$	$D^{ST}_{3,5} = 100$	$R^{ST}_{3,5} \leq D^{ST}_{3,5}$ ok

Tabela 7 – Resumo da restrição de *deadline* para o Exemplo Ilustrativo

4) Verificar se a restrição de protocolo é respeitada (Equação 5.39):

$$\sum_{h=1}^{|\mathbb{N}|} H_h + \theta \leq FC_{bus} \leq P_{min}^{ST} - (1 + \delta_{f_{max}})$$
$$3 + 1 + 3 + 1 \leq 10 \leq 12 - (1 + 1)$$
$$8 \leq 10 \leq 10. \tag{5.54}$$

Como pode ser observado na Equação 5.54, a restrição de protocolo é respeitada para a alocação de *slots* estáticos que está sendo avaliada.

Como ambas as restrições de *deadline* e de protocolo são respeitadas, o *Adaptive Proportional Allocation Scheme* gerou uma solução escalonável para o sistema FlexRay.

5.3.4 Avaliação Experimental do Método de Alocação Proposto

Para avaliar o *Adaptive Proportional Allocation Scheme* proposto, seu comportamento será comparado, através de experimentos, com uma abordagem do estado da arte relacionado ao ST. Publicações recentes que abordam o escalonamento do segmento Estático do FlexRay propõem técnicas que associam um fluxo de mensagens com um identificador de *slot* exclusivo e com uma repetição de ciclo. Exemplos de tais publicações são (SCHMIDT; SCHMIDT, 2010a), (KANG et al., 2012) e (HANZáLEK et al., 2011). Para esta avaliação comparativa foi escolhido como *benchmark* o método descrito em (SCHMIDT; SCHMIDT, 2010a). A metodologia proposta em (SCHMIDT; SCHMIDT, 2010a) utiliza ILP para definir uma associação ótima entre um fluxo de mensagens, um $FrameID$ único e uma repetição de ciclos. São comparados o número de sistemas considerados escalonáveis por ambos (*benchmark* e APAS) e, quando um sistema é considerado escalonável, também são comparados o número de *slots* utilizados em cada método.

5.3.4.1 Condições Para os Experimentos

A metodologia descrita em (SCHMIDT; SCHMIDT, 2010a) assume que os parâmetros FlexRay, tais como a duração do ciclo FlexRay, o tamanho de cada segmento e o tamanho dos *slots* estáticos, são fornecidos e não estão sujeitos a otimização. Para que seja possível uma

comparação justa, não será executado o passo 1 do Algoritmo 2, e tais parâmetros serão utilizados como entradas do mesmo. Também não será executado o passo 4 do Algoritmo 2: se a restrição de protocolo não for respeitada, o algoritmo irá retornar falha em vez de decrementar FC_{bus}.

Nos experimentos simulados, serão considerados os parâmetros FlexRay utilizados pela BMW (SCHEDL, 2007). Estes parâmetros são apresentados na Tabela 8.

FC_{bus}	5 ms
ST_{bus}	3 ms
$gNumberOfStaticSlots$	91 static slots
$gdStaticSlot$	32.967 μs

Tabela 8 – Parâmetros para o sistema FlexRay. Baseado em (SCHEDL, 2007)

O NIT e o SW serão considerados parte do DN. Da Tabela 8 e das Equações 5.11 e 5.12 segue-se que:

$$\begin{aligned} FC_{bus} &= ST_{bus} + DN_{bus} + \theta \\ DN_{bus} + \theta &= 5ms - 3ms \\ \theta &= 2ms. \end{aligned} \tag{5.55}$$

Nos experimentos serão considerados sistemas com 15 ECUs interconectadas por um barramento FlexRay com um canal de comunicação operando a 10Mb/s.

Ambas as metodologias serão avaliadas sob utilizações de rede correspondentes a 10, 20, 30, 40, 50 e 60%. Deve-se salientar que, devido à natureza periódica do ciclo FlexRay, o DN e os segmentos de controle representam uma carga de 40%. Para cada grupo de parâmetros e para cada nível de utilização, foram gerados 100 sistemas utilizando-se a ferramenta Netcarbench. O Netcarbench (BRAUN et al., 2007) é um *software* livre que gera conjuntos de fluxos de mensagens para sistemas automotivos com base em um conjunto de parâmetros definidos pelo usuário. O Netcarbench recebe a utilização de rede desejada, os períodos dos streams de mensagens e o tamanho do *slot* estático como parâmetro.

A metodologia proposta em (SCHMIDT; SCHMIDT, 2010a) pode ser configurada para minimizar tanto o número de *slots* estáticos alocados quanto o *jitter* de resposta. Como o método proposto nessa seção

não utiliza *jitter* como parâmetro, foi feita a escolha de minimizar apenas o número de *slots* alocados para cada nodo.

Para os experimentos, a metodologia proposta em (SCHMIDT; SCHMIDT, 2010a) foi implementada utilizando-se o GLPK versão 4.47 (GLPK, 2014). As restrições que compõem tal proposta foram obtidas no artigo e mapeadas para a linguagem de entrada do GLPK. Cabe ressaltar que tais restrições são descritas no artigo em uma linguagem semelhante à linguagem de programação do GLPK. Já o APAS foi implementado como uma biblioteca C.

5.3.4.2 Comparação do APAS com a Técnica do Estado da Arte

Diferente da proposta apresentada nesta seção, a metodologia proposta em (SCHMIDT; SCHMIDT, 2010a) assume que os períodos dos fluxos de mensagens devem ser múltiplos inteiros de FC_{bus}, e que a geração de mensagens pelos fluxos é sincronizada com o ciclo FlexRay. Devido a este fato, foram realizados experimentos simulados para dois cenários. Para o primeiro cenário, assumiu-se que os períodos dos fluxos de mensagens não são múltiplos inteiros de FC_{bus} e que a geração de mensagens não está sincronizada com o FC. Para o segundo cenário, assumiu-se que os períodos dos fluxos de mensagens são múltiplos inteiros de FC e que a geração de mensagens está sincronizada com o ciclo.

A seguir, serão descritos cada cenário e apresentados os resultados das simulações.

5.3.4.3 Cenário 1: os períodos dos fluxos de mensagens não são múltiplos de FC, e a geração de mensagens não está sincronizada com o ciclo FlexRay

O primeiro cenário considera que os períodos dos fluxos de mensagens não são inteiros de FC_{bus}, e que a geração de mensagens não é sincronizada com o FC. Para este cenário, foram considerados períodos equivalentes a 11, 23, 59, 104, 133, 214, 501 e 1002 ms.

Para respeitar a restrição sobre os períodos dos fluxos e geração de mensagens que assumida em (SCHMIDT; SCHMIDT, 2010a), para o primeiro cenário foi definido um novo conjunto de fluxos de mensagens $\mathbb{S}^{ST\prime}$ cujos períodos e *deadlines* foram reduzidos de acordo com as Equações 5.56 e 5.57. O novo conjunto $\mathbb{S}^{ST\prime}$ foi utilizado como entrada

na metodologia proposta em (SCHMIDT; SCHMIDT, 2010a). Esta redução é necessária devido ao já citado fato de que a metodologia proposta em (SCHMIDT; SCHMIDT, 2010a) assume que a) existe uma forte sincronização entre a geração de mensagens pelos fluxos e o ciclo FlexRay e que b) os períodos dos fluxos são múltiplos inteiros de FC_{bus}.

$$P_{h,j}^{ST\prime} = \lfloor P_{h,j}^{ST} / FC_{bus} \rfloor \times FC_{bus},\ \forall S_{h,j}^{ST} \in \mathbb{S}^{ST} \tag{5.56}$$

$$D_{h,j}^{ST\prime} = P_{h,j}^{ST\prime},\ \forall S_{h,j}^{ST} \in \mathbb{S}^{ST}. \tag{5.57}$$

Na Figura 37 é apresentado o número de sistemas considerados escalonáveis por ambos o APAS e o *benchmark*. Como pode ser observado, a efetividade do APAS é mais alta que a do *benchmark*. Na Figura 38 são mostradas as alocações de *slots* para o primeiro cenário. Na figura são exibidas a média e a faixa das alocações. Pode-se observar que o APAS aloca um número menor de *slots* que o benchmark, e que a diferença no número de *slots* aumenta de acordo com o aumento da carga na rede.

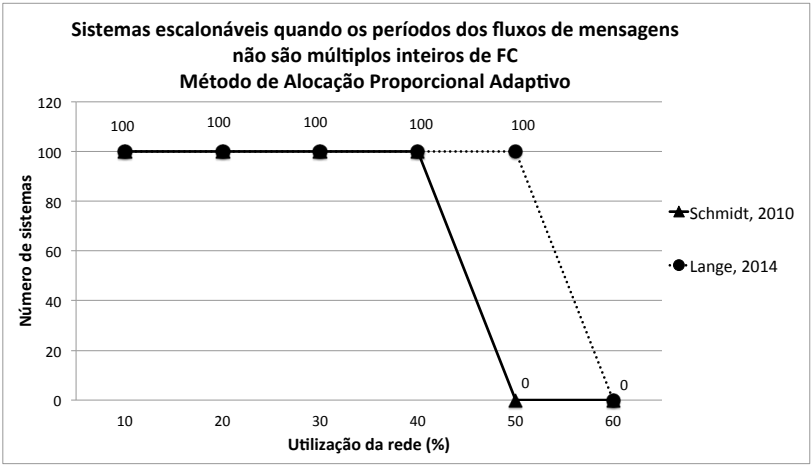

Figura 37 – Resultados para o cenário 1: número de sistemas escalonáveis

Quando analisando as Figuras 37 e 38 fica claro que, para o caso onde não existe sincronização entre a geração de mensagens e o FC, e onde os períodos dos fluxos não são múltiplos de FC, o APAS é mais eficaz e requer um número menor de *slots* para um mesmo conjunto de

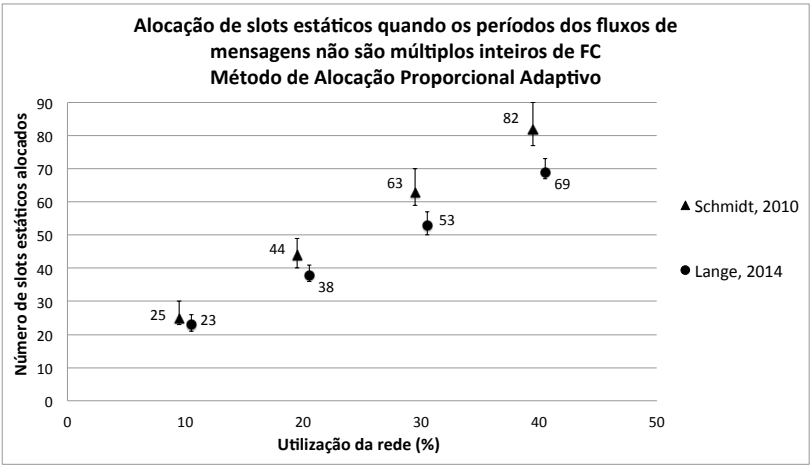

Figura 38 – Resultados para o cenário 1: Alocação de *slots* estáticos

fluxos.

5.3.4.4 Cenário 2: Períodos dos fluxos são múltiplos inteiros de FC

Para o segundo cenário, assumiu-se que os períodos dos fluxos de mensagens são múltiplos inteiros de FC_{bus}, com valores de 10, 20, 25, 100, 155, 200, 500 e 1000 ms. Como os períodos dos fluxos são múltiplos inteiros de FC_{bus}, a restrição sobre os períodos e a geração de mensagens que é assumida no *benchmark* é respeitada, não sendo necessárias modificações nos valores dos períodos.

Na Figura 39 são exibidos os números de sistemas considerados escalonáveis por cada método, e na Figura 40 são exibidos a média e a faixa da alocação de *slots* para este cenário. Quando analisando ambas as Figuras 39 e 40, pode-se observar que o APAS aloca um número ligeiramente menor de *slots* estáticos e, como consequência, o APAS é ligeiramente mais efetivo que o *benchmark* para o caso em que os períodos dos fluxos de mensagens são múltiplos inteiros de FC.

Este resultado pode ser explicado pela natureza do método proposto em (SCHMIDT; SCHMIDT, 2010a): apesar do método utilizar técnicas de otimização e definir uma alocação de *slots* estáticos considerada otimizada para o conjunto de fluxos de mensagens recebido como entrada, o mesmo deve respeitar a restrição imposta pelo parâmetro

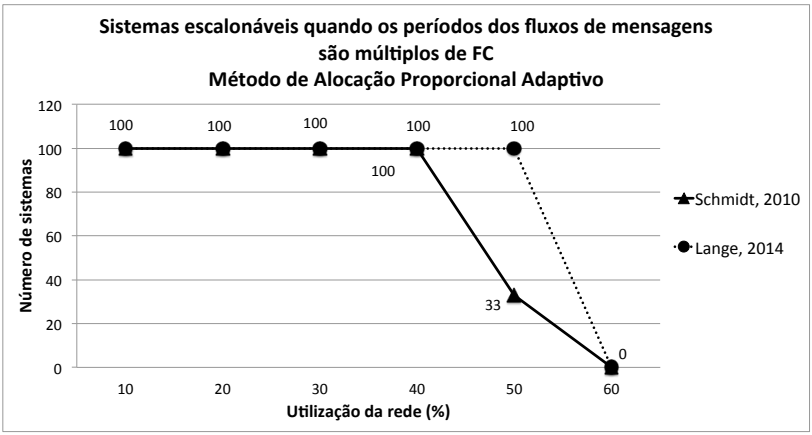

Figura 39 – Resultados para o cenário 2: número de sistemas escalonáveis

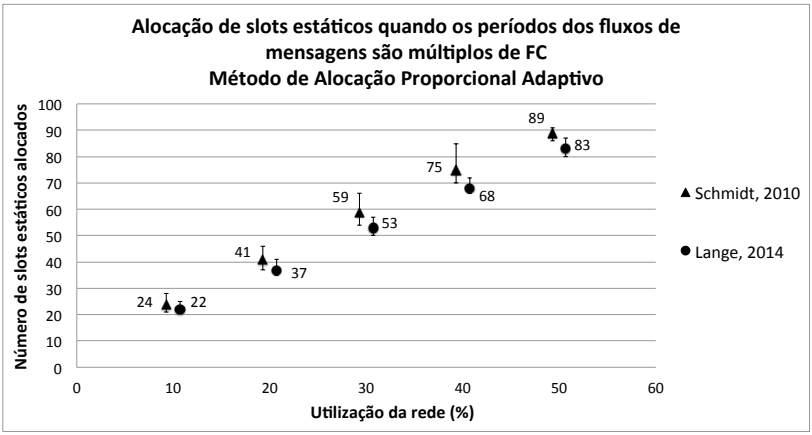

Figura 40 – Resultados para o cenário 2: alocação de *slots* estáticos

relacionado à repetição de ciclos. Como esta restrição não existe no APAS, o padrão de transmissões é diferente, sendo o APAS mais flexível em relação à utilização dos *slots* estáticos alocados.

5.4 CONTRIBUIÇÕES AO ESTADO DA ARTE

Como identificado no Capítulo 3, as propostas da literatura que abordam o projeto do segmento Estático do FlexRay são geralmente baseadas em técnicas que requerem grande esforço computacional, como por exemplo, ILP. E usualmente estas propostas assumem que os fluxos de mensagens dos nodos do sistema estão associados com *slots* estáticos exclusivos, sendo que a especificação do FlexRay define apenas que a associação deve ser exclusiva para *slots* estáticos e nodos do sistema.

Neste capítulo é apresentada uma nova abordagem para o escalonamento do segmento Estático do FlexRay. Na descrição desta nova abordagem, foi demonstrada a viabilidade de se definir a alocação de *slots* estáticos para cada nodo utilizando técnicas tradicionais para a análise de tempo de resposta considerando-se os requisitos temporais impostos pelo conjunto de fluxos de mensagens de cada nodo.

A abordagem proposta inclui dois métodos que utilizam as técnicas tradicionais para a análise de tempo de resposta na definição da alocação de *slots* estáticos para nodos FlexRay. Tais métodos são chamados, respectivamente, de Método de Alocação Proporcional e Método de Alocação Proporcional Adaptativo. Ambos são capazes de considerar conjuntos de fluxos com períodos que não são múltiplos de FC, sendo também capazes de considerar o caso em que a geração de mensagens nos fluxos não está sincronizada com o FC.

No capítulo foi explicado como as técnicas propostas podem ser integradas em um *middleware* de tempo real em cada nodo, tirando vantagem dos métodos de alocação propostos. Também foi explicado como o Método de Alocação Proporcional proposto pode ser integrado em um ambiente AUTOSAR permitindo seu uso em implementações FlexRay reais, e apresentada uma comparação entre o Método de Alocação Proporcional Adaptativo e uma das técnicas do estado da arte sobre escalonamento no segmento Estático do FlexRay que utiliza ILP. Tal comparação demonstra que o APAS é mais eficaz e requer um número menor de *slots* para um mesmo conjunto de fluxos do que o método do estado da arte, o que pode ser explicado pelo fato de que o padrão de transmissões é diferente, sendo o APAS mais flexível em relação à utilização dos *slots* estáticos alocados.

Uma questão que permanece em aberto é a análise do custo computacional do *middleware* de tempo real necessário para esta proposta, e seu impacto em uma implementação real. Esta questão também pode vir a ser explorada como uma extensão deste trabalho.

6 ESCALONAMENTO PROBABILÍSTICO DO SEGMENTO DINÂMICO DO FLEXRAY

O Sistema de Comunicação FlexRay, descrito no Capítulo 2, utiliza um método de controle de acesso ao meio baseado em um ciclo de comunicação composto por quatro segmentos. Dois destes segmentos, NIT e SW, são relacionados a operações de controle do protocolo como, por exemplo, sincronização de relógios. O Segmento Estático de um ciclo de comunicação é o componente *time-triggered* do FlexRay, e é projetado para acomodar a transmissão de mensagens geradas por fluxos de mensagens periódicos que possuem características de tempo real crítico.

O Segmento Dinâmico (DN) é o componente *event-triggered* do FlexRay. É projetado para acomodar a transmissão de mensagens geradas por fluxos esporádicos que possuem requisitos rígidos de tempo real, fluxos esporádicos cujos requisitos de tempo real não são críticos, e fluxos aperiódicos que não possuem requisitos de tempo real (NAVET; SIMONOT-LION, 2008; KHALGUI et al., 2013). Os *slots* do segmento Dinâmico podem ter tamanhos variados, e a arbitragem no segmento é baseada na prioridade dos fluxos que transmitem nesse segmento (SCHMIDT; SCHMIDT, 2010a; ZENG et al., 2011; KANG et al., 2012; TANASA et al., 2013). Devido às características do Segmento Dinâmico, fluxos de mensagens associados com *slots* de baixa prioridade possuem uma probabilidade de transmissão mais baixa que os fluxos de mensagens associados com *slots* de prioridade alta. Como consequência, se o DN estiver sobrecarregado, a transmissão de mensagens com baixa prioridade pode ser postergada indefinidamente.

Apesar do DN ser projetado para acomodar a transmissão de mensagens geradas por fluxos esporádicos com características de tempo real, ou por fluxos aperiódicos que não possuem restrições temporais, como mostra o levantamento do estado da arte apresentado no Capítulo 3, Seção 3.2, página 63, a maioria dos trabalhos que abordam o escalonamento e a análise temporal do DN assume um modelo de sistema onde as aplicações geram sinais que são empacotados em mensagens com características de tempo real crítico (por exemplo, (SCHMIDT; SCHMIDT, 2010b; ZENG et al., 2011; KANG et al., 2012)). Exceções são as propostas apresentadas por Kim e Park (2009) e Tanasa et al. (2013). Mas o foco de tais propostas é a obtenção da taxa de perda de *deadlines* (*deadline miss ratio*) dos fluxos de mensagens, sem considerar problemas relacionados ao escalonamento dos fluxos de mensagens.

Entretanto, um sistema veicular também possui aplicações que possuem características de tempo real não crítico ou aplicações que não possuem restrições temporais. Por exemplo, fluxos de mensagens associados com sistemas de manutenção são geralmente aperiódicos. Escalonar fluxos de mensagens aperiódicos assumindo artificialmente que os mesmos possuem características temporais rígidas pode resultar em subutilização da rede, já que é necessário garantir que as restrições temporais de tais fluxos sejam respeitadas através da alocação de recursos para os mesmos.

Neste capítulo é abordada a questão do escalonamento, no Segmento Dinâmico do FlexRay, de fluxos de mensagens aperiódicos que possuem características temporais não críticas ou que não possuem restrições temporais. São apresentadas duas propostas que tiram vantagem da flexibilidade que fluxos aperiódicos possuem em relação a restrições de tempo real. Em ambas as propostas considera-se que mensagens geradas por fluxos que possuem características críticas de tempo real (*hard real-time*) são transmitidas ou no Segmento Estático do FlexRay, ou com identificadores de alta prioridade do Segmento Dinâmico. Já a transmissão de mensagens geradas por fluxos aperiódicos, ou por fluxos que não possuem características críticas de tempo real (*soft real-time*), é coordenada utilizando-se um mecanismo de *backoff* probabilístico.

Nos mecanismos propostos, os fluxos de mensagens aperiódicos são associados com uma *probabilidade de backoff*. Sempre que o sistema detecta que o DN está sobrecarregado e que, como consequência, mensagens estão sendo postergadas indefinidamente, o mesmo entra em um modo especial de arbitragem chamado de *modo backoff*. No modo *backoff*, a probabilidade de *backoff* de cada fluxo aperiódico é utilizada para definir se uma mensagem irá competir ou não pelo barramento no ciclo de comunicação atual.

A primeira proposta, chamada "Método de Escalonamento Probabilístico para o Segmento Dinâmico do FlexRay", utiliza um mecanismo de arbitragem que, quando o DN está sobrecarregado, muda simultaneamente todos os nodos para o modo *backoff*.

Já a segunda proposta, chamada "Método de Escalonamento Probabilístico do Segmento Dinâmico do FlexRay Descentralizado", utiliza um mecanismo descentralizado. Neste mecanismo, cada fluxo de mensagens está associado a um *middleware* de tempo real. Com base nos parâmetros do fluxo de mensagens ao qual está associado, um *middleware* de tempo real consegue identificar se o fluxo está enviando rajadas de mensagens. Caso o fluxo esteja enviando rajadas, o *middleware* entra em modo *backoff* por um período de tempo pré-determinado con-

tribuindo, desta forma, para que as mensagens geradas por fluxos com baixas prioridades não sejam postergadas indefinidamente.

No restante deste capítulo será primeiramente apresentado o modelo de sistema considerado (Seção 6.1), seguido da apresentação de ambas as propostas (Seções 6.2 e 6.3). A apresentação de cada proposta contém uma descrição detalhada de cada método, exemplos numéricos ilustrando seu uso e a apresentação de um experimento simulado. Por fim, na Seção 6.4 são feitas as considerações finais sobre o capítulo.

As propostas apresentadas neste capítulo são originais, e uma versão preliminar foi parcialmente publicada em (LANGE; OLIVEIRA, 2014).

6.1 MODELO DE SISTEMA

As propostas apresentadas neste capítulo utilizam o modelo de sistema apresentado no Capítulo 4. Para facilitar o entendimento do texto, os pontos daquele capítulo que são relevantes para esta proposta são resumidos a seguir.

Considera-se um sistema composto por $|\mathbb{N}|$ nodos interconectados através de um barramento FlexRay com um canal de comunicação. Cada nodo N_h ($h = 1, 2, ..., |\mathbb{N}|$) é composto por uma CPU e um controlador de comunicação FlexRay (CC). A operação do CC é independente da CPU. Considera-se que o CC de um nodo possui um *buffer* de entrada para cada *slot* dinâmico que lhe foi atribuído. Seguindo a linha dos demais trabalhos da literatura sobre o FlexRay (discutidos no Capítulo 3), esta proposta não considera o uso do segundo canal de comunicação, mas este poderia ser utilizado, por exemplo, para fins de redundância na rede, replicando as configurações definidas para o primeiro canal.

O Segmento Estático do FlexRay não é abordado neste capítulo, mas seguindo a especificação do FlexRay (FLEXRAY, 2015), considera-se que cada nodo N_h é associado com, no mínimo, um *slot* do ST.

Para as propostas que serão apresentadas a seguir, interessam fluxos de mensagens aperiódicos ou esporádicos sem características críticas de tempo real cujas mensagens são transmitidas no Segmento Dinâmico de um barramento FlexRay. Segundo Liu (2000), a demora na transmissão de uma mensagem gerada por tais fluxos é indesejável, mas tolerável.

Define-se que cada nodo N_h conectado a um barramento FlexRay possui um conjunto $\mathbb{S}_h^{ap}$ de fluxos de mensagens aperiódicos e/ou

esporádicos sem características críticas de tempo real. $\mathbb{S}_h^{ap}$ possui $|\mathbb{S}_h^{ap}|$ fluxos de mensagens.

O conjunto de todos os fluxos de mensagens aperiódicos e/ou esporádicos sem características críticas de tempo real de um dado sistema é dado por $\mathbb{S}^{ap}$. Portanto,

$$\mathbb{S}^{ap} = \mathbb{S}_1^{ap} \cup \mathbb{S}_2^{ap} \cup ... \cup \mathbb{S}_{|\mathbb{S}_h^{ap}|}^{ap}. \tag{6.1}$$

Para simplificar a leitura, no restante deste capítulo os tipos de fluxos que compõem $\mathbb{S}^{ap}$ serão chamados simplesmente de "fluxos de mensagens aperiódicos" ou simplesmente "fluxos aperiódicos".

Cada fluxo de mensagens $S_{h,j}^{ap} \in \mathbb{S}^{ap}$ é definido por uma tupla $\{FID_{h,j}^{ap}, C_{h,j}^{ap}, pLatestTx_{h,j}^{ap}\}$ onde:

- $h = 1, 2, ..., |\mathbb{N}|$ denota o número do nodo;
- $j = 1, 2, ..., |\mathbb{S}_h^{ST}|$ é o identificador do fluxo de mensagens no nodo;
- $FID_{h,j}^{ap}$ é o $FrameID$ de $S_{h,j}^{ap}$;
- $C_{h,j}^{ap}$ é o tamanho em segundos de $S_{h,j}^{ap}$, calculado utilizando-se a Equação 2.3, apresentada no Capítulo 2, Seção 2.1.1.3, página 50;
- $pLatestTx_{h,j}^{ap}$ é o $pLatestTx$ de $S_{h,j}^{ap}$, calculado utilizando-se a Equação 2.1, apresentada no Capítulo 2, Seção 2.1.1.3, página 47;

Cada fluxo de mensagens aperiódico gera uma sequência infinita de mensagens. A l-ésima mensagem gerada por $S_{h,j}^{ap}$ é representada por $m_{h,j}^{ap,l}$, onde l é o número da mensagem.

Neste capítulo considera-se que o $FrameID$ de cada fluxo de mensagem em $\mathbb{S}^{ap}$ é arbitrariamente definido, existindo porém a restrição de que o $FrameID$ de um fluxo deve permitir que este respeite seu $pLatestTx$.

Por fim, o conjunto de fluxos de mensagens com prioridades mais altas que um fluxo $S_{h,j}^{ap}$ é representado por $hp(S_{h,j}^{ap})$.

6.2 MÉTODO DE ESCALONAMENTO PROBABILÍSTICO PARA O SEGMENTO DINÂMICO DO FLEXRAY

Nesta seção será apresentada a primeira proposta para um método probabilístico para o escalonamento de fluxos de mensagens ape-

riódicos no Segmento Dinâmico do FlexRay.

O método proposto utiliza um mecanismo de *backoff* probabilístico. Este mecanismo de *backoff* considera que cada fluxo de mensagem aperiódico $S^{ap}_{h,j} \in \mathbb{S}^{ap}$, além de estar associado com a tupla $\{FID^{ap}_{h,j}, C^{ap}_{h,j}, pLatestTx^{ap}_{h,j}\}$, também está associado com dois valores probabilísticos: $\overline{\mathcal{P}^{bk}_{h,j}}$ e $\mathcal{P}^{bk}_{h,j}$. Estes valores representam, respectivamente, a probabilidade de uma mensagem gerada por $S^{ap}_{h,j}$ ser selecionada ou não para transmissão em um dado *slot* do segmento Dinâmico quando o sistema está em modo *backoff*. Em outras palavras, estes valores representam a *probabilidade de backoff* de $S^{ap}_{h,j}$. Na proposta apresentada nesta seção, considera-se que o valor de $\overline{\mathcal{P}^{bk}_{h,j}}$ é arbitrário, e o valor de $\mathcal{P}^{bk}_{h,j}$ é dado por $\mathcal{P}^{bk}_{h,j} = 1 - \overline{\mathcal{P}^{bk}_{h,j}}$. Nesta seção não será abordado o problema da definição de valores ótimos para $\overline{\mathcal{P}^{bk}_{h,j}}$ e $\mathcal{P}^{bk}_{h,j}$.

Além das probabilidades de *backoff* de cada fluxo, o mecanismo de *backoff* proposto também utiliza um *middleware* de tempo real (*Real-Time Midleware*, RTM) para coordenar a transmissão de mensagens. O RTM é implementado na CPU de cada nodo de rede e contém um *buffer* FIFO $B_{h,j}$ para cada fluxo de mensagens $S^{ap}_{h,j}$. O RTM é ilustrado na Figura 41.

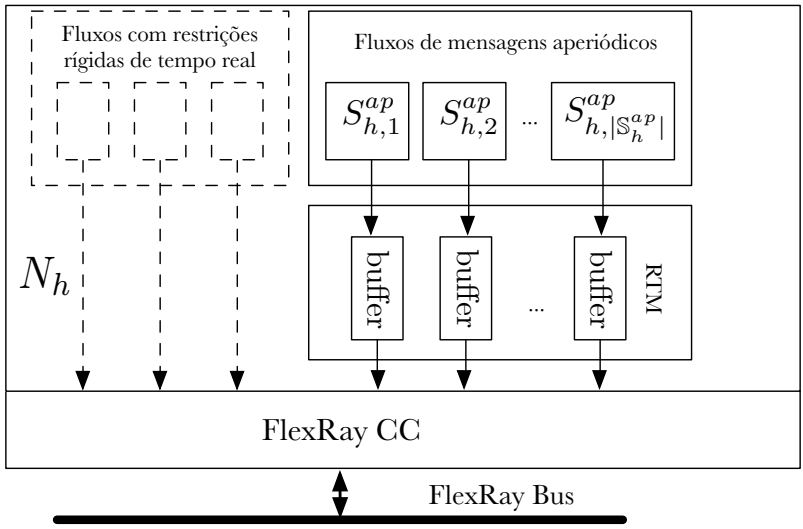

Figura 41 – *Middleware de Tempo Real*

Quando uma mensagem $m_{h,j}^{ap,l}$ é gerada por $S_{h,j}^{ap}$, a mesma não é diretamente enviada para o controlador de comunicação, mas sim armazenada no respectivo *buffer*.

Durante a operação normal do sistema, quando uma mensagem $m_{h,j}^{ap,l}$ chega no *buffer*, o RTM simplesmente a transfere para o CC, onde a mesma irá competir pelo barramento utilizando seu $FrameID$, de acordo com a operação normal do FlexRay. Mas sempre que um RTM detectar que o sistema está sobrecarregado e mensagens geradas por fluxos aperiódicos estão sendo postergadas indefinidamente, o mesmo envia uma mensagem específica instruindo todos os nodos do sistema a entrar em modo *backoff*.

Durante o modo de *backoff*, a operação do sistema em relação ao ST e a fluxos de mensagens que possuem restrições temporais críticas mas transmitem no DN continua normal. Já a transferência de uma mensagem gerada por um fluxo aperiódico $S_{h,j}^{ap}$ do RTM para o CC depende da probabilidade de *backoff* de $S_{h,j}^{ap}$, de acordo com a seguinte operação: antes do início do *slot* dinâmico associado com $S_{h,j}^{ap}$, o RTM verifica se existe uma mensagem $m_{h,j}^{ap,l}$ armazenada em $B_{h,j}$. Se existir, o RTM faz um sorteio e decide, com base no parâmetro probabilístico $\overline{\mathcal{P}_{h,j}^{bk}}$, se $m_{h,j}^{ap,l}$ será transferida para o CC, onde irá competir pelo barramento no ciclo FlexRay atual (potencialmente contribuindo para postergar o envio de uma mensagem com prioridade mais baixa), ou se $m_{h,j}^{ap,l}$ será postergada para o ciclo FlexRay seguinte, aumentando as chances de transmissão de mensagens com prioridades mais baixas.

Quando o sistema detecta que não existe mais a postergação indefinida de mensagens geradas por fluxos aperiódicos, ele sai do modo *backoff*, retornando ao modo de operação normal.

Esta proposta considera que fluxos de mensagens com restrições temporais rígidas e que estão alocadas no DN utilizam $FrameIDs$ de alta prioridade. No DN, quanto menor o valor de um $FrameID$, maior sua prioridade. Considera-se, também, que a definição destes $FrameIDs$ é realizada utilizando-se um dos métodos apresentados no Capítulo 3 como, por exemplo, aqueles propostos em (ZENG et al., 2011; KANG et al., 2012).

Existem diversos algoritmos que podem ser utilizados para disparar uma mudança no modo de operação do sistema (do modo normal para o modo *backoff* e vice-versa). Na Seção 6.2.3 será apresentada uma solução simples, apesar de que outras soluções podem ser mais apropriadas dependendo das características de um sistema.

6.2.1 Probabilidade de transmissão de uma mensagem aperiódica no DN

Como já discutido neste capítulo, no método que está sendo proposto, o RTM de um nodo sinaliza uma mudança para o modo *backoff* sempre que o mesmo detecta que mensagens geradas por fluxos aperiódicos estão sendo postergadas indefinidamente.

Quando em modo *backoff*, um RTM utiliza as probabilidades de *backoff* de $S^{ap}_{h,j}$ para decidir aleatoriamente se uma mensagem $m^{ap,l}_{h,j}$ será transferida do respectivo *buffer* para o CC (onde irá competir pelo barramento) no ciclo FlexRay atual, ou se $m^{ap,l}_{h,j}$ permanecerá no *buffer* até o ciclo FlexRay seguinte, quando será tomada uma nova decisão.

Entretanto, quando em modo *backoff*, a transmissão de $m^{ap,l}_{h,j}$ em um dado ciclo FlexRay não depende apenas da probabilidade de *backoff* de $S^{ap}_{h,j}$, mas sim de três fatores:

- da probabilidade de *backoff* dos fluxos de mensagens em $hp(S^{ap}_{h,j})$;
- de $pLatestTx^{ap}_{h,j}$;
- da probabilidade de *backoff* de $S^{ap}_{h,j}$.

Dados estes fatores, é possível calcular a probabilidade geral de uma mensagem gerada por $S^{ap}_{h,j}$ ser transmitida em um dado ciclo FlexRay. Esta probabilidade geral é chamada de *probabilidade de transmissão* $\mathcal{P}^{tr}_{h,j}$, e o cálculo de seu valor é feito com o auxílio de uma árvore binária de probabilidades $\mathfrak{T}$. O cálculo da probabilidade de transmissão $\mathcal{P}^{tr}_{h,j}$ considera que todos os fluxos de mensagens aperiódicos de um sistema sempre têm uma mensagem para transmitir.

Cada nível de $\mathfrak{T}$ é associado com um único fluxo de mensagens aperiódico. Quanto mais alto o nível, maior a prioridade do fluxo, ou seja, o primeiro nível representa o fluxo com $FrameID = 1$, o segundo nível representa o fluxo com $FrameID = 2$ e assim por diante. As folhas do último nível de $\mathfrak{T}$ estão relacionadas a um fluxo de mensagens que está sendo analisado.

Um caminho da raiz de $\mathfrak{T}$ até uma folha do último nível representa uma sequência de eventos que pode resultar ou não na transmissão de uma mensagem $m^{ap,l}_{h,j}$. Tais eventos representam a transmissão ou não de mensagens geradas por fluxos em $hp(S^{ap}_{h,j})$.

Um exemplo de árvore de probabilidades $\mathfrak{T}$ é dado na Figura 42, onde:

- $S_{h,j}^{ap}$ representa o caso onde o mecanismo de *backoff* decidiu que uma mensagem de $S_{h,j}^{ap}$ será transmitida no ciclo FlexRay atual;

- $\overline{S_{h,j}^{ap}}$ representa o caso onde o mecanismo de *backoff* decidiu que uma mensagem de $S_{h,j}^{ap}$ não será transmitida no ciclo FlexRay atual;

- $S_{h,1}^{ap} \wedge S_{h,2}^{ap} \wedge \overline{S_{h,3}^{ap}}$ representa a situação onde uma mensagem de $S_{h,1}^{ap}$ é transmitida, uma mensagem de $S_{h,2}^{ap}$ também é transmitida, mas uma mensagem de $S_{h,3}^{ap}$ não é transmitida no ciclo FlexRay atual;

- $\mathcal{P}$ indica a probabilidade do nó da árvore ser alcançado;

- Ct representa o valor do contador de *minislots* (MS) naquele ponto do caminho. É importante notar que Ct também inclui os MS não utilizados relacionados a *slots* dinâmicos que não estão associados com nenhum fluxo de mensagens.

$\mathfrak{T}$ é construída, durante a fase de projeto, utilizando-se o Algoritmo 3. O Algoritmo 3 possui três passos. No Passo 1 $\mathfrak{T}$ é inicializada e o nó raiz é criado (Linha 1). No Passo 2 (Linhas 2 a 8), os nós relacionados aos fluxos em $hp(S_{h,j}^{ap})$ são inseridos em $\mathfrak{T}$, e os respectivos caminhos são criados. No Passo 3 (Linhas 9 a 13) as folhas relacionados ao fluxo de mensagens $S_{h,j}^{ap}$ sob análise são inseridos em $\mathfrak{T}$. Mas ao invés de inserir folhas em todos os caminhos, as folhas relacionados a $S_{h,j}^{ap}$ são inseridas apenas nos caminhos que realmente permitem a transmissão de uma mensagem gerada por $S_{h,j}^{ap}$. A decisão sobre a inserção ou não de uma folha no último nível depende de $pLatestTx_{h,j}^{ap}$ e do valor de Ct. Por fim, é retornada a probabilidade de transmissão $\mathcal{P}_{h,j}^{tr}$ de $S_{h,j}^{ap}$.

Dado o Algoritmo 3, as probabilidades de transmissão de todos os fluxos de mensagens aperiódicos de um dado sistema FlexRay podem ser calculadas utilizando o Algoritmo 4.

A seguir será apresentado um exemplo que demonstra como podem ser calculadas as probabilidades de transmissão de um conjunto de fluxos de mensagens aperiódicos.

6.2.2 Exemplo 1

Para exemplificar como as probabilidades de transmissão de um conjunto de fluxos de mensagens aperiódicos podem ser calculadas

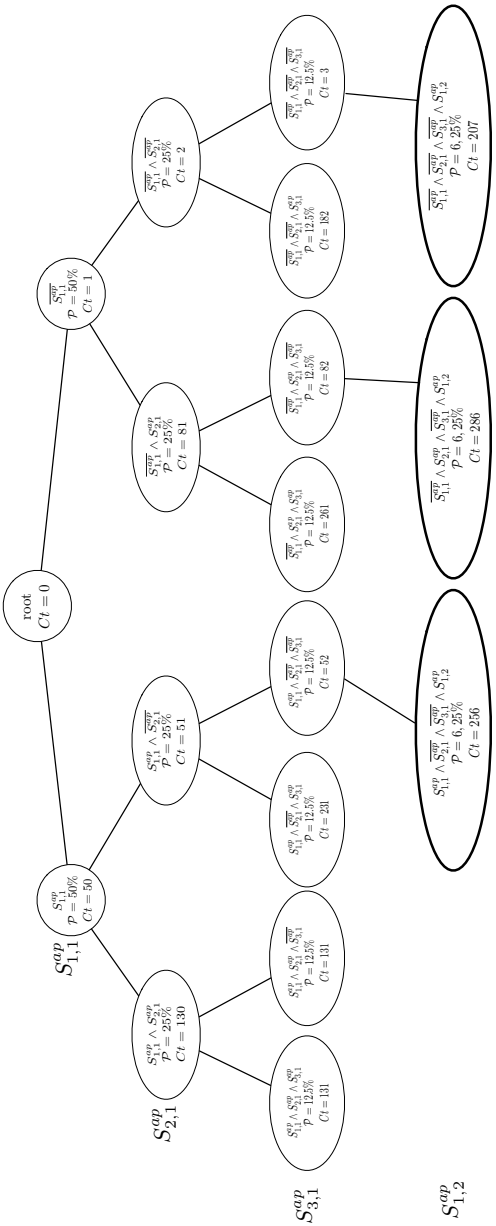

Figura 42 – Exemplo de árvore de probabilidades $\mathfrak{T}$

Algoritmo 3 Cálculo da probabilidade de transmissão de um fluxo de mensagens aperiódico

Entrada: $S_{h,j}^{ap}$ e $hp(S_{h,j}^{ap})$
Saída: $\mathcal{P}_{h,j}^{tr}$

1: Inicializa $\mathfrak{T}$
2: **for all** $S_{hp}^{ap} \in hp(S_{h,j}^{ap})$ **do**
3: Vá para o último nível *last* de $\mathfrak{T}$
4: **for all** node $\in$ *last* **do**
5: InsertStreamHP(node, S_{hp}^{ap}, left)
6: InsertStreamHP(node, S_{hp}^{ap}, right)
7: **end for**
8: **end for**
9: Vá para o último nível *last* de $\mathfrak{T}$
10: $\mathcal{P}_{h,j}^{tr} = 0$
11: **for all** folha $\in$ *last* **do**
12: $\mathcal{P}_{h,j}^{tr} = \mathcal{P}_{h,j}^{tr}$+ InsertStream(node, $S_{h,j}^{ap}$)
13: **end for**
14: **Retorne** $\mathcal{P}_{h,j}^{tr}$

15: **function** InsertStreamHP(nó, S_{temp}^{ap}, lado)
16: **if** nó.$ct \leq pLatestTx_{temp}^{ap}$ **then**
17: ctTemp = nó.$ct + C_{temp}^{dp}$
18: **else**
19: ctTemp = nó.$ct + 1$
20: **end if**
21: **if** lado = left **then**
22: nó.left = new node
23: nó.left.$ct = ctTemp$
24: nó.left.$\mathcal{P} = \mathcal{P}_{temp}^{tr} \times$ nó.$\mathcal{P}$
25: **else**
26: nó.right = new node
27: nó.right.$ct = ctTemp$
28: nó.right.$\mathcal{P} = \mathcal{P}_{temp}^{tr} \times$ nó.$\mathcal{P}$
29: **end if**
30: **end function**

```
31: function InsertStream(folha, S^ap_temp)
32:     if folha.ct ≤ pLatestTx^ap_temp then
33:         folha.left = new node
34:         folha.left.ct = folha.ct + C^ap_temp
35:         folha.left.P = P^tr_temp × folha.P
36:         retorna folha.left.P
37:     else
38:         retorna 0
39:     end if
40: end function
```

Algoritmo 4 Cálculo da probabilidade de qualquer fluxo de mensagens

Entrada: $\mathcal{S}^{ap}$

```
1: for all S^ap_{h,j} ∈ 𝕊^ap do
2:     Probabilidade de transmissão de S^ap_{h,j} = CreateTree(hp(S^ap_{h,j}),
       S^ap_{h,j})
3: end for
```

utilizando-se os Algoritmos 3 e 4, considere um sistema FlexRay com os parâmetros apresentados na Tabela 9. Para facilitar o entendimento, a explicação será simplificada considerando-se que todos os parâmetros do sistema, incluindo aqueles relacionados aos fluxos de mensagens aperiódicos, serão dados em múltiplos inteiros de *minislots*.

Parâmetro	**valor (em MS)**
FC_{bus}	725
ST_{bus}	435
DN_{bus}	290

Tabela 9 – Parâmetros de sistema para o Exemplo 1

O conjunto de fluxos de mensagens aperiódicos é dado na Tabela 10. Note que os *slots* dinâmicos de 4 a 7, 9 a 14 e 16 a 19 não estão associados com nenhum fluxo de mensagens. Estes *slots* poderiam estar, por exemplo, reservados para usos futuros.

Para este exemplo, a probabilidade de *backoff* de todos os fluxos de mensagens foi definida, de forma arbitrária, como sendo igual a 50%. Isto é,

$$\forall S^{ap}_{h,j} \in \mathbb{S}^{ap}, \overline{\mathcal{P}^{bk}_{h,j}} = 50\%. \tag{6.2}$$

Fluxo	*FrameID*	Nodo	$C_{h,j}^{ap}$	$pLatestTx$ **de** $S_{h,j}^{ap}$	$\overline{\mathcal{P}_{h,j}^{bk}}$
$S_{1,1}^{ap}$	1	1	50	90	50%
$S_{2,1}^{ap}$	2	2	80	210	50%
$S_{3,1}^{ap}$	3	3	180	110	50%
$S_{1,2}^{ap}$	8	1	200	90	50%
$S_{4,1}^{ap}$	15	4	20	270	50%
$S_{1,3}^{ap}$	20	1	200	90	50%

Tabela 10 – Conjunto de fluxos de mensagens aperiódicos para o Exemplo 1

Por questões de clareza, será apresentada apenas o processo completo para o cálculo da probabilidade de transmissão de $S_{1,2}^{ap}$. A lista completa das probabilidades de transmissão calculadas é dada na Tabela 11.

Para calcular a probabilidade de transmissão de $S_{1,2}^{ap}$ são seguidos os seguintes passos. Inicialmente, $\mathfrak{T}$ é inicializada, o nó raiz é criado e os nós relacionados aos fluxos de mensagem em $hp(S_{1,2}^{ap})$ são inseridos (Passos 1 e 2 do Algoritmo 3). Os Passos 1 e 2 geram os níveis 1 a 4 da árvore representada na Figura 42.

A seguir, $S_{1,2}^{ap}$ é inserido em $\mathfrak{T}$, e a soma das probabilidades de transmissão das folhas do último nível de $\mathfrak{T}$ são retornadas, dando uma probabilidade de transmissão de 18.75% para $S_{1,2}^{ap}$. É importante notar que as folhas do último nível de $\mathfrak{T}$ representam as chances efetivas de uma mensagem gerada por $S_{1,2}^{ap}$ ser transmitida. Essa representação é consistente com a operação do Algoritmo 3.

Fluxo de mensagens	**Probabilidade de transmissão calculada (%)**	**Resultado da simulação (%)**
$S_{1,1}^{ap}$	50.000	49.75
$S_{2,1}^{ap}$	50.000	51.25
$S_{3,1}^{ap}$	37.500	38.75
$S_{1,2}^{ap}$	18.750	18.50
$S_{4,1}^{ap}$	40.625	38.60
$S_{1,3}^{ap}$	6.260	6.55

Tabela 11 – Probabilidades de transmissão para o Exemplo 1

Para validar as probabilidades de transmissão que foram calculadas utilizando-se o Algoritmo 3, o comportamento do mecanismo de escalonamento probabilístico proposto nesta seção foi simulado. O simulador utilizado implementa o MAC do DN e o RTM descrito nesta seção. No simulador, a decisão sobre a transmissão de uma mensagem é tomada com o auxílio de um gerador de números aleatórios. O simulador foi implementado em C++.

Para fins de comparação, foram utilizados na simulação os mesmos parâmetros apresentados nas Tabelas 9 e 10, e foram simulados 2.000 ciclos FlexRay. Os resultados da simulação são apresentados na Tabela 11. Como pode ser observado, os valores calculados são bastante próximos dos valores obtidos na simulação.

6.2.3 Seleção de modo

Como discutido anteriormente, no método proposto nesta seção um sistema FlexRay pode estar em dois modos de operação distintos: modo normal e modo *backoff*. A seguir será apresentada uma solução simples para disparar uma troca de modo.

Inicialmente, o sistema está em modo de operação normal e segue a operação normal do FlexRay. Quando uma mensagem $m_{h,j}^{ap,l}$ gerada por um fluxo $S_{h,j}^{ap}$ chega no *buffer*, o *middleware* de tempo real simplesmente transfere $m_{h,j}^{ap,l}$ para o controlador de comunicação.

Se $S_{h,j}^{ap}$ gera uma segunda mensagem $m_{h,j}^{ap,l+1}$, mas $m_{h,j}^{ap,l}$ ainda está no CC aguardando para ser transmitida, o RTM mantém $m_{h,j}^{ap,l+1}$ no *buffer* até que $m_{h,j}^{ap,l}$ seja transmitida (ou seja, até que $m_{h,j}^{ap,l}$ saia do CC). Mas se após um número predefinido de ciclos FlexRay $m_{h,j}^{ap,l}$ não foi transmitida, e $m_{h,j}^{ap,l+1}$ ainda está no *buffer*, o RTM assume que o sistema está sobrecarregado e mensagens aperiódicas estão sendo postergadas indefinidamente. Quando isso acontece, o RTM envia uma mensagem especial para todos os demais nodos da rede sinalizando que o sistema deve entrar em modo *backoff*. Nesta proposta se considera que os dados que disparam uma troca de modo são transmitidos em uma mensagem do DN com alta prioridade.

Quando em modo *backoff*, cada nodo utiliza as probabilidades de *backoff* de seus fluxos aperiódicos para decidir sobre a transmissão de mensagens. E cada RTM continua monitorando o tamanho de seus *buffers*. Se após um número pré-definido de ciclos FlexRay um RTM sente que as mensagens aperiódicas estão sendo enviadas para a rede,

ele assume que a condição de sobrecarga passou e o sistema pode retomar o modo de operação normal. Este RTM então envia para os demais nodos de rede uma mensagem especial sinalizando que considera que a operação normal pode ser retomada.

Se todos os nodos de rede sinalizam que a condição de sobrecarga passou, o sistema retorna para o modo de operação normal.

6.3 MÉTODO DE ESCALONAMENTO PROBABILÍSTICO DO SEGMENTO DINÂMICO DO FLEXRAY DESCENTRALIZADO

Na Seção 6.2 foi apresentado um mecanismo probabilístico para arbitrar a transmissão de mensagens geradas por fluxos aperiódicos no DN. No entanto, naquele método a decisão sobre uma mudança do modo de operação normal para o modo *backoff* é global, com todos os nodos de rede mudando de forma simultânea. Tal mudança de modo coordenada requer a troca de mensagens de controle adicionais entre os diversos nodos de rede.

Nessa seção é apresentada uma extensão do Método de Escalonamento Probabilístico do Segmento Dinâmico do FlexRay proposto na Seção 6.2.

O método que será apresentado nesta seção é descentralizado: cada fluxo de mensagem aperiódico $S^{ap}_{h,j}$, além de estar associado com a tupla $\{FID^{ap}_{h,j}, C^{ap}_{h,j}, pLatestTx^{ap}_{h,j}\}$ e com os valores probabilísticos $\overline{\mathcal{P}^{bk}_{h,j}}$ e $\mathcal{P}^{bk}_{h,j}$, também está associado com um *middleware* de tempo real $RTM_{h,j}$ exclusivo. De forma semelhante ao RTM apresentado na Seção 6.2, cada $RTM_{h,j}$ possui um *buffer* que armazena as mensagens geradas por $S^{ap}_{h,j}$. Assim como na proposta apresentada na Seção 6.2, durante o modo de operação normal, quando um $RTM_{h,j}$ recebe uma mensagem $m^{ap,l}_{h,j}$ gerada por $S^{ap}_{h,j}$, o mesmo imediatamente transfere $m^{ap,l}_{h,j}$ para o controlador de comunicação, onde a mensagem irá competir pelo barramento utilizando $FID^{ap}_{h,j}$ e $pLatestTx^{ap}_{h,j}$. Quando $RTM_{h,j}$ detecta que $S^{ap}_{h,j}$ está enviando rajadas de mensagens, potencialmente causando a postergação indefinida de mensagens com prioridades mais baixas, o mesmo entra em modo *backoff*, onde a decisão sobre o envio de $m^{ap,l}_{h,j}$ para o CC depende de $\overline{\mathcal{P}^{bk}_{h,j}}$. Mas diferente do método proposto na Seção 6.2, a mudança de modo em um *middleware* de tempo real é individual e independente dos demais *middlewares* de tempo real do sistema. Ou seja, em uma dada situação, $RTM_{h,j}$ pode estar em modo *backoff*, enquanto $RTM_{h,j+1}$ está no modo de operação normal.

Um exemplo de nodo com diversos fluxos de mensagens aperiódicos e um *middleware* de tempo real por fluxo é apresentado na Figura 43.

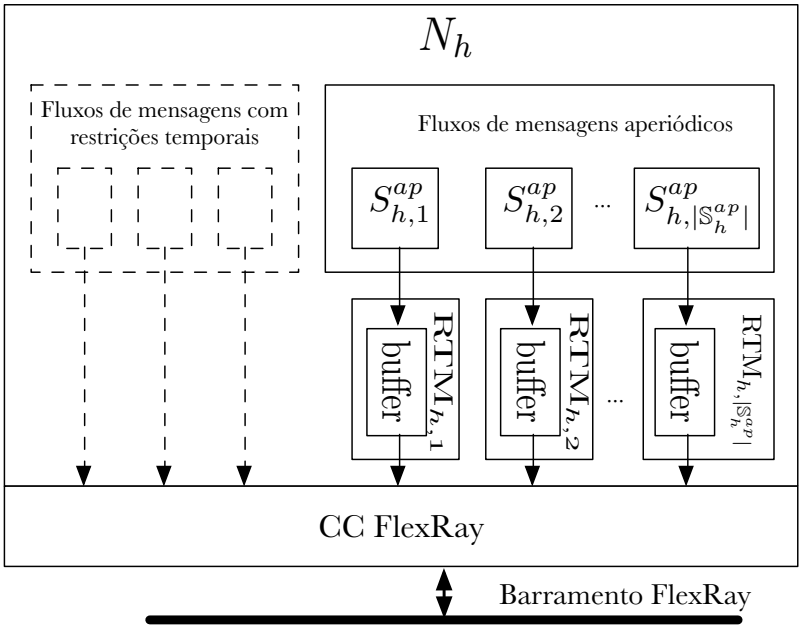

Figura 43 – *Middleware de Tempo Real*

Nesta proposta, o mecanismo que dispara uma troca de modo em cada $RTM_{h,j}$ é baseado no número de transmissões de mensagens geradas por $S^{ap}_{h,j}$. Cada $RTM_{h,j}$ está associado com três parâmetros:

- Um parâmetro $\mathcal{N}_{h,j}$ que indica o número de mensagens **consecutivas** de $S^{ap}_{h,j}$ que $RTM_{h,j}$ pode transferir para o CC antes de entrar em modo *backoff*;
- Um parâmetro $\mathcal{NB}_{h,j}$ que indica o número de ciclos FlexRay que $RTM_{h,j}$ permanece em modo *backoff*;
- Um parâmetro Λ_{min} que indica o número mínimo de oportunidades de transmissão que se deseja que qualquer fluxo de mensagens aperiódico tenha durante um intervalo de tempo correspondente a Δ_{FC} ciclos FlexRay.

Para ilustrar o mecanismo proposto, considere um sistema FlexRay com $DN_{bus} = 20$ MS e três fluxos de mensagens aperiódicos cujos valores são dados na Tabela 12. $S^{ap}_{h,1}$, $S^{ap}_{h,2}$ e $S^{ap}_{h,3}$ estão no mesmo nodo N_h e estão associados, respectivamente, com $RTM_{h,1}$, $RTM_{h,2}$ e $RTM_{h,3}$. Considere que os fluxos de mensagens geram uma nova mensagem imediatamente após a mensagem anterior ser transferida para o CC.

Fluxo	$C^{ap}_{h,j}$	$FrameID$	$pLatestTx$ de $S^{ap}_{h,j}$	$\mathcal{N}_{h,j}$	$\mathcal{NB}_{h,j}$	$\overline{\mathcal{P}_{h,j}}$
$S^{ap}_{h,1}$	11	1	10	3	3	50%
$S^{ap}_{h,2}$	6	2	15	5	5	50%
$S^{ap}_{h,3}$	5	3	16	3	3	50%

Tabela 12 – Conjunto de fluxos de mensagens aperiódicos para o Exemplo 2

Considere que o sistema inicia no modo de operação normal. No primeiro ciclo FlexRay, $S^{ap}_{h,1}$ envia uma mensagem que ocupa 11 MS, e $S^{ap}_{h,2}$ envia uma mensagem que ocupa outros 6 MS, não restando nenhum MS para a transmissão de uma mensagem gerada por $S^{ap}_{h,3}$. Se não for utilizado um mecanismo de *backoff*, esse comportamento seguirá sendo repetido, com a transmissão das mensagens geradas por $S^{ap}_{h,3}$ sendo postergadas indefinidamente.

Entretanto, se for utilizado o mecanismo de *backoff* proposto neste capítulo, no terceiro ciclo FlexRay $\mathcal{N}_{h,1}$ é atingido, e $RTM_{h,1}$ entra em modo *backoff*. No quarto ciclo FlexRay, $RTM_{h,1}$ irá definir de forma aleatória, com base em $\overline{\mathcal{P}^{bk}_{h,1}}$, se uma mensagem gerada por $S^{ap}_{h,1}$ será transferida para o CC. Suponha que $RTM_{h,1}$ define que uma mensagem **não** será transferida. Agora, o *slot* dinâmico associado com $S^{ap}_{h,1}$ tem tamanho de apenas 1 MS, uma mensagem gerada por $S^{ap}_{h,2}$ ocupa 6 MS, e como restou espaço no segmento Dinâmico, uma mensagem gerada por $S^{ap}_{h,3}$ pode ser finalmente transmitida.

A seguir será abordada a questão da definição dos parâmetros associados com cada fluxo de mensagens aperiódicas. Será, também, apresentado um conjunto de algoritmos que podem ser utilizados na definição de tais parâmetros.

6.3.1 Definição dos parâmetros

Inicialmente, considere o conjunto de fluxos de mensagens aperiódicos $\mathbb{S}^{ap}$ apresentado na Tabela 13. Para este exemplo, os nodos nos quais tais fluxos estão alocados não é relevante, pois interessam apenas o tamanho das mensagens e o $FrameID$ associado com cada fluxo. Considere um segmento Dinâmico com tamanho de 10 MS e um intervalo de tempo correspondente a 4 ciclos FlexRay. Considere que um fluxo sempre tem uma mensagem pronta para ser transmitida e que seu tamanho é dado em múltiplos de MS.

Fluxo de mensagens	$\boldsymbol{C^{ap}_{h,j}}$ **(MS)**	$FID^{ap}_{h,j}$
$S^{ap}_{h,1}$	3	1
$S^{ap}_{h,2}$	3	2
$S^{ap}_{h,3}$	3	3
$S^{ap}_{h,4}$	3	4
$S^{ap}_{h,5}$	3	5

Tabela 13 – Conjunto de mensagens aperiódicas $\mathbb{S}^{ap}$

É fácil observar que sem um mecanismo de *backoff*, os fluxos $S^{ap}_{h,1}$, $S^{ap}_{h,2}$ e $S^{ap}_{h,3}$ irão transmitir em todos os ciclos FlexRay, com uma utilização de 100%.

Considere agora que o fluxo de mensagens $S^{ap}_{h,1}$ tem uma probabilidade de *backoff* $\overline{\mathcal{P}^{bk}_{h,1}} = 50\%$ e que $RTM_{h,1}$ está no modo *backoff*. Com tal probabilidade de *backoff*, $S^{ap}_{h,1}$ irá transmitir em 2 ciclos FlexRay, liberando 2 MS nos outros dois ciclos. Estes 2 MS podem ser utilizados para a transmissão de mensagens geradas por $S^{ap}_{h,4}$. De forma semelhante, se $S^{ap}_{h,2}$ e $S^{ap}_{h,3}$ também tiverem $\mathcal{P}^{bk}_{h,2} = 50\%$ e $\mathcal{P}^{bk}_{h,3} = 50\%$, e $RTM_{h,2}$ e $RTM_{h,3}$ também estiverem em modo *backoff*, existe uma probabilidade, mesmo que baixa, das mensagens geradas por $S^{ap}_{h,1}$, $S^{ap}_{h,2}$ e $S^{ap}_{h,3}$ sofrerem *backoff* no mesmo ciclo FlexRay, permitindo a transmissão de mensagens geradas por $S^{ap}_{h,4}$ e $S^{ap}_{h,5}$.

Se todos os fluxos de mensagens aperiódicos de um conjunto $\mathbb{S}^{ap}$ possuírem uma probabilidade de *backoff* não nula, então existe uma probabilidade, mesmo que baixa, de que nenhuma das mensagens geradas pelos fluxos em $\mathbb{S}^{ap}$ seja postergada indefinidamente, pois neste caso podem existir cenários onde apenas as mensagens de um dos fluxos em $\mathbb{S}^{ap}$ não sofrem *backoff* em determinados ciclos FlexRay.

Apesar de que mesmo uma pequena probabilidade de *backoff* é

suficiente para minimizar a postergação indefinida mesmo em mensagens de baixa prioridade, um fluxo pode ter apenas uma oportunidade de transmissão durante um longo tempo de execução do sistema. Por isso, nesta proposta considera-se que um fluxo $S^{ap}_{h,j}$ deve ter no mínimo Λ_{min} oportunidades de transmitir suas mensagens para o barramento durante um intervalo de tempo correspondente a Δ_{FC} ciclos FlexRay. Nesta proposta, considera-se que Λ_{min} é igual para todos os fluxos de mensagens em $\mathbb{S}^{ap}$.

Devido ao MAC do Segmento Dinâmico, para que um fluxo de mensagens $S^{ap}_{h,j}$ consiga atingir Λ_{min}, é necessário limitar a utilização de rede causada por $hp(S^{ap}_{h,j})$. A limitação da utilização causada por $hp(S^{ap}_{h,j})$ pode ser alcançada através de uma seleção cuidadosa de valores para os parâmetros probabilísticos associados a cada fluxo de mensagens em $\mathbb{S}^{ap}$.

Suponha que exista um fluxo $S^{ap}_{h,j}$ que tem $FID^{ap}_{h,j} = 1$, para o qual se quer $\Lambda_{min} = 50$ transmissões de mensagens durante um intervalo de tempo $\Delta_{FC} = 100$ ciclos FlexRay. Como $S^{ap}_{h,j}$ tem o maior $FrameID$ do sistema, não sofre interferências de outros fluxos de mensagens. Dado este cenário, possíveis valores para os parâmetros probabilísticos de $S^{ap}_{h,j}$ são apresentados na Tabela 14.

Sistema A	**Sistema B**	**Sistema C**
$\mathcal{N}_{h,j}$=50 Msgs	$\mathcal{N}_{h,j} = 0$ Msgs	$\mathcal{N}_{h,j} = 5$ Msgs
$\mathcal{NB}_{h,j}$=50 FC	$\mathcal{NB}_{h,j}$=100 FC	$\mathcal{NB}_{h,j}$=15 FC
$\overline{\mathcal{P}^{bk}_{h,j}}$=0%	$\overline{\mathcal{P}^{bk}_{h,j}}$=50%	$\overline{\mathcal{P}^{bk}_{h,j}}$=66.66%

Tabela 14 – Possíveis valores para os parâmetros probabilísticos de $S^{ap}_{h,j}$

Usando o primeiro conjunto de valores (Sistema A), $S^{ap}_{h,j}$ irá transmitir 50 mensagens e, a seguir, $RTM_{h,j}$ entrará em modo *backoff*. Como $\overline{\mathcal{P}^{bk}_{h,j}}$=0%, isto é, a probabilidade de uma mensagem gerada por $S^{ap}_{h,j}$ não sofrer *backoff* e ser enviada para o CC é zero, nos próximos 50 ciclos FlexRay $S^{ap}_{h,j}$ não irá transmitir nenhuma outra mensagem, resultando em 50 transmissões de mensagens em um intervalo de 100 ciclos FlexRay.

Com o segundo conjunto de valores (Sistema B), $RTM_{h,j}$ estará sempre em modo *backoff* ($\mathcal{N}_{h,j} = 0$ mensagens e $\mathcal{NB}_j$=100 ciclos FlexRay) e, em cada ciclo FlexRay, uma mensagem gerada por $S^{ap}_{h,j}$ terá 50% de chance de ser enviada para o CC e, consequentemente, ser transmitida no barramento. Note que, novamente, o resultado são 50 transmissões de mensagens em um intervalo de 100 ciclos FlexRay.

Por fim, com o terceiro conjunto de valores (Sistema C), a cada 20 ciclos FlexRay, nos primeiros 5 ciclos FlexRay serão transmitidas no barramento 5 mensagens de $S^{ap}_{h,j}$, resultando em 25 mensagens no intervalo de 100 ciclos FlexRay. Nos 15 ciclos restantes uma mensagem gerada por $S^{ap}_{h,j}$ tem uma probabilidade de ser transferida para o CC igual a 33,34%, resultando em cerca de 25 mensagens adicionais no intervalo de 100 ciclos FlexRay. No total, tem-se novamente uma média de 50 transmissões de mensagens em um intervalo de 100 ciclos FlexRay.

Como pode ser observado no exemplo acima, existe uma interdependência entre $\mathcal{N}_{h,j}$, $\mathcal{NB}_{h,j}$ e $\overline{\mathcal{P}^{bk}_{h,j}}$. Nesta proposta, foi feita a opção de primeiro definir $\overline{\mathcal{P}^{bk}_{h,j}}$, para então, com base nas características de cada fluxo de mensagens aperiódico, definir $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$. Na próxima seção será apresentado um método para definir $\overline{\mathcal{P}^{bk}_{h,j}}$.

6.3.2 Definição de $\overline{\mathcal{P}^{bk}_{h,j}}$ e $\mathcal{P}^{bk}_{h,j}$ (Algoritmo 5)

$\overline{\mathcal{P}^{bk}_{h,j}}$ e $\mathcal{P}^{bk}_{h,j}$ representam, respectivamente, a probabilidade de uma mensagem gerada por $S^{ap}_{h,j}$ não sofrer ou sofrer *backoff*, isto é, representam respectivamente a probabilidade de uma mensagem de $S^{ap}_{h,j}$ ser transferida ou não ser transferida do *buffer* de $RTM_{h,j}$ para o CC quando em modo *backoff*. $\overline{\mathcal{P}^{bk}_{h,j}}$ e $\mathcal{P}^{bk}_{h,j}$ são valores complementares, isto é, $\mathcal{P}^{bk}_{h,j} = 1 - \overline{\mathcal{P}^{bk}_{h,j}}$.

Estes valores devem ser escolhidos de forma a permitir que a) Λ_{min} seja atingido por $S^{ap}_{h,j}$ e b) que a máxima utilização de rede $U(S^{ap}_{h,j})_{max}$ causada por $S^{ap}_{h,j}$ não impeça que fluxos de mensagens com prioridades mais baixas também atinjam Λ_{min}. Nesta proposta, a definição do valor de $\mathcal{P}^{bk}_{h,j}$ é feita utilizando-se o Algoritmo 5.

O Algoritmo 5 testa se Λ_{min} se mantém para todos os fluxos de mensagens em $\mathbb{S}^{ap}$. Se Λ_{min} não é atingido para um dado fluxo de mensagens em $\mathbb{S}^{ap}$, é aumentada a probabilidade de *backoff* $\mathcal{P}^{bk}_{h,j}$ de fluxos de mensagens com prioridades mais altas que a de tal fluxo. O algoritmo é repetido até que todos os fluxos de mensagens em $\mathbb{S}^{ap}$ atinjam Λ_{min}, ou até que não seja possível para algum dos fluxos em $\mathbb{S}^{ap}$ atingir Λ_{min} (caso em que o sistema é considerado não satisfatório). O Algoritmo 5 utiliza simulação, e considera que um sistema sob análise está sob carga máxima, isto é, todos os fluxos de mensagens em $\mathbb{S}^{ap}$ sempre tem uma mensagem pronta a ser transmitida. O raciocínio por trás dessa premissa é: se um sistema é considerado escalonável sob

carga máxima, então o mesmo também será escalonável sob uma carga menor.

Os parâmetros de entrada do Algoritmo 5 são o conjunto de fluxos de mensagens aperiódicos $\mathbb{S}^{ap}$, um intervalo de tempo Δ_{FC} dado em múltiplos inteiros de ciclo FlexRay, o número mínimo de oportunidades de transmissão Λ_{min} que um fluxo em $\mathbb{S}^{ap}$ deve alcançar durante Δ_{FC} e o tamanho DN_{bus} do segmento Dinâmico. A saída do algoritmo é as probabilidade de *backoff* $\mathcal{P}^{bk}_{h,j}$, e a utilização de rede máxima $U(S^{ap}_{h,j})_{max}$ durante o intervalo de tempo Δ_{FC}.

O Algoritmo 5 utiliza quatro variáveis auxiliares:

- $\mathbb{S}^{ap}_{teste}$: um subconjunto de $\mathbb{S}^{ap}$ composto pela união de $S^{ap}_{h,j}$ com $hp(S^{ap}_{h,j})$ ($\mathbb{S}^{ap}_{teste} = S^{ap}_{h,j} \cup hp(S^{ap}_{h,j})$);
- $\mathcal{P}^{bk}_{max}$: maior probabilidade de *backoff* dentre os fluxos de mensagens em $\mathbb{S}^{ap}_{teste}$ ($\forall S^{ap}_{p} \in \mathbb{S}^{ap}_{teste}$, $\mathcal{P}^{bk}_{max} \geq \mathcal{P}^{bk}_{p}$);
- STD: indica o fluxo de mensagem $S^{ap}_{p} \in \mathbb{S}^{ap}_{teste}$ cuja probabilidade de *backoff* $\mathcal{P}^{bk}_{p}$ deve ser incrementada;
- TC_p: indica o número de transmissões de mensagens de cada $S^{ap}_{p} \in \mathbb{S}^{ap}_{teste}$ após uma simulação.

O Algoritmo 5 tenta definir valores que tornem possível que todo fluxo $S^{ap}_{h,j}$ em $\mathbb{S}^{ap}_{teste}$ atinja Λ_{min}. O mesmo inicia definindo cada $\mathcal{P}^{bk}_{h,j}$ como 0 (Linhas 1 a 3). Então, para um dado fluxo $S^{ap}_{h,j}$, é executado um laço onde primeiramente é definido o conjunto $\mathbb{S}^{ap}_{teste} = S^{ap}_{h,j} \cup hp(S^{ap}_{h,j})$ (Linha 5).

Na sequência, é chamado o Algoritmo 6 (Linha 9). O Algoritmo 6 recebe como entradas $\mathbb{S}^{ap}_{teste}$, Δ_{FC} e DN_{bus}, simula um sistema FlexRay em modo *backoff* e retorna o número TC_p de cada fluxo de mensagens em $\mathbb{S}^{ap}_{teste}$ durante o intervalo de tempo Δ_{FC}. O Algoritmo 6 será descrito na Seção 6.3.3.

No passo seguinte é checado se existe um fluxo $S^{ap}_{p} \in \mathbb{S}^{ap}_{teste}$ cujo TC_p é menor que Λ_{min}. Se existe, uma variável de controle *recalcularSistema* é definida como verdadeira (*true*) (Linhas 10 a 14).

Se *recalcularSistema* é verdadeiro, então o algoritmo encontra $\mathcal{P}^{bk}_{max}$ de $\mathbb{S}^{ap}_{teste}$ (Linha 16) e define STD como sendo o fluxo de maior prioridade S^{ap}_{hp} em $\mathbb{S}^{ap}_{teste}$ cujo $\mathcal{P}^{bk}_{hp}$ é menor que $\mathcal{P}^{bk}_{max}$. Se a probabilidade de *backoff* de todos os fluxos em $\mathbb{S}^{ap}_{teste}$ é igual a $\mathcal{P}^{bk}_{max}$, STD é definido como sendo o fluxo de maior prioridade S^{ap}_{1} em $\mathbb{S}^{ap}_{teste}$ e $\mathcal{P}^{bk}_{STD}$ é incrementado em 1% (Linhas 17 a 22). Se, após o incremento, $\mathcal{P}^{bk}_{STD}$

for igual ou maior que 100%, o Algoritmo 5 para e retorna com a indicação de que o sistema não é escalonável (uma probabilidade de *backoff* $\mathcal{P}^{bk}_{h,j} = 100\%$ significa que a chance de um fluxo $S^{ap}_{h,j}$ transmitir uma mensagem é zero) (Linha 24). Por fim, é executada a próxima iteração do laço.

Se $recalcularSistema$ é falso, o Algoritmo 5 executa para o fluxo $S^{ap}_{h,j+1}$.

Se é possível encontrar $\mathcal{P}^{bk}_{h,j}$ que permite que todos os fluxos em $\mathbb{S}^{ap}_{teste}$ alcancem Λ_{min}, é calculada a utilização máxima $U(S^{ap}_{h,j})_{max}$ de cada fluxo em $\mathbb{S}^{ap}_{teste}$ e o Algoritmo 5 termina retornando $\mathcal{P}^{bk}_{h,j}$ e $U(S^{ap}_{h,j})_{max}$ para todos os fluxos em $\mathbb{S}^{ap}_{teste}$ (Linha 32).

6.3.3 Simulador FlexRay com modo *backoff* (Algoritmo 6)

O Algoritmo 5 depende da simulação de um sistema FlexRay executando em modo *backoff*. Esta simulação é realizada pelo Algoritmo 6. O Algoritmo 6 recebe como entrada um conjunto de fluxos de mensagens aperiódicos $\mathbb{S}^{ap}_{sim}$, Δ_{FC} e DN_{bus}, e retorna o número TC_p de transmissões efetuadas por cada fluxo S^{ap}_{p} em $\mathbb{S}^{ap}_{sim}$ durante o intervalo de tempo Δ_{FC}.

O algoritmo inicia definindo TC_p de todos os fluxos em $\mathbb{S}^{ap}_{sim}$ como 0 (Linha 1). Então, é executado um laço que simula um sistema FlexRay executando durante Δ_{FC} ciclos FlexRay (Linha 2). O número do ciclo FlexRay ao qual corresponde uma iteração do laço é dado por $FC_{counter}$. Em cada iteração do laço, é simulado o MAC do Segmento Dinâmico (Linhas 5 a 22). No início de cada iteração, o contador de *slots* dinâmicos $vSlotCounter$ e o contador de *minislots* $MS_{counter}$ são definidos como zero. É, então, iniciado outro laço onde $MS_{counter}$ varia de 1 a DN_{bus}. Neste laço, pode ocorrer uma das seguintes ações:

- Se existe um fluxo S^{ap}_{p} associado com o $FrameID$ atual ($FID^{ap}_{p} = vSlotCounter$), então é verificado se $pLatestTx^{ap}_{p}$ é respeitado (Linha 6). Se $pLatestTx^{ap}_{p}$ não é respeitado, $MS_{counter}$ é incrementado em 1 MS e $vSlotCounter$ passa a apontar para o próximo *slot* dinâmico (Linhas 7 a 17). Se $pLatestTx^{ap}_{p}$ é respeitado, é simulado o mecanismo de *backoff* probabilístico (Linhas 9 a 16). Durante essa simulação:
  1. Uma variável *rand* recebe um valor aleatório inteiro entre 0 e 100% (Linha 10);
  2. se *rand* é maior que $\mathcal{P}^{bk}_{p}$ a mensagem gerada por S^{ap}_{p} será

Algoritmo 5 Algoritmo para definir $\overline{\mathcal{P}_{h,j}^{bk}}$, $\mathcal{P}_{h,j}^{bk}$ e $U(S_{h,j}^{ap})_{max}$

Entrada: $\mathbb{S}^{ap}$, Δ_{FC}, Λ_{min}, DN_{bus}
Saída: $\mathcal{P}_{h,j}^{bk}$ e $U(S_{h,j}^{ap})_{max}$ $\forall S_{h,j}^{ap} \in \mathbb{S}^{ap}$

1: **for all** $S_{h,j}^{ap} \in \mathbb{S}^{ap}$ **do**
2: $\quad \mathcal{P}_{h,j}^{bk} \leftarrow 0$
3: **end for**
4: **for all** $S_{h,j}^{ap} \in \mathbb{S}^{ap}$ **do**
5: $\quad \mathbb{S}_{teste}^{ap} = S_{h,j}^{ap} \cup hp(S_{h,j}^{ap})$
6: $\quad terminate \leftarrow false$
7: $\quad$ recalcularSistema $\leftarrow false$
8: $\quad$ **while** terminate = false **do**
9: $\quad\quad$ Algoritmo6($\mathbb{S}_{teste}^{ap}, \Delta_{FC}, DN_{bus}$)
10: $\quad\quad$ **for all** $S_p^{ap} \in \mathbb{S}_{teste}^{ap}$ **do**
11: $\quad\quad\quad$ **if** $TC_p < \Lambda_{min}$ **then**
12: $\quad\quad\quad\quad$ recalcularSistema $\leftarrow true$
13: $\quad\quad\quad$ **end if**
14: $\quad\quad$ **end for**
15: $\quad\quad$ **if** recalculateSystem = $true$ **then**
16: $\quad\quad\quad$ Encontre $\mathcal{P}_{max}^{bk}$
17: $\quad\quad\quad$ **if** $\forall S_{hp}^{ap} \in HP(S_{h,j}^{ap}), \mathcal{P}_{hp}^{bk} = \mathcal{P}_{max}^{bk}$ **then**
18: $\quad\quad\quad\quad$ STD = S_1^{ap}
19: $\quad\quad\quad$ **else**
20: $\quad\quad\quad\quad$ STD $\leftarrow$ fluxo S_{hp}^{ap} com a maior prioridade em $\mathbb{S}_{teste}^{ap}$ cujo $\mathcal{P}_{hp}^{bk}$ é menor que $\mathcal{P}_{max}^{bk}$
21: $\quad\quad\quad$ **end if**
22: $\quad\quad\quad$ Incrementa $\mathcal{P}_{STD}^{bk}$ de STD em 1%
23: $\quad\quad\quad$ **if** $\mathcal{P}_{STD}^{bk} \geq 100\%$ **then**
24: $\quad\quad\quad\quad$ Retorna sistema não é escalonável
25: $\quad\quad\quad$ **end if**
26: $\quad\quad$ **else**
27: $\quad\quad\quad$ terminate = true
28: $\quad\quad$ **end if**
29: $\quad$ **end while**
30: **end for**
31: $\forall S_{h,j}^{ap} \in \mathbb{S}^{ap}, U(S_{h,j}^{ap})_{max} \leftarrow TC_{h,j}/\Delta_{FC}$
32: **Retorna:** $\mathcal{P}_{h,j}^{bk}$ e $U(S_{h,j}^{ap})_{max}$ $\forall S_{h,j}^{ap} \in \mathbb{S}^{ap}$

transferida para o CC no ciclo FlexRay atual (ou seja, a mensagem não sofre *backoff*). Neste caso, TC_p é incrementado em 1, $MS_{counter}$ é incrementado em um número de MS equivalente a C_p^{ap} (Linha 11);

3. Se $rand$ é menor que $\mathcal{P}_p^{bk}$, a transferência da mensagem gerada por S_p^{ap} para o CC será postergada e a decisão sobre a transferência será tomada novamente em outro ciclo FlexRay (ou seja, a mensagem sofre *backoff*). Neste caso, $MS_{counter}$ é incrementado em 1 MS (Linha 14).

- Se não existe um fluxo S_p^{ap} associado com o $FrameID$ atual, $MS_{counter}$ é incrementado em 1 MS (Linhas 18 e 19).

Ao final de cada iteração do laço, $vSlotCounter$ passa a apontar para o próximo *slot* dinâmico.

A simulação do MAC do DN é repetida durante Δ_{FC} ciclos FlexRay. Após o final da simulação, é retornado o número TC_p de transmissões efetuadas por cada fluxo S_p^{ap} em $\mathbb{S}_{sim}^{ap}$.

6.3.4 Definição de $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{hj}$ (Algoritmo 7)

O Algoritmo 5 proposto anteriormente assume um cenário de pior caso no qual todos os fluxos de mensagens aperiódicos em um conjunto $\mathbb{S}^{ap}$ têm uma mensagem pronta para ser transmitida em todos os ciclos FlexRay, e como consequência os *middlewares* de tempo real associados aos fluxos em $\mathbb{S}^{ap}$ estão sempre em modo *backoff*.

Em um sistema real, essa premissa pode ser verdadeira apenas durante curtos intervalos de tempo, ou apenas alguns fluxos de mensagens aperiódicos podem estar com carga total durante situações específicas. Sendo assim, em um sistema real os fluxos em $\mathbb{S}^{ap}$ podem permanecer em modo de operação normal até que ocorra uma condição de sobrecarga, quando então o *middleware* de tempo real relacionado a cada fluxo irá ou não tomar uma decisão sobre a troca de modo de operação.

No Método de Escalonamento Probabilístico do Segmento Dinâmico do FlexRay Descentralizado proposto nesta seção, uma troca para o modo *backoff* é disparada com base no número $\mathcal{N}_{h,j}$ de transmissões consecutivas de mensagens geradas por um fluxo $S_{h,j}^{ap}$, e a mudança para o modo de operação normal é disparada com base no parâmetro $\mathcal{NB}_{h,j}$, que indica o número de ciclos FlexRay que $S_{h,j}^{ap}$ deve permanecer em modo *backoff*.

Algoritmo 6 Simulador FlexRay com Mecanismo de *Backoff*

Entrada: $\mathbb{S}_{sim}^{ap}$, Δ_{FC}, DN_{bus}
Saída: O número TC_p de transmissões de mensagens de cada $S_p^{ap} \in \mathbb{S}_{sim}^{ap}$

1: $\forall S_p^{ap} \in \mathbb{S}_{sim}^{ap}, TC_p \leftarrow 0$
2: **for** $FC_{counter} \leftarrow 1$ to N_{Δ} **do**
3: $vSlotCounter \leftarrow 1$
4: $MS_{counter} \leftarrow 1$
5: **while** $MS_{counter} \leq DN_{bus}$ **do**
6: **if** Existe um $S_p^{ap} \in \mathbb{S}_{sim}^{ap}$ tal que $FID_p^{ap} = vSlotCounter$ **then**
7: **if** $MS_{counter} \geq pLatestTx_p^{ap}$ **then**
8: $MS_{counter} \leftarrow MS_{counter} + 1$
9: **else**
10: $rand \leftarrow$ número aleatório inteiro entre 0 e 100%
11: **if** $rand > \mathcal{P}_p^{bk}$ **then**
12: $TC_p \leftarrow TC_p + 1$
13: $MS_{counter} \leftarrow MS_{counter} + C_p^{ap}$
14: **else**
15: $MS_{counter} \leftarrow MS_{counter} + 1$
16: **end if**
17: **end if**
18: **else**
19: $MS_{counter} \leftarrow MS_{counter} + 1$
20: **end if**
21: $vSlotCounter \leftarrow vSlotCounter + 1$
22: **end while**
23: **end for**
24: **Retorna:** número TC_p de transmissões de mensagens de cada $S_p^{ap} \in \mathbb{S}_{sim}^{ap}$

Será apresentado a seguir um algoritmo (Algoritmo 7) para definir ambos $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$ para cada fluxo de mensagens aperiódico $S^{ap}_{h,j}$ em $\mathbb{S}^{ap}$. O algoritmo utiliza os valores de $\overline{\mathcal{P}^{bk}_{h,j}}$, $\mathcal{P}^{bk}_{h,j}$ e $U(S^{ap}_{h,j})_{max}$, que são definidos utilizando-se o Algoritmo 5.

As entradas do Algoritmo 7 são o conjunto de fluxos de mensagens aperiódicos $\mathbb{S}^{ap}$, um valor inicial arbitrário $\mathcal{N}^{init}_{h,j}$ para $\mathcal{N}_{h,j}$, Δ_{FC}, Λ_{min}, uma tolerância *tol* para Λ_{min} dada em ciclos FlexRay, e DN_{bus}.

O Algoritmo 7 inicia executando o Algoritmo 5 para definir os valores iniciais para $\mathcal{P}^{bk}_{h,j}$ e $U(S^{ap}_{h,j})_{max}$ (Linha 1). A seguir, para todos os fluxos em $\mathbb{S}^{ap}$, $\mathcal{NB}_{h,j}$ é definido como 0, e são definidas duas variáveis auxiliares, $U(S^{ap}_{h,j})_{init}$, que recebe $U(S^{ap}_{h,j})_{max}$ como valor inicial, e $\mathcal{P}^{bk,init}_{h,j}$ que recebe $\mathcal{P}^{bk}_{h,j}$ como valor inicial (Linhas 2 a 6). A função destas variáveis auxiliares é meramente armazenar os valores originalmente definidos para $\mathcal{P}^{bk}_{h,j}$ e $U(S^{ap}_{h,j})_{max}$.

É iniciado então o laço que define $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$ para cada fluxo em $\mathbb{S}^{ap}$ (Linha 7). Os valores para $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$ de um fluxo $S^{ap}_{h,j}$ são definidos sem levar em consideração qualquer outro fluxo em $\mathbb{S}^{ap}$. O raciocínio por trás de tal lógica é que o mecanismo de arbitragem probabilística proposto nesta seção é descentralizado, e a máxima utilização $U(S^{ap}_{h,j})_{max}$ que pode ser causada por $S^{ap}_{h,j}$ sem incorrer na postergação de mensagens com prioridades mais baixas é conhecida *a priori* através da execução do Algoritmo 5. Entretanto, como $\mathcal{P}^{bk}_{h,j}$ é um valor probabilístico, e $U(S^{ap}_{h,j})_{max}$ é obtido com base em uma variável aleatória, os valores iniciais para tais parâmetros podem ser otimistas, e durante sua execução o Algoritmo 7 pode fazer pequenos ajustes nestes valores.

Cada iteração do laço que inicia na Linha 7 executa o Algoritmo 8 para definir valores iniciais para $TC_{h,j}$ e para uma variável auxiliar $U(S^{ap}_{h,j})_{atual}$ que representa a utilização atual de $S^{ap}_{h,j}$. Neste ponto também são definidos valores iniciais para TC_{hp} e para a utilização atual $U(S^{ap}_{hp})_{atual}$ de cada fluxo S^{ap}_{hp} em $hp(S^{ap}_{h,j})$ (Linha 8). O Algoritmo 8 simula um sistema FlexRay que possui o método de arbitragem probabilístico completo, incluindo o mecanismo de troca de modo de operação. Tal algoritmo será discutido na Seção 6.3.5. A partir da Linha 9 é executado um laço aninhado que roda até que i) a variável auxiliar $U(S^{ap}_{h,j})_{atual}$ seja menor que $U(S^{ap}_{h,j})_{max}$ e ii) $TC_{h,j}$ fique dentro de uma faixa de valores dada por Λ_{min} mais ou menos uma tolerância arbitrária.

Em cada iteração do laço aninhado é feito um teste que verifica se $U(S^{ap}_{h,j})_{atual}$ e $TC_{h,j}$ ficam dentro de suas respectivas faixas de valores (Linha 10). Se o resultado de tal teste for verdadeiro o laço aninhado

finaliza, e é iniciado um novo laço para definir os valores para o fluxo $S^{ap}_{h,j+1}$. Se o resultado do teste for falso, o Algoritmo 7 testa diferentes combinações de valores para os parâmetros associados com $S^{ap}_{h,j}$ (da Linha 10 até a Linha 28). Primeiro, o algoritmo tenta incrementar $\mathcal{NB}_{h,j}$ em 1 ciclo FlexRay. Se o novo valor para $\mathcal{NB}_{h,j}$ ultrapassa Δ_{FC} mas $U(S^{ap}_{h,j})_{atual}$ e $TC_{h,j}$ ainda estão fora das faixas de valores desejadas, $\mathcal{N}_{h,j}$ é decrementado em 1 ciclo FlexRay. Se $\mathcal{N}_{h,j}$ se torna menor ou igual a 0, mas $U(S^{ap}_{h,j})_{atual}$ e $TC_{h,j}$ continuam fora da faixa de valores desejadas, $\mathcal{P}^{bk}_{h,j}$ é incrementado em 1%. Se $\mathcal{P}^{bk}_{h,j}$ se torna maior que 100%, $U(S^{ap}_{h,j})_{max}$ é incrementado em 1%. Por fim, se $U(S^{ap}_{h,j})_{max}$ excede 100%, o sistema é considerado não satisfatório.

De forma complementar a definição de $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$, durante cada iteração do laço aninhado que inicia na Linha 9, também é verificado se os valores previamente definidos para $\mathcal{N}_{hp}$ e $\mathcal{NB}_{hp}$ de cada fluxo $S^{FR^{ap}}_{hp}$ em $hp(S^{FR^{ap}}_{hp})$ se mantêm. Se não for possível atingir estes valores, o sistema pode ser considerado como não sendo satisfatório, já que pode ser impossível definir $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$ tais que Λ_{min} seja respeitado. Este teste é executado entre as linhas 29 e 49. No teste, é primeiramente testado se $U(S^{ap}_{hp})_{atual}$ não excede $U(S^{ap}_{hp})_{max}$. Se $U(S^{ap}_{hp})_{max}$ é excedido, é definido um novo valor para $\mathcal{NB}_{hp}$. Novamente, se $\mathcal{NB}_{hp}$ excede Δ_{FC}, $\mathcal{N}_{hp}$ é decrementado em um ciclo FlexRay. Se com o decremento $\mathcal{N}_{hp}$ se torna menor ou igual a 0, é definido um novo valor para $\mathcal{P}^{bk}_{hp}$. Se $\mathcal{P}^{bk}_{hp}$ se torna maior que 100, $U(S^{ap}_{hp})_{max}$ é incrementado. Por fim, se $U(S^{ap}_{hp})_{max}$ ultrapassa 100%, o sistema é considerado não escalonável.

Após a execução dos laços, se o Algoritmo 7 encontra para cada $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$ valores que tornem possível atingir Λ_{min}, o mesmo termina retornando $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$ para cada $S^{ap}_{h,j}$ em $\mathbb{S}^{ap}$.

6.3.5 Simulador FlexRay com troca de modo de operação (Algoritmo 8)

O Algoritmo 7 utiliza o Algoritmo 8, que simula um sistema FlexRay que implementa o Método de Escalonamento Probabilístico do Segmento Dinâmico do FlexRay Descentralizado completo, ou seja, um sistema que implementa o mecanismo de *backoff* e o mecanismo de troca de modo de operação.

O Algoritmo 8 recebe como entradas o fluxo de mensagens aperiódico $S^{ap}_{h,j}$ a ser testado, o conjunto de fluxos de mensagens aperiódicos $\mathbb{S}^{ap}$, um intervalo de tempo Δ_{FC} dado em ciclos FlexRay e o tamanho

Algoritmo 7 Algoritmo para definir $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$

Entrada: $\mathbb{S}^{ap}$, $\mathcal{N}_{h,j}^{init}$, Δ_{FC}, Λ_{min}, tol, DN_{bus}
Saída: $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$ para todo $S_{h,j}^{ap}$ em $\mathbb{S}^{ap}$

```
1: Algoritmo5(𝕊^ap, Δ_FC, Λ_min, DN_bus)
2: for all S^ap_{h,j} ∈ 𝕊^ap do
3:     NB_{h,j} ← 0
4:     U(S^ap_{h,j})_init ← U(S^ap_{h,j})_max
5:     P^{bk,init}_{h,j} ← P^bk_{h,j}
6: end for
7: for all S^ap_{h,j} ∈ 𝕊^ap do
8:     Algoritmo8(S^ap_{h,j}, 𝕊^ap, Δ_FC, DN_bus)
9:     while U(S^ap_{h,j})_atual > U(S^ap_{h,j})_max ou TC_{h,j} <(Λ_min − tol) do
10:        if U(S^ap_{h,j})_atual > U(S^ap_{h,j})_max ou TC_{h,j} <(Λ_min − tol) then
11:            NB_{h,j} = (NB_{h,j} + 1)
12:            if NB_{h,j} > Δ_FC then
13:                N_{h,j} ← (N_{h,j} − 1)
14:                NB_{h,j} ← 1
15:                if N_{h,j} ≤ 0 then
16:                    N_{h,j} ← 0
17:                    NB_{h,j} ← Δ_FC
18:                    P^bk_{h,j} ← (P^bk_{h,j} + 1)
19:                    if P^bk_{h,j} > 100 then
20:                        U(S^ap_{h,j})_max ← (U(S^ap_{h,j})_max + 1)
21:                        if U(S^ap_{h,j})_max > 100% then
22:                            Retorna: sistema não satisfatório
23:                        end if
24:                        P^bk_{h,j} ← P^{bk,init}_{h,j}
25:                    end if
26:                end if
27:            end if
28:        end if
```

29: **for all** $S_{hp}^{ap} \in hp(S_{h,j}^{ap})$ **do**
30: **if** $U(S_{hp}^{ap})_{atual} > U(S_{hp}^{ap})_{max}$ **then**
31: $\mathcal{NB}_{hp} \leftarrow (\mathcal{NB}_{hp} + 1)$
32: **if** $\mathcal{NB}_{hp} > \Delta_{FC}$ **then**
33: $\mathcal{N}_{hp} \leftarrow (\mathcal{N}_{hp} - 1)$
34: $\mathcal{NB}_{hp} \leftarrow 1$
35: **if** $\mathcal{N}_{hp} \leq 0$ **then**
36: $\mathcal{N}_{hp} \leftarrow \mathcal{N}_{hp}^{init}$
37: $\mathcal{NB}_{hp} \leftarrow 0$
38: $\mathcal{P}_{hp}^{bk} \leftarrow (\mathcal{P}_{hp}^{bk} + 1)$
39: **if** $\mathcal{P}_{hp}^{bk} > 100$ **then**
40: $U(S_{hp}^{ap})_{max} \leftarrow (U(S_{hp}^{ap})_{max} + 1)$
41: **if** $U(S_{hp}^{ap})_{max} > 100\%$ **then**
42: **Retorna:** sistema não satisfatório
43: **end if**
44: $\mathcal{P}_{hp}^{bk} \leftarrow \mathcal{P}_{hp}^{bk,init}$
45: **end if**
46: **end if**
47: **end if**
48: **end if**
49: **end for**
50: Algoritmo5($\mathbb{S}^{ap}$, Δ_{FC}, Λ_{min}, DN_{bus})
51: **end while**
52: **end for**
53: **Retorna:** $\mathcal{N}_{h,j}$ e $\mathcal{NB}_{h,j}$ para todo $S_{h,j}^{ap} \in \mathbb{S}^{ap}$

DN_{bus} do segmento Dinâmico. O algoritmo retorna como saídas o número TC_p de transmissões de mensagens e a utilização atual $U(S_p^{ap})$ de cada fluxo de mensagens S_p^{ap} de um subconjunto $\mathbb{S}_{sim}^{ap}$ definido como $\mathbb{S}_{sim}^{ap} = S_{h,j}^{ap} \cup hp(S_{h,j}^{ap})$.

Inicialmente, para cada fluxo S_p^{ap} em $\mathbb{S}_{sim}^{ap}$, o algoritmo define como 0 as seguintes variáveis (Linhas 2 a 4):

- TC_p: contador que indica o número de transmissões de mensagens;
- CMT_p contador que indica o número de transmissões **consecutivas** de mensagens;
- Um contador FCB_p que indica o número atualizado de ciclos FlexRay em modo *backoff*.

Além da inicialização das variáveis acima, uma variável RTM_p que indica o modo de operação do *middleware* de tempo real associado com S_p^{ap} é definida como "normal" (Linha 5).

Assim como no Algoritmo 6, o Algoritmo 8 contém um laço que simula a execução de um sistema FlexRay durante Δ_{FC} ciclos FlexRay. O número do ciclo FlexRay atual é armazenado em $FC_{counter}$ e, a cada iteração do laço, é simulado um mecanismo de arbitragem do DN que contém o mecanismo de *backoff* probabilístico.

No início de cada iteração, o contador $vSlotCounter$ (que armazena o número do *slot* dinâmico atual) e o contador de *minislots* $MS_{counter}$ são definidos como 0. É então iniciado outro laço onde $MS_{counter}$ é incrementado de 1 a DN_{bus}. Neste laço, pode ocorrer uma das seguintes ações:

- Se existe um fluxo S_p^{ap} em $\mathbb{S}_{sim}^{ap}$ associado com o $FrameID$ atual ($FID_p^{ap} = vSlotCounter$), é verificado se $pLatestTx_p^{ap}$ é respeitado. Se $pLatestTx_p^{ap}$ não é respeitado, $MS_{counter}$ é incrementado em 1 MS e CMT_p é definido como 0 (Linhas 11 a 12). Se $pLatestTx_p^{ap}$ é respeitado (Linha 14), é simulado o mecanismo de *backoff* probabilístico. Durante essa simulação:

  1. Se RTM_p^{ap} está no modo normal, uma mensagem é transferida para o CC, $MS_{counter}$ é incrementado de C_p^{ap} MS, TC_p e CMT_p são incrementados de 1. No entanto, se CMT_p ultrapassar $\mathcal{N}_p$, o *middleware* de tempo real RTM_p^{ap} muda para o modo *backoff* (ou seja, RTM_p^{ap} detectou o envio de rajadas de mensagens por parte de S_p^{ap}) (Linhas 15 a 22).

Neste caso, as variáveis CMT_p e FCB_p são reiniciadas para zero.

2. Se RTM_p^{ap} está em modo *backoff*, é tomada uma das seguintes ações (Linhas 24 a 36):
   - FCB_p é incrementado de 1. Se FCB_p se torna maior que $\mathcal{NB}_p$, RTM_p^{ap} é definido como "normal", significando que no próximo ciclo FlexRay o *middleware* de tempo real associado com S_p^{ap} irá operar em modo normal;
   - Uma variável *rand* recebe um valor aleatório inteiro entre 0 e 100% (Linha 29);
   - Se *rand* for maior que $\mathcal{P}_p^{bk}$ a mensagem gerada por S_p^{ap} é transferida para o CC no ciclo FlexRay atual (ou seja, a mensagem não sofre *backoff*). Neste caso, TC_p é incrementado de 1 e $MS_{counter}$ é incrementado de C_p^{ap} MS (Linhas 30 a 32).
   - Se *rand* for menor que $\mathcal{P}_p^{bk}$, a mensagem gerada por S_p^{ap} será postergada e uma nova decisão sobre a transferência para o CC será tomada no ciclo FlexRay seguinte. Neste caso, $MS_{counter}$ é incrementado de 1 MS (Linhas 33 a 35).
3. O valor de $U(S_{h,j}^{ap})_{atual}$ é atualizado.

- Se não existe um fluxo em $\mathbb{S}_{sim}^{ap}$ associado com o $FrameID$ atual, $MS_{counter}$ é incrementado em 1 MS.

No final de cada iteração do laço, $vSlotCounter$ passa a apontar para o próximo *slot* estático

Após o final da simulação, o algoritmo retorna TC_p e $U(S_p^{ap})$ de cada fluxo em $\mathbb{S}_{sim}^{ap}$.

6.3.6 Exemplo Numérico

Será apresentado a seguir um exemplo numérico que ilustra o uso do Método de Escalonamento Probabilístico do Segmento Dinâmico do FlexRay Descentralizado.

O exemplo considera um sistema FlexRay executando durante um intervalo de tempo Δ_{FC} correspondente a 1000 ciclos FlexRay, e o conjunto de parâmetros para um sistema FlexRay apresentado na

Algoritmo 8 Simulador FlexRay com mecanismo de *backoff* e troca de modo de operação

Entrada: $S^{ap}_{h,j}, \mathbb{S}^{ap}, \Delta_{FC}, DN_{bus}$
Saída: TC_p e $U(S^{ap}_p)$ para cada $S^{ap}_p \in \mathbb{S}^{ap}_{sim}$

$\mathbb{S}^{ap}_{sim} = S^{ap}_{h,j} \cup hp(S^{ap}_{h,j})$
$\forall S^{ap}_p \in \mathbb{S}^{ap}_{sim}, TC_p \leftarrow 0$
$\forall S^{ap}_p \in \mathbb{S}^{ap}_{sim}, CMT_p \leftarrow 0$
$\forall S^{ap}_p \in \mathbb{S}^{ap}_{sim}, FCB_p \leftarrow 0$
$\forall S^{ap}_p \in \mathbb{S}^{ap}_{sim}, RTM^{ap}_p \leftarrow Normal$
for $FC_{counter} \leftarrow 1$ to Δ_{FC} **do**
    $vSlotCounter \leftarrow 1$
    $MS_{counter} \leftarrow 1$
    while $MS_{counter} \leq DN_{bus}$ **do**
        if existe $S^{ap}_p \in \mathbb{S}^{ap}_{sim}$ tal que $FID^{ap}_p = vSlotCounter$ **then**
            if $MS_{counter} \geq pLatestTx^{ap}_p$ **then**
                $CMT_p \leftarrow 0$
                $MS_{counter} \leftarrow (MS_{counter} + 1)$
            else
                if $RTM^{ap}_p = Normal$ **then**
                $TC_p \leftarrow (TC_p + 1)$
                $CMT_p \leftarrow (CMT_p + 1)$
                if $CMT_p \geq \mathcal{N}_p$ **then**
                $RTM^{ap}_p \leftarrow Backoff$
                $CMT_p \leftarrow 0$
                $FCB_p \leftarrow 0$
                end if
                $MS_{counter} \leftarrow (MS_{counter} + C^{ap}_p)$
                else
                $FCB_p \leftarrow (FCB_p + 1)$
                if $FCB_p \geq \mathcal{NB}_p$ **then**
                $RTM^{ap}_p \leftarrow Normal$
                end if
                $rand \leftarrow$ número aleatório inteiro entre 1 e 100%
                if $rand > \mathcal{P}^{bk}_p$ **then**

$TC_p \leftarrow (TC_p + 1)$
$MS_{counter} \leftarrow (MS_{counter} + C_l)$
else
$MS_{counter} \leftarrow (MS_{counter} + 1)$
end if
end if
$U(S_p^{ap})_{atual} \leftarrow (TC_p / \Delta_{FC})$
end if
else
$MS_{counter} \leftarrow (MS_{counter} + 1)$
$vSlotCounter \leftarrow (vSlotCounter + 1)$
end if
$vSlotCounter \leftarrow (vSlotCounter + 1)$
end while
end for
Retorna: TC_p e $U(S_p^{ap})$ para cada $S_p^{ap} \in \mathbb{S}_{sim}^{ap}$

Parâmetro	**Valor**
FC_{bus}	725 MS
ST_{bus}	435 MS
DN_{bus}	290 MS
Δ_{FC}	1000 FCs
Λ_{min}	120 FCs
$\mathcal{N}_{h,j}^{init}$	10 FCs
Tolerância	20 FCs

Tabela 15 – Parâmetros FlexRay para o exemplo numérico

Tabela 15. Os valores apresentados na Tabela 15 são baseados nos parâmetros apresentados em (SCHEDL, 2007).

O exemplo também considera o conjunto de fluxos de mensagens aperiódicos $\mathbb{S}^{ap}$ apresentado nas Tabelas 16 e 17. Este conjunto foi gerado utilizando-se o *software* Netcarbench (BRAUN et al., 2007). Ambos $C^{ap}_{h,j}$ e $pLatestTx^{ap}_{h,j}$ são dados em múltiplos inteiros de *minislots*. Neste exemplo, considera-se que os fluxos em $\mathbb{S}^{ap}$ sempre possuem uma mensagem pronta para ser transferida para o CC.

Na Figura 44 é apresentado o número de transmissões de mensagens, dentro do intervalo de tempo $\Delta_{FC} = 1000$ FCs, quando não é utilizado um mecanismo de *backoff*. Pode-se observar que os fluxos com alta prioridade sempre transmitem, com uma utilização de 100%, enquanto os fluxos com baixas prioridades nunca conseguem acesso ao barramento, sendo postergados indefinidamente.

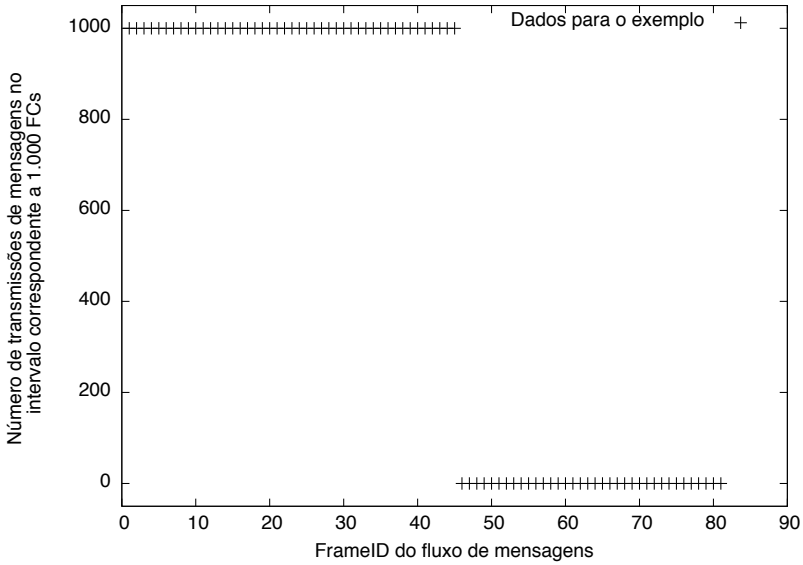

Figura 44 – Exemplo numérico: número de transmissões sem mecanismo de *backoff*

Na Figura 45 é apresentado o número de transmissões de mensagens, dentro do intervalo de tempo $\Delta_{FC} = 1000$ FCs, quando é utilizado o mecanismo de *backoff* proposto nesta seção. Apesar de os fluxos

Fluxo	N_h	$C_{h,j}^{ap}$	$FrameID$	$pLatestTx_{h,j}^{ap}$
$S_{1,1}^{ap}$	1	7	1	284
$S_{1,2}^{ap}$	1	7	2	284
$S_{1,3}^{ap}$	1	7	3	284
$S_{2,1}^{ap}$	2	7	4	283
$S_{2,2}^{ap}$	2	7	4	283
$S_{2,3}^{ap}$	2	8	5	283
$S_{2,4}^{ap}$	2	5	6	283
$S_{3,1}^{ap}$	3	8	7	283
$S_{3,2}^{ap}$	3	6	8	283
$S_{3,3}^{ap}$	3	8	9	283
$S_{3,4}^{ap}$	3	6	10	283
$S_{3,5}^{ap}$	3	6	11	283
$S_{3,6}^{ap}$	3	7	12	283
$S_{3,7}^{ap}$	3	8	13	283
$S_{3,8}^{ap}$	3	8	14	283
$S_{3,9}^{ap}$	3	8	15	283
$S_{3,10}^{ap}$	3	8	16	283
$S_{4,1}^{ap}$	4	6	17	283
$S_{4,2}^{ap}$	4	5	18	283
$S_{4,3}^{ap}$	4	7	19	283
$S_{4,4}^{ap}$	4	5	20	283
$S_{4,5}^{ap}$	4	8	21	283
$S_{4,6}^{ap}$	4	8	22	283
$S_{4,7}^{ap}$	4	6	23	283
$S_{4,8}^{ap}$	4	4	24	283
$S_{4,9}^{ap}$	4	6	25	283
$S_{4,10}^{ap}$	4	5	26	283
$S_{4,11}^{ap}$	4	5	27	283
$S_{5,1}^{ap}$	5	4	28	284
$S_{5,2}^{ap}$	5	3	29	284
$S_{5,3}^{ap}$	5	7	30	284
$S_{5,4}^{ap}$	5	2	31	284
$S_{5,5}^{ap}$	5	6	32	284
$S_{6,1}^{ap}$	6	3	33	283
$S_{6,2}^{ap}$	6	7	34	283
$S_{6,3}^{ap}$	6	6	35	283
$S_{6,4}^{ap}$	6	5	36	283
$S_{6,5}^{ap}$	6	7	37	283
$S_{6,6}^{ap}$	6	8	38	283
$S_{6,7}^{ap}$	6	8	39	283
$S_{6,8}^{ap}$	6	7	40	283

Tabela 16 – Conjunto de fluxos de mensagens aperiódicos $\mathbb{S}^{ap}$ para o exemplo numérico

Fluxo	N_h	$C_{h,j}^{ap}$	$FrameID$	$pLatestTx_{h,j}^{ap}$
$S_{7,1}^{ap}$	7	8	41	283
$S_{7,2}^{ap}$	7	7	42	283
$S_{7,3}^{ap}$	7	8	43	283
$S_{7,4}^{ap}$	7	7	44	283
$S_{7,5}^{ap}$	7	8	45	283
$S_{7,6}^{ap}$	7	8	46	283
$S_{7,7}^{ap}$	7	3	47	283
$S_{8,1}^{ap}$	8	7	48	283
$S_{8,2}^{ap}$	8	5	49	283
$S_{8,3}^{ap}$	8	8	50	283
$S_{8,4}^{ap}$	8	8	51	283
$S_{8,5}^{ap}$	8	8	52	283
$S_{8,6}^{ap}$	8	7	53	283
$S_{8,7}^{ap}$	8	8	54	283
$S_{8,8}^{ap}$	8	8	55	283
$S_{8,9}^{ap}$	8	8	56	283
$S_{8,10}^{ap}$	8	2	57	283
$S_{8,11}^{ap}$	8	5	58	283
$S_{9,1}^{ap}$	9	7	59	283
$S_{9,2}^{ap}$	9	6	60	283
$S_{9,3}^{ap}$	9	7	61	283
$S_{9,4}^{ap}$	9	8	62	283
$S_{9,5}^{ap}$	9	4	63	283
$S_{10,1}^{ap}$	10	8	64	283
$S_{10,2}^{ap}$	10	8	65	283
$S_{10,3}^{ap}$	10	3	66	283
$S_{10,4}^{ap}$	10	8	67	283
$S_{10,5}^{ap}$	10	2	68	283
$S_{10,6}^{ap}$	10	6	69	283
$S_{10,7}^{ap}$	10	8	70	283
$S_{10,8}^{ap}$	10	6	71	283
$S_{10,9}^{ap}$	10	8	72	283
$S_{10,10}^{ap}$	10	8	73	283
$S_{11,1}^{ap}$	11	7	74	284
$S_{11,2}^{ap}$	11	1	75	284
$S_{11,3}^{ap}$	11	6	76	284
$S_{11,4}^{ap}$	11	5	77	284
$S_{11,5}^{ap}$	11	6	78	284
$S_{12,1}^{ap}$	12	3	79	283
$S_{12,2}^{ap}$	12	5	80	283
$S_{12,3}^{ap}$	12	8	81	283

Tabela 17 – Conjunto de fluxos de mensagens aperiódicos $\mathbb{S}^{ap}$ para o exemplo numérico - continuação

com $FrameIDs$ baixos (e portanto, prioridades altas dentro do DN) continuarem com números de transmissões maiores que os fluxos com $FrameIDs$ mais altos, pode-se observar que nenhum dos fluxos sofre postergação indefinida, ou seja, todos os fluxos tiveram um mínimo de oportunidades de transmissão no intervalo de tempo, o que está de acordo com a proposta apresentada nesta seção.

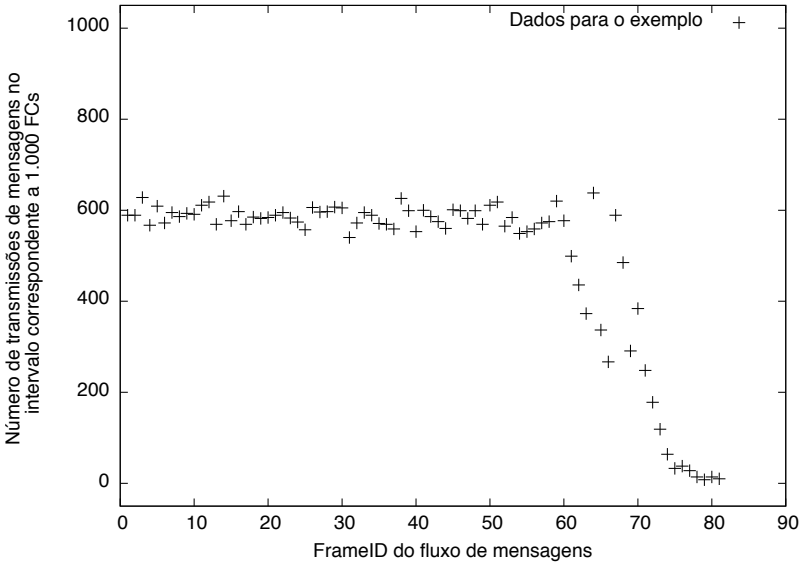

Figura 45 – Exemplo numérico: número de transmissões, para os fluxos de mensagens em $\mathbb{S}^{ap}$, utilizando o mecanismo de *backoff* proposto

De forma complementar à Figura 45, a Figura 46 apresenta as probabilidades de *backoff* para cada fluxo de mensagens em $\mathbb{S}^{ap}$.

Pode-se observar que os fluxos de mensagens com $FrameIDs$ entre 63 e 74 possuem probabilidades de *backoff* mais alta que a média dos fluxos com $FrameIDs$ mais baixos, enquanto as probabilidades de *backoff* dos fluxos com $FrameIDs$ acima de 75 retornam ao patamar anterior. Para os fluxos com $FrameIDs$ 80 e 81 a probabilidade de *backoff* é 0. Este comportamento pode ser explicado pela natureza do MAC do DN e pelo algoritmo utilizado para definir os parâmetros relacionados a cada fluxo, conforme explicado a seguir.

Devido ao mecanismo de arbitragem do DN, os fluxos de men-

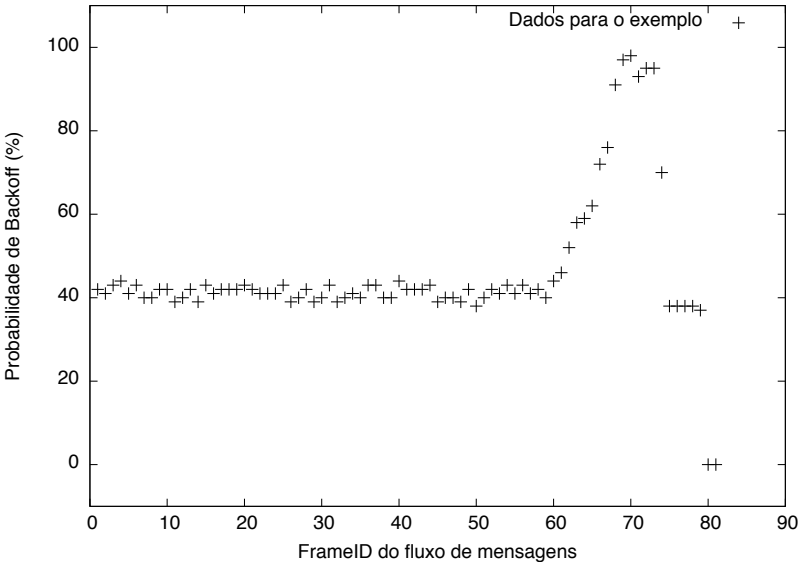

Figura 46 – Exemplo numérico: probabilidades de *backoff* para os fluxos de mensagens em $\mathbb{S}^{ap}$

sagens que transmitem nesse segmento são gulosos, isto é, se houver uma mensagem pronta para ser transmitida, e houver a possibilidade de transmitir, essa mensagem sempre será transmitida.

O Método de Escalonamento Probabilístico do Segmento Dinâmico do FlexRay Descentralizado proposto neste capítulo garante que todos os fluxos de mensagens terão um mínimo de oportunidades de transmissão dentro de um intervalo Δ_{FC} predefinido, sem se preocupar com a distribuição destas oportunidades. A partir do momento em que todos os fluxos possuem uma probabilidade de *backoff* que garante o mínimo de oportunidades de transmissão para fluxos com prioridades mais baixas, o algoritmo encerra, sem tentar igualar todas as probabilidades de *backoff*.

6.4 CONTRIBUIÇÕES PARA O ESTADO DA ARTE

O FlexRay é um sistema de comunicação para uso em aplicações automotivas baseado em um ciclo de comunicação que contém, além de dois segmentos de controle, um segmento (chamado Segmento Estático) projetado para transmitir mensagens geradas por fluxos de mensagens periódicos com restrições temporais rígidas (*hard real-time*), e um segmento (chamado Segmento Dinâmico) projetado para transmitir mensagens geradas por fluxos ativados por eventos (*event-triggered*).

Apesar do Segmento Dinâmico ser projetado para transmitir mensagens geradas por uma classe variada de fluxos de mensagens, classe esta que inclui deste fluxos de mensagens esporádicos com características temporais rígidas até fluxos de mensagens aperiódicos sem restrições temporais, como apresentado no Capítulo 3 a maioria dos trabalhos existentes considera modelos de sistema que contêm apenas fluxos esporádicos com características temporais rígidas.

Entretanto, um sistema veicular também possui aplicações com características de *soft real-time*, ou que não possuem restrições temporais. Por exemplo, fluxos de mensagens associados com sistemas de manutenção são geralmente aperiódicos. Escalonar fluxos de mensagens aperiódicos no Segmento Dinâmico assumindo artificialmente que os mesmos possuem características de *hard real-time* pode resultar em subutilização da rede, já que é necessário garantir que as restrições de tempo real de tais fluxos sejam respeitadas através da alocação de recursos para os mesmos. Por outro lado, simplesmente atribuir baixas prioridades do DN para os fluxos aperiódicos, sem considerar o efeito dos fluxos com prioridades mais altas, pode fazer com que as mensagens

geradas por estes fluxos sejam postergadas indefinidamente. Devido às características do Segmento Dinâmico, mensagens geradas por fluxos de mensagens associados com prioridades baixas têm chances menores de serem transmitidas.

Neste capítulo foi abordada a questão do escalonamento de fluxos de mensagens aperiódicos no Segmento Dinâmico do FlexRay. Foram apresentadas duas propostas para métodos de arbitragem do DN que tiram vantagem da flexibilidade que fluxos aperiódicos possuem em relação a restrições de tempo real.

Nos mecanismos propostos, os fluxos de mensagens aperiódicos de um sistema são associados com uma *probabilidade de backoff* e um *middleware* de tempo real específico utiliza a probabilidade de *backoff* para definir se uma mensagem gerada por um fluxo aperiódico irá competir ou não pelo barramento no ciclo de comunicação atual, influenciando nas chances que mensagens com prioridades mais baixas têm de serem transmitidas.

A primeira proposta, chamada Método de Escalonamento Probabilístico para o Segmento Dinâmico do FlexRay e apresentada na Seção 6.2, considera um sistema onde a decisão pela entrada no chamado modo *backoff* é tomada de forma global. Já a segunda proposta, chamada Método de Escalonamento Probabilístico do Segmento Dinâmico do FlexRay Descentralizado e apresentada na Seção 6.3, utiliza um método descentralizado, onde cada fluxo de mensagens aperiódico está associado com um *middleware* de tempo real exclusivo. Cada *middleware* de tempo real decide localmente, com base nas características do fluxo associado, se opera em modo normal ou em modo *backoff*. O modo de operação de cada *middleware* é independente do modo de operação dos demais *middlewares* de tempo real.

Para cada uma das propostas descritas neste capítulo foram apresentados exemplos numéricos ilustrando seu uso.

É importante salientar que o objetivo dos métodos propostos é aumentar a chance de transmissão de fluxos de mensagens aperiódicos que transmitem no segmento Dinâmico do FlexRay. Como aumentam as chances de transmissão de mensagens que antes seriam potencialmente postergadas indefinidamente, possivelmente o atraso médio de transmissão das mensagens aperiódicas será melhorado. No entanto, o atraso médio não é uma das métricas exploradas neste trabalho.

Futuras extensões destas propostas podem incluir novos métodos para a definição dos parâmetros probabilísticos de cada fluxo de mensagens aperiódicos. Estes métodos poderiam utilizar, por exemplo, técnicas de otimização como Programação Linear. Uma questão que

permanece em aberto é a análise do custo computacional do *middleware* de tempo real necessário para esta proposta, e seu impacto em uma implementação real. Esta questão também pode vir a ser explorada como uma extensão deste trabalho.

7 CONCLUSÃO

Este trabalho se insere na área de protocolos de tempo real, abordando especificamente o Sistema de Comunicação FlexRay, um protocolo de tempo real para usos automotivos. Os objetos de estudo deste trabalho foram os mecanismos de escalonamento de fluxos de mensagens para o FlexRay, bem como as técnicas utilizadas na análise de tempo de resposta em sistemas que utilizam tal protocolo. O levantamento do estado da arte relacionado ao FlexRay mostrou que existem pontos relacionados ao assunto que podem ser explorados de forma a propor contribuições originais.

Desta forma, o objetivo geral desta tese foi a elaboração e a avaliação de mecanismos para o escalonamento e análise de tempo de resposta de sistemas que utilizem o Sistema de Comunicação FlexRay. Para atingir os objetivos específicos foi realizada uma revisão crítica da literatura sobre o protocolo FlexRay. Esta revisão resultou na definição dos seguintes objetivos específicos a serem perseguidos:

1. Desenvolver um mecanismo para escalonar, no segmento Estático do FlexRay, fluxos de mensagens cujos períodos não são múltiplos inteiros do ciclo de comunicação e cuja geração de mensagens não está sincronizada com o ciclo de comunicação;

2. Desenvolver um mecanismo para escalonar, no segmento Dinâmico do FlexRay, fluxos de mensagens que não possuem características temporais rígidas, como, por exemplo, fluxos de mensagens aperiódicos.

O primeiro objetivo específico desta tese foi motivado pelo fato de que as propostas da literatura que abordam o segmento Estático do FlexRay, em geral, consideram modelos de sistema restritivos, onde os períodos dos fluxos de mensagens são múltiplos inteiros do tamanho do ciclo de comunicação. Além disso, parte considerável das propostas existentes considera que a geração de mensagens pelos fluxos está sincronizada com o ciclo FlexRay. Entretanto, muitas vezes estas restrições são impostas de forma artificial, por exemplo, reduzindo o período de um ciclo para que o mesmo se torne um múltiplo inteiro do ciclo FlexRay. Mas este tipo de ação pode resultar em uma subutilização do sistema, pois é necessário alocar recursos para garantir sua escalonabilidade.

Já o segundo objetivo específico teve como motivação o fato de que, assim como ocorre com o segundo objetivo específico, grande parte

da literatura adota modelos de sistema restritivos para propostas relacionadas ao segmento Dinâmico do FlexRay. Apesar do segmento Dinâmico ser projetado para acomodar a transmissão de mensagens geradas por fluxos esporádicos com características de tempo real crítico, fluxos esporádicos com características de tempo real não críticas, e fluxos aperiódicos que não possuem restrições de tempo real, a literatura em geral considera modelos de sistema onde os fluxos de mensagens possuem características de tempo real crítico.

O resultado final deste trabalho materializou-se na proposição, implementação e avaliação de duas propostas de métodos para escalonamento no segmento Estático do FlexRay, e duas propostas para o escalonamento no segmento Dinâmico do FlexRay.

As duas propostas para métodos para o escalonamento do segmento Estático do FlexRay são apresentadas no Capítulo 5. No capítulo foi demonstrada a viabilidade de se definir a alocação de *slots* estáticos para cada nodo utilizando técnicas tradicionais para a análise de tempo de resposta, considerando-se os requisitos temporais impostos pelo conjunto de fluxos de mensagens de cada nodo. Tais métodos são chamados respectivamente de Método de Alocação Proporcional e Método de Alocação Proporcional Adaptativo. Ambos são capazes de considerar conjuntos de fluxos com períodos que não são múltiplos de FC, sendo, também, capazes de considerar o caso em que a geração de mensagens nos fluxos não está sincronizada com o FC. No capítulo foi explicado como as técnicas propostas podem ser integradas em um *middleware* de tempo real em cada nodo, tirando vantagem dos métodos de alocação propostos. Também foi explicado como o Método de Alocação Proporcional proposto pode ser integrado em um ambiente AUTOSAR permitindo seu uso em implementações FlexRay reais, e apresentada uma comparação entre o Método de Alocação Proporcional Adaptativo (APAS) e uma das técnicas do estado da arte sobre escalonamento no segmento estático do FlexRay que utiliza ILP. Tal comparação demonstra que o APAS é mais eficaz e requer um número menor de *slots* para um mesmo conjunto de fluxos do que o método do estado da arte, o que pode ser explicado pelo fato de que o padrão de transmissões é diferente, sendo o APAS mais flexível em relação à utilização dos *slots* estáticos alocados.

As propostas relacionadas ao segmento Dinâmico do FlexRay são descritas no Capítulo 6 e abordam a questão do escalonamento de fluxos de mensagens aperiódicos no Segmento Dinâmico do FlexRay. Foram apresentados dois mecanismos para métodos de arbitragem do DN que tiram vantagem da flexibilidade que fluxos aperiódicos pos-

suem em relação a restrições de tempo real. Em ambos os mecanismos, os fluxos de mensagens aperiódicos de um sistema são associados com uma probabilidade de *backoff* e um *middleware* de tempo real específico utiliza a probabilidade de *backoff* para definir se uma mensagem gerada por um fluxo aperiódico irá competir ou não pelo barramento no ciclo de comunicação atual, influenciando nas chances que mensagens com prioridades mais baixas têm de serem transmitidas. No primeiro mecanismo proposto, chamado Método de Escalonamento Probabilístico para o Segmento Dinâmico do FlexRay, a decisão pela entrada no modo de arbitragem que utiliza as probabilidades de *backoff* de cada fluxo de mensagens é tomada de forma centralizada. O segundo mecanismo proposto, chamado Método de Escalonamento Probabilístico do Segmento Dinâmico do FlexRay Descentralizado, utiliza um método descentralizado onde cada fluxo de mensagens aperiódico está associado com um *middleware* de tempo real exclusivo. Cada *middleware* de tempo real decide localmente, com base nas características do fluxo associado, se opera em modo normal ou em modo *backoff*. O modo de operação de cada *middleware* é independente do modo de operação dos demais *middlewares* de tempo real. Para ambas as propostas descritas foram apresentados exemplos numéricos ilustrando seu uso.

Nas Tabelas 18 e 19 são resumidas as contribuições desta Tese para o estado da arte relacionado ao protocolo FlexRay.

Para apresentar o trabalho produzido à comunidade científica de sistemas de tempo real, foram produzidos artigos. Tais artigos foram submetidos para revisão em simpósios nacionais e internacionais. Foram publicados um capítulo de livro e quatro artigos, sendo um artigo em em simpósio nacional, dois em congressos internacionais e um em periódico internacional. As publicações resultantes deste trabalho são listadas a seguir:

1. LANGE, R. et al. *Guaranteeing Real-Time Message Deadlines in the FlexRay Static Segment Using a On-Line Scheduling Approach*. In: Proceedings of the 9th IEEE International Workshop on Factory Communication Systems. Lemgo/Detmold, Germany: IEEE, 2012. p. 301-310. ISBN 978-1-4673-0692-8.

2. LANGE, R.; OLIVEIRA, R. S. de. *Hybrid Flexray/CAN Automotive Networks*. Embedded Computing Systems: Applications, Optimization, and Advanced Design, IGI Global, p. 323-342, 2013. doi:10.4018/978-1-4666-3922-5.

3. LANGE, R.; OLIVEIRA, R. S. de. *Probabilistic Scheduling of the*

Proposta	**Capítulo e página**	**Resumo**	**Publicado em**
Revisão bibliográfica	Capítulo 2 e Capítulo 3	Capítulo de livro apresentando o estado da arte sobre sistemas automotivos FlexRay/CAN Híbridos	(LANGE; OLIVEIRA, 2013)
Método de Alocação Proporcional	Capítulo 5, Seção 5.2, página 91	Método para o escalonamento do ST que utiliza técnicas tradicionais para a análise de tempo de resposta na definição da alocação de *slots* estáticos.	(LANGE et al., 2012)
Método de Alocação Proporcional Adaptativo	Capítulo 5, Seção 5.3, página 111	Extensão do Método de Alocação Proporcional que considera um *freeze instant* por *slot* estático.	(LANGE et al., 2015)
Método de Escalonamento Probabilístico Para o Segmento Dinâmico do FlexRay	Capítulo 6, Seção 6.2, página 138	Método probabilístico para o escalonamento de fluxos de mensagens aperiódicos no DN	(LANGE; OLIVEIRA, 2014)

Tabela 18 – Resumo das contribuições para o estado da arte relacionado ao protocolo FlexRay

Proposta	Capítulo e página	Resumo	Publicado em
Método de Escalonamento Probabilístico do Segmento Dinâmico do FlexRay Descentralizado	Capítulo 6, Seção 6.3, página 148	Método probabilístico descentralizado para o escalonamento de fluxos de mensagens aperiódicos no DN. Extensão do método proposto em (LANGE; OLIVEIRA, 2014)	

Tabela 19 – Resumo das contribuições para o estado da arte relacionado ao protocolo FlexRay - continuação

FlexRay Dynamic Segment. In: 12th IEEE International Conference on Industrial Informatics. Porto Alegre: [s.n.], 2014.

4. LANGE, R. et al. *A scheme for slot allocation of the FlexRay static segment based on response time analysis*. Computer Communications, 2015. ISSN 01403664. Qualis A1 na Engenharias IV.

Considera-se que as propostas apresentadas nesta Tese podem ser estendidas de forma a gerar novas contribuições para o estado da arte. As propostas para o segmento estático apresentadas no Capítulo 5 podem incluir novos métodos para a definição da alocação de *slots* estáticos. Estes métodos poderiam utilizar, por exemplo, técnicas de otimização como Programação Linear (LP).

Já os métodos para o escalonamento probabilístico do segmento Dinâmico, apresentados no Capítulo 6, podem incluir novos mecanismos para a definição dos parâmetros probabilísticos de cada fluxo de mensagens aperiódico. Estes métodos também poderiam utilizar, por exemplo, técnicas de otimização como Programação Linear.

Por fim, todas as propostas apresentadas neste trabalho podem ser implementadas em ambientes compostos por *hardwares* reais, de forma a comparar os resultados obtidos nas simulações com resultados obtidos com aqueles obtidos em implementações reais.

REFERÊNCIAS

AGRAWAL, G.; CHEN, B.; ZHAO, W.; DAVARI, S. Guaranteeing Synchronous Message Deadlines With the Timed Token Medium Access Control Protocol. **IEEE Transactions on Computers**, IEEE Computer Society, v. 43, n. 3, p. 327–339, mar. 1994. ISSN 00189340.

ALMEIDA, L.; FONSECA, J. A.; FONSECA, P. Flexible Time Triggered Communication System Based on the Controller Area Network: Experimental Results. In: **Fieldbus Technology International Conference (FeT 1999)**. [S.l.: s.n.], 1999. v. 99, p. 24–31.

ANDERSSON, B.; TOVAR, E. The utilization bound of non-preemptive rate-monotonic scheduling in Controller Area Networks is 25%. In: **Proceedings of the International Symposium on Industrial Embedded Systems (SIES'09)**. Lausane, Switzerland: IEEE, 2009. p. 11–18. ISBN 978-1-4244-4109-9.

AUDSLEY, N.; BURNS, A.; RICHARDSON, M.; TINDELL, K.; WELLINGS, A. J. Applying New Scheduling Theory to Static Priority Pre-Emptive Scheduling. **Software Engineering Journal**, v. 8, p. 284–292, 1993.

AUTOSAR, G. R. **AUTOSAR Specification Release 4.0 Rev. 3.0**. 2012. Disponível em: <http://www.autosar.org/>.

BONET, E.; OLIVEIRA, R. S. de; LANGE, R. Discussion on the Usage of Real-Time Calculus to Model the FlexRay-Bus. **2011 Brazilian Symposium on Computing System Engineering**, IEEE, p. 83–85, 2011.

Bosch. **Controller Area Network Specification v2.0**. 1991. Disponível em: <http://www.bosch.de>.

Bosch. **CAN with Flexible Data-rate**. 2015. Disponível em: <http://www.bosch-semiconductors.de/>.

BRAUN, C.; HAVET, L.; NAVET, N. NETCARBENCH: A Benchmark for Techniques and Tools used in the Design of Automotive Communication Systems. In: **Proceedings of the 7th**

IFAC International Conference on Fieldbuses & Networks in Industrial & Embedded Systems - FeT 2007. Toulouse, France: [s.n.], 2007. p. 321–328.

BURNS, A.; WELLINGS, A. J. **Real-Time Systems and Programming Languages: Ada 95, Real-Time Java, and Real-Time POSIX**. Third. [S.l.]: Addison Wesley, 2001.

CENA, G.; VALENZANO, V. On the Properties of the Flexible Time Division Multiple Access Technique. **IEEE Transactions on Industrial Informatics**, v. 2, n. 2, p. 86–94, maio 2006. ISSN 1551-3203.

CHOKSHI, D. B.; BHADURI, P. Performance Analysis of FlexRay-based Systems Using Real-time Calculus, Revisited. In: **Proceedings of the 2010 ACM Symposium on Applied Computing**. New York, NY, USA: ACM, 2010. (SAC '10), p. 351–356. ISBN 978-1-60558-639-7.

DAVIS, R. I.; BURNS, A.; BRIL, R. J.; LUKKIEN, J. J. Controller Area Network (CAN) Schedulability Analysis: Refuted, Revisited and Revised. **Real-Time Systems**, v. 35, n. 3, p. 239–272, jan. 2007. ISSN 0922-6443.

DING, S.; TOMIYAMA, H.; TAKADA, H. An Effective GA-Based Scheduling Algorithm for FlexRay Systems. **IEICE Transactions on Information and Systems**, E91-D, n. 8, p. 2115–2123, ago. 2008. ISSN 0916-8532.

DVORAK, J.; HANZALEK, Z. FlexRay Static Segment Scheduling on Two Independent Channels With Gateway. In: IEEE. **Proceedings of 2015 IEEE World Conference on Factory Communication Systems (WFCS)**. [S.l.], 2015. p. 1–4.

FLEXRAY. **ISO 17458: Road Vehicles - FlexRay Communications System**. 2015. Disponível em: <http://www.iso.org/>.

GEORGE, L.; RIVIERRE, N.; SPURI, M. Preemptive and Non-Preemptive Real-Time Uniprocessor Scheduling. In: **Technical Report, INRIA**. [S.l.]: INRIA, 1996.

GLPK. **GNU Linear Programming Kit**. 2014. Disponível em: <http://www.gnu.org/software/glpk/>.

GRENIER, M.; HAVET, L.; NAVET, N. Configuring the Communication on FlexRay - The Case of the Static Segment. In: **Proceedings of the 4th European Congress Embedded Real Time Software (ERTS)**. Toulouse, France: [s.n.], 2008.

GROW, R. M. **Timed Token Protocol for Local Area Networks**. jul. 10 1984. US Patent 4,459,588.

HAGIESCU, A.; BORDOLOI, U. D.; CHAKRABORTY, S.; SAMPATH, P.; GANESAN, P. V. V.; RAMESH, S. Performance Analysis of FlexRay-Based ECU Networks. In: **The Design Automation Conference (DAC)**. [S.l.]: ACM Press New York, NY, USA, 2007. p. 284–289.

HANZáLEK, Z.; BENES, D.; WARAUS, D. Time Constrained FlexRay Static Segment Scheduling. In: **Proceedings of the 10th International Workshop on Real-Time Networks**. Porto, Portugal: [s.n.], 2011. p. 61–67.

IEEE. **Institute of Electrical and Electronics Engineers**. 2015. Disponível em: <http://www.ieee.org/>.

ISO11898-1. **ISO 1189811: Road Vehicles - Interchange of Digital Information - Controller Area Network (CAN) for High-Speed Communication**. International Organization for Standardization, 2003. Disponível em: <http://www.iso.org/>.

KANG, M.; PARK, K.; JEONG, M. K. Frame Packing for Minimizing the Bandwidth Consumption of the FlexRay Static Segment. **IEEE Transactions on Industrial Electronics**, v. 60, n. 9, p. 4001–4008, 2012. ISSN 0278-0046.

KHALGUI, M.; MOSBAHI, O.; VALENTINI, G. **Embedded Computing Systems: Applications, Optimization, and Advanced Design**. 1st. ed. Hershey, PA, USA: IGI Global, 2013. ISBN 9781466639225.

KIM, B.; PARK, K. Probabilistic Delay Model of Dynamic Message Frame in Flexray Protocol. **IEEE Transactions on Consumer Electronics**, v. 55, n. 1, p. 77–82, fev. 2009. ISSN 0098-3063.

LANGE, R.; OLIVEIRA, R. S. de. Hybrid FlexRay/CAN Automotive Networks. **Embedded Computing Systems: Applications, Optimization, and Advanced Design**, IGI Global, p. 323–342, 2013.

LANGE, R.; OLIVEIRA, R. S. de. Probabilistic Scheduling of the FlexRay Dynamic Segment. In: **12th IEEE International Conference on Industrial Informatics**. Porto Alegre: [s.n.], 2014. p. 201–206. ISBN 978-1-4799-4905-2.

LANGE, R.; VASQUES, F.; OLIVEIRA, R. S. de; PORTUGAL, P. A Scheme for Slot Allocation of the FlexRay Static Segment Based on Response Time Analysis. **Computer Communications**, Elsevier, v. 63, p. 65–76, 2015.

LANGE, R.; VASQUES, F.; PORTUGAL, P.; OLIVEIRA, R. S. de. Guaranteeing Real-Time Message Deadlines in the FlexRay Static Segment Using a On-Lne Scheduling Approach. In: **Proceedings of the 9th IEEE International Workshop on Factory Communication Systems**. Lemgo/Detmold, Germany: IEEE, 2012. p. 301–310. ISBN 978-1-4673-0692-8.

LEEN, G.; HEFFERNAN, D. TTCAN: A New Time-Triggered Controller Area Network. **Microprocessors and Microsystems**, v. 26, n. 2, p. 77–94, 2002.

LEHOCZKY, J. P. Fixed Priority Scheduling of Periodic Task Sets With Arbitrary Deadlines. In: **Proceedings of the 11th Real-Time Systems Symposium**. [S.l.: s.n.], 1990. p. 201–209.

LIU, C. L.; LAYLAND, J. W. Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment. **Journal of the ACM**, ACM, v. 20, n. 1, p. 46–61, jan. 1973. ISSN 00045411.

LIU, J. W. S. **Real-time systems**. [S.l.]: Prentice Hall, 2000. ISBN 0130996513.

Lo Bello, L. The Case for Ethernet in Automotive Communications. **Special Issue on the Tenth International Workshop on Real-Time Networks**, v. 8, n. 4, p. 7–15, 2011.

LUKASIEWYCZ, M.; GLASS, M.; TEICH, J.; MILBREDT, P. FlexRay Schedule Optimization of the Static Segment. In: **Proceedings of the 7th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis - CODES+ISSS '09**. Grenoble, France: ACM Press, 2009. p. 363–372. ISBN 9781605586281.

MALCOLM, N. The Timed-Token Protocol for Real-Time Communications. **IEEE Computer**, IEEE, v. 27, n. 1, p. 35–41, jan. 1994. ISSN 0018-9162.

MARINCA, D.; MINET, P.; GEORGE, L. Analysis of Deadline Assignment Methods in Distributed Real-Time Systems. **Computer Communications**, v. 27, n. 15, p. 1412–1423, set. 2004. ISSN 01403664.

NAVET, N.; SIMONOT-LION, F. (Ed.). **Automotive Embedded Systems Handbook**. 1. ed. [S.l.]: CRC Press, 2008. ISBN 978-0-8493-8026-6.

OUEDRAOGO, L.; KUMAR, R. Computation of the Precise Worst-Case Response Time of FlexRay Dynamic Messages. **IEEE Transactions on Automation Science and Engineering**, v. 11, n. 2, p. 537–548, abr. 2014. ISSN 1545-5955.

PARK, I.; SUNWOO, M. FlexRay Network Parameter Optimization Method for Automotive Applications. **IEEE Transactions on Industrial Electronics**, v. 58, n. 4, p. 1449–1459, abr. 2011. ISSN 0278-0046.

POP, T.; POP, P.; ELES, P.; ANDREI, A. Timing Analysis of the FlexRay Communication Protocol. **Real-Time Systems**, IEEE, v. 39, n. 1-3, p. 205–235, 2008.

SCHEDL, A. Goals and Architecture of FlexRay at BMW. In: **Vector FlexRay Symposium**. Germany: [s.n.], 2007.

SCHMIDT, E. G.; ALKAN, M.; SCHMIDT, K.; YURUKLU, E.; KARAKAYA, U. Performance Evaluation of FlexRay/CAN Networks Interconnected by a Gateway. In: **International Symposium on Industrial Embedded System (SIES)**. [S.l.]: IEEE, 2010. p. 209–212. ISBN 978-1-4244-5839-4.

SCHMIDT, K.; SCHMIDT, E. Message Scheduling for the FlexRay Protocol: The Static Segment. **IEEE Transactions on Vehicular Technology**, v. 58, n. 5, p. 2170–2179, 2009. ISSN 0018-9545.

SCHMIDT, K.; SCHMIDT, E. Optimal Message Scheduling for the Static Segment of FlexRay. In: **Proceedings of the 72nd Vehicular Technology Conference - Fall**. Ottawa, Canada: IEEE, 2010. p. 1–5. ISBN 978-1-4244-3573-9.

SCHMIDT, K.; SCHMIDT, E. G. Schedulability Analysis and Message Schedule Computation for the Dynamic Segment of FlexRay. In: IEEE (Ed.). **72nd IEEE Vehicular Technology Conference - Fall**. Ottawa, ON: IEEE, 2010. p. 1–5. ISBN 978-1-4244-3573-9.

SPRINGER. **Springer Science+Business Media**. 2015. Disponível em: <http://www.springer.com/>.

STEINBACH, T.; KORF, F.; SCHMIDT, T. C. Comparing time-triggered Ethernet with FlexRay: An evaluation of competing approaches to real-time for in-vehicle networks. In: **2010 IEEE International Workshop on Factory Communication Systems Proceedings**. Nancy, France: IEEE, 2010. p. 199–202. ISBN 978-1-4244-5460-0.

TANASA, B.; BORDOLOI, U. D.; ELES, P.; PENG, Z. Probabilistic Timing Analysis for the Dynamic Segment of FlexRay. In: **2013 25th Euromicro Conference on Real-Time Systems**. [S.l.]: IEEE, 2013. p. 135–144. ISBN 978-0-7695-5054-1.

TANASA, B.; BORDOLOI, U. D.; KOSUCH, S.; ELES, P.; PENG, Z. Schedulability Analysis for the Dynamic Segment of FlexRay: A Generalization to Slot Multiplexing. In: **18th IEEE Real Time and Embedded Technology and Applications Symposium**. [S.l.]: IEEE, 2012. p. 185–194. ISBN 978-1-4673-0883-0.

THIELE, L.; CHAKRABORTY, S.; NAEDELE, M. Real-Time Calculus for Scheduling Hard Real-Time Systems. In: **Proceedings of the 2000 IEEE International Symposium on Circuits and Systems**. [S.l.: s.n.], 2000. v. 4, p. 101–104. ISBN 0-7803-5482-6. ISSN 02714310.

TINDELL, K.; BURNS, A.; WELLINGS, A. J. Calculating Controller Area Network (CAN) Message Response Times. **Control Engineering Practice**, v. 3, n. 8, p. 1163–1169, 1995.

TOVAR, E.; VASQUES, F. Guaranteeing Real-Time Message Deadlines in Profibus Networks. In: **Proceedings of the 10th Euromicro Workshop on Real-Time Systems**. Berlin, Germany: [s.n.], 1998. p. 79–86.

TOVAR, E.; VASQUES, F. Cycle Time Properties of the PROFIBUS Timed Token Protocol . **Computer Communications**, Elsevier Press, v. 22, p. 1206–1216, 1999.

TTTech Computertechnik AG. **TTEthernet Specification**. 2015. Disponível em: <http://www.ttagroup.org>.

VAZ, R. M. **Automatizando o Agendamento de Mensagens no Protocolo FlexRay**. Trabalho de Conclusão de Curso. Faculdade de Tecnologia (FATEC) Santo André, 2011.

VECTOR. **Vector Informatik GmbH**. 2012. Disponível em: <http://www.vector.com/>.

ZENG, H.; Di Natale, M.; GHOSAL, A.; SANGIOVANNI-VINCENTELLI, A. Schedule Optimization of Time-Triggered Systems Communicating Over the FlexRay Static Segment. **IEEE Transactions on Industrial Informatics**, v. 7, n. 1, p. 1–17, fev. 2011. ISSN 1551-3203.

ZENG, H.; GHOSAL, A.; Di Natale, M. **Timing Analysis and Optimization of Flexray Dynamic Segment**. [S.l.]: IEEE, 2010. 1932–1939 p.

ZURAWSKI, R. **The Industrial Communication Technology Handbook**. [S.l.]: CRC Press, 2005. ISBN 0849330777.