Jorge Luis González Reaño

Ajuste de tráfego intrachip obtido por simulação no nível de transação a modelos de séries autossimilares

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Engenharia Elétrica.

Área de concentração: Microeletrônica.

Orientador: Prof. Dr. Wang Jiang Chau

São Paulo
2013

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com anuência de seu orientador.

São Paulo, 23 de outubro de 2013.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

González Reaño, Jorge Luis
Ajuste de tráfego intrachip obtido por simulação no nível de transação a modelos de séries autossimilares / J.L. González Reaño. -- ed. rev. -- São Paulo, 2013.
80 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Sistemas Eletrônicos.

1. Modelos em Séries Temporais 2. Dependência de Longa Duração 3. Redes e Comunicação de Dados 4. Tráfego 5. Sistemas Embutidos 6. Microeletrônica I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Sistemas Eletrônicos II. t.

Aos meus pais, Jorge e Gloria, porque são minha inspiração e força diária.

Agradecimentos

A Deus porque é a minha meta e quem me impulsiona.

Ao Prof. Wang pela formação como pesquisador sendo sempre um exemplo humano e acadêmico.

Ao Prof. Roberto Amazonas pela extrema ajuda neste trabalho.

À minha família e à Ruth, pelo amor e presença, em momentos de alegria e dificuldade.

Ao Gustavo, à Mary e aos meus amigos da Bijari: Juliana, Natasha, Fábio e Ricardo, pela amizade e ajuda que me brindaram durante este período.

Aos meus colegas do Grupo de Projeto de Sistemas Eletrônicos Integrados e Software Aplicado, especialmente a: Mario, Carlos, Joel e Cristopher; e ao Prof. Marius Strum.

Aos meu amigos Mary, Kalun e Ary que me ajudaram a chegar a Brasil.

Ao Chrisnael e aos meus amigos que estão no Peru pela amizade sem fronteiras.

Ao Cnpq pelo apoio financeiro fornecido durante os dois anos de mestrado, e à CAPES, pelo financiamento do portal de periódicos eletrônicos, que permite o acesso ao conhecimento científico.

"Essencialmente, todos os modelos são errados. Mas alguns são úteis."

George E. P. Box

Resumo

Este trabalho visa dar uma contribuição para o aumento de eficiência no fluxo de projeto de sistemas integrados, especificamente na avaliação de desempenho da comunicação entre os seus blocos componentes. É proposto o uso de modelagem e simulação de hardware em alto nível, no nível de transações, denominado TLM, para aproveitar a redução de esforço e tempo que se pode oferecer ao projeto de sistemas integrados, diferentemente de enfoques convencionais em níveis mais baixos de descrição, como o nível de registradores (RTL). É proposta uma forma de análise do tráfego intrachip produzido na comunicação de elementos do sistema, visando-se o uso dos resultados obtidos para descrição de geradores de tráfego.

A principal contribuição deste trabalho é a proposta da análise de séries de tráfego obtido durante simulação de plataformas de hardware descritas no nível TLM usando-se métodos estatísticos conhecidos da área de estudo de séries temporais. A análise permite ao projetista ter maior compreensão da natureza estatística do tráfego intrachip, denominada dependência de curta ou longa duração (SRD e LRD), para o posterior ajuste de modelos usados na geração de séries sintéticas que representem tal natureza.

Os resultados da análise mostraram que o tráfego obtido por simulação TLM tem natureza similar em relação ao da do tráfego obtido por simulação num nível mais baixo de abstração, do tipo de precisão por ciclos, indicando que o tráfego TLM pode ser usado para a representação do tráfego intrachip. Outra contribuição deste trabalho é a proposta de ajuste de modelos paramétricos autossimilares usando-se a decomposição da série de tráfego original, tendo sido feita uma comparação dos resultados desta com o ajuste convencional feito a modelos sem decomposição. Estas contribuições foram agrupadas dentro de uma metodologia detalhada, apresentada neste documento, para a qual experimentos foram realizados. Os resultados a partir das séries sintéticas autossimilares geradas pelos modelos estimados, apresentaram semelhança nos indicadores de SRD e LRD em relação às séries originais TLM, mostrando ser favorável o uso futuro destas séries sintéticas na implementação de geradores de tráfego.

Palavras-chave: Tráfego intrachip, Modelagem no Nível de Transações, Dependência de longa duração, modelos autossimilares, ajuste por decomposição.

Abstract

It is objective of this work to make a contribution to improve the efficency of the integrated systems design flow, specifically on the evaluation of communication performance between component blocks. The use of high level hardware modeling and simulation, at the transaction level, knownas TLM, is proposed, in order to take advantage of the reduction of effort and time for the integrated system design; that in contrast to the traditional approaches, which use lower hardware description level, such as register transfer level (RTL). A methodolgy to evaluate the intra-chip traffic produced by the communication between system elements is proposed.

The main contribution of this work is the analysis of traffic time series obtained by simulation of hardware platforms modeled in TLM, using well-known statistical methods for time series analysis. The analysis allows the system developer to understand the statistical nature of the intra-chip traffic, also known as short and long range dependence (SRD and LRD), for later adjustment and accurate representation of the traffic nature in synthetic series.

The analysis results have shown that traffic traces obtained by TLM simulation has similar statistical nature as the traffic traces obtained at lower abstraction level, as cycle accurate type, which indicates that TLM traffic could be used to represent intrachip traffic. Another contribution of this work is a fitting procedure to auto-similar parametric models thought the decomposition of the original traffic, and its comparison to the results of the conventional fitting, when applied to models that are not decomposed. These contributions were grouped andincluded in the detailed methodology presented in this document, being a series of experiments carried out. The results related to self-similar synthetic series, obtained from the fitted models, havae shown similiraty to the SRD and LRD indicators of the original TLM series, what favors the use of synthetic series future for the implementation of traffic generators.

Keywords: Intra-chip traffic, Transaction Level Modeling, Long Range Dependence, Self-similar models, decomposition fitting.

-

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos e Notação

ARFIMA	Auto Regressive Fractional Moving Average
ARMA	Auto Regressive Moving Average
ASIC	Application Specific Integrated Circuit
CA	Cycle Accurate
CI	Circuito Integrado
DWPT	Discrete Wavelet Packet Transform
DWT	Discrete Wavelet Transform
EP	Elemento Processante
ESL	Electronic System Level
FAC	Função de Autocorrelação
fBM	Fractional Brownian Motion
fGN	Fractional Gaussian Noise
GPP	General Purpose Processor
GSEIS	Grupo de Projeto de Sistemas Eletrônicos Integrados e Software Aplicado
H	Hurst
HDL	Hardware Description Language
HW/SW	Hardware/Software
IP Core	Intellectual Property Core
ISS	Instruction Set Simulator
LME	Laboratôrio de Microeletrônica

LRD	Long Range Dependence
MA	Moving Average
MFT	Multiresolution Fourier Transform
MIMD	Multiple Instructions – Multiple Data
MIPS	Microprocessor without Interlocked Pipeline Stages
MJPEG	Motion JPEG
MPSoC	Multi Processor System on Chip
NoC	Network on Chip
R/S	Range over Standard Deviation
RC	Reconfigurable Core
RTL	Register Transfer Level
SoC	System on Chip
SRD	Short Range Dependence
TLM	Transaction Level Modeling
TLMDT	TLM with Distributed Time
TTM	Time to Market
USP	Universidade de São Paulo
VCD	Value Change Dump
VCI	Virtual Component Interface
VHDL	VHSIC Hardware Description Language

$2D, 3D$	indica dimensão espacial
x_t	série de tráfego intrachip
H	parâmetro de Hurst
$1/f^{\alpha}$	singularidade na frequência do periodograma com tendência a infinito
T'_t, S'_t, R'_t	Elementos da decomposição Tendência, Periódico e Restante

Sumário

Capítulo 1

Introdução Geral

Apresentamos nesta seção introdutória o contexto e as motivações para a modelagem de um gerador de tráfego sintético para redes intrachip e também detalhamos os objetivos do trabalho realizado.

1.1 Contexto

Com o desenvolvimento constante de novas técnicas de processos em microeletrônica, a tecnologia de fabricação de circuitos integrados (CIs) tem progredido de forma ininterrupta no objetivo de obter maior integração de dispositivos dentro de uma mesma área. Com a redução do tamanho dos dispositivos CMOS, atualmente as dimensões mínimas estão abaixo de 22nm, o que tem causado uma mudança na potencialidade de integração dos chips na indústria, passando de milhões para bilhões, o uso de transistores em um único CI (DAMARAJU et al., 2012).

A alta integração dos CIs tem levado a chips com uma estrutura interna composta por componentes heterogêneos como múltiplos módulos de processamento programáveis[1], hardware de propósito específico[2], periféricos, memórias embarcadas e uma arquitetura de comunicação para interconectar os componentes acima. Em conjunto, operam como um sistema, conhecido pelo o termo MPSoC (do inglês *Multi Processor System on Chip*). Segundo (PASRICHA; DUTT, 2008), os MPSoCs consistem de estruturas de chip complexas, dedicadas a atender às necessidades computacionais específicas de uma aplicação e aplicam-se para eles os conceitos conhecidos pela computação paralela referentes à arquitetura de múltiplas instruções-múltiplos-dados (MIMD) da taxonomia de Flynn (FLYNN, 1972 apud PASRICHA; DUTT, 2008).

Se a capacidade de processamento de hardware tem crescido enormemente, por outro lado, é cada vez maior a complexidade no desenvolvimento do software embarcado para que seja executado com eficiência, e em tempo real, nos sistemas MPSoC. O software passa a ser um aspecto influente no tempo para o produto chegar ao mercado (em inglês, *time to market*: TTM); em média, quarenta por cento desse tempo é utilizado para preparar o código da aplicação, outros quarenta por cento do tempo é destinado à preparação do hardware e vinte por cento restante

[1]Podem ser processadores multi núcleo de propósito geral (GPP) ou núcleos reconfiguráveis (RC).

[2]Podem ser processadores com conjunto de instruções para aplicações especificas (ASIP) ou circuitos integrados de aplicação específica (ASIC).

para conseguir executar o software no hardware (COPPOLA et al., 2008).

Ante a problemática do desenvolvimento dos MPSoCs e por causa da extensa funcionalidade a ser integrada entre o hardware e o software, os projetistas adotaram a metodologia do nível de sistema eletrônico (ESL) (BAILEY; MARTIN; PIZIALI, 2007). Esta metodologia minimiza a complexidade e esforço do projeto, apesar de permitir experimentar novas formas de se alcançar a funcionalidade desejada, pela combinação dos componentes das arquiteturas (hardware e software). Em (GARZON, 2008), três estratégias, representadas na Figura 1.1, são apresentadas:

1. Particionamento do projeto nos seus domínios de representação. Esses domínios correspondem às primeiras três etapas da metodologia ESL. Levando aos seguintes modelos:

   (a) O modelo funcional, que está no nível mais abstrato criado a partir dos algoritmos otimizados da aplicação alvo do projeto definidos numa linguagem de alto nível como C ou C++.

   (b) O modelo de arquitetura, resultante da exploração das arquiteturas potenciais e de um processo de seleção fundamentado na análise das melhores possibilidades de execução dos algoritmos; estas são baseadas num particionamento HW/SW e o mapeamento das tarefas nos módulos de hardware.

   (c) O modelo de comunicação entre os processos, definido por uma arquitetura de comunicação que cumpra com os requisitos e restrições do modelo de arquitetura.

2. Uso de modelos de abstração maior que o do nível de transferência entre registradores (RTL) para descrição do hardware, como, por exemplo, o modelo de nível de transações (TLM) que pode ser usado na modelagem da arquitetura de comunicação. A descrição TLM modela a comunicação entre módulos com funções de chamada que representam transações; a comunicação pode considerar variáveis de tempo como o tempo do inicio, tempo do final da transação e carga de dados (*payload*). A Figura 1.1 detalha os componentes e elementos existentes em cada nível de abstração do projeto em questão, o qual tem uma relação inversa com o grau de complexidade do projeto permitindo descrições de sistema concisas e de rápida simulação em comparação as simulações de projetos em RTL.

3. Reaproveitamento de blocos modulares dentro do sistema; baseado na existência de blocos de IP (do inglês, *Intellectual Property Cores*) pré-projetados e pré-verificados por terceiros[3]. Os blocos de IP contêm a descrição de um processador, um bloco de memória ou algum circuito que atenda aos requisitos de uma função de processamento específica (WILTON; SALEH, 2001). Podem ser *hardcores* quando correspondem a uma biblioteca de um projeto anterior e não é permitida a sua alteração, ou podem ser *softcores* que são bibliotecas modificáveis pelo menos nos parâmetros de operação. Os blocos são padronizados para facilitar o reaproveitamento e tendem a ser agrupados sob a abordagem de projeto baseado em plataformas (em inglês, *platform based design*). Define-se plataforma

[3]Alguns exemplos de vendedores de IP comerciais são Xillinx, Altera, ARM e Lattice.

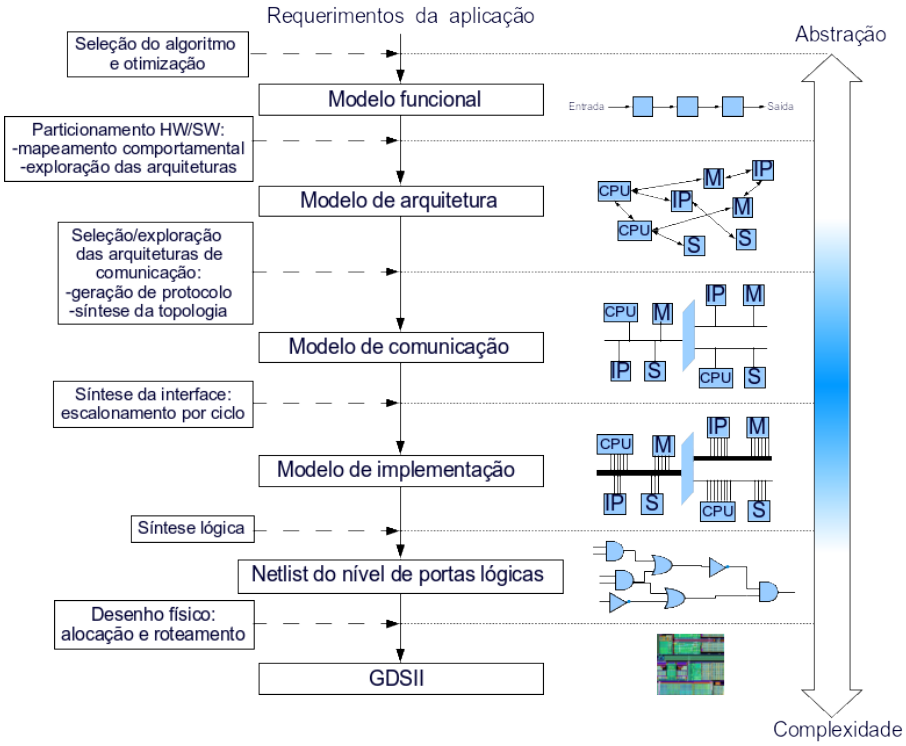


Figura 1.1: Relação entre abstração e complexidade no fluxo ideal da metodologia ESL - Anexado de (PASRICHA; DUTT, 2008)[p.5]

(KEUTZER et al., 2000), como um arranjo pré-configurado de um determinado número de blocos IP, do tipo *hard* ou *soft*, de algum domínio específico de aplicação ou para uma aplicação final.

Após a representação do projeto por seus modelos mais abstratos, este necessita passar por uma série de refinamentos, seguindo para o nível de sinais e pinos, que corresponde a descrições e modelos no nível comportamental RTL. Posteriormente, o modelo RTL é usado nas ferramentas de síntese para traduzir o projeto ao nível de portas lógicas criando-se um netlist. Completada a verificação, posicionamento dos módulos e o roteamento no nível físico podem ser realizados, e o arquivo GDSII [4](do inglês, *Graphic Database System II*) pode ser gerado, para, daí, seguir para fabricação.

Dentro da metodologia ESL é de vital importância a estratégia adotada para o modelo de comunicação;dito depende a consecução de associação e interoperabilidade entre os blocos IP ou as plataformas que integram o MPSoC. Trata-se de um dos aspectos mais importantes que irá conduzir o projeto de circuitos integrados nos próximos anos (MARCULESCU, 2010). O modelo de comunicação é selecionado a critério dos projetistas para que se cumpra as expectativas do projeto resumidas em (NICOPOULOS; NARAYANAN; DAS, 2009): (1) desempenho, (2) consumo de área em silício, (3) eficiência de potência/energia, (4) confiabilidade, e (5) variabilidade. Objetiva-se, assim, que forneça benefícios como a flexibilidade para se programar, de acordo com as características de cada módulo. Através do sistema de comunicação, o impacto

[4]Formato padrão de layout de CI's, proprietario da Cadence Design Systems.

das interconexões frente às expectativas de projeto pode ser estimado, avaliando-se o desempenho da rede, a relação entre energia e temperatura de funcionamento, a confiabilidade, o custo e o TTM.

Existe um conjunto amplo de estruturas de interconexão, mas dois tipos são tipicamente realizados nos chips: os barramentos, incluindo-se seus derivados, como os *crossbars*, e as redes intrachip ou NoCs (do inglês, *Network on Chip*), ambos espelhados nos princípios e fundamentos aplicados aos barramentos e as redes de computadores e às Internet. Apesar da similaridade com estas últimas e da existência de características herdadas delas, as variáveis consideradas para o dimensionamento da rede são dissimilares nos ambientes intrachip e extrachip, dada a grande escala de valores de parâmetros que as separam. As NoCs são originadas pela combinação de metodologias em projeto MPSoC, de padrões usados dentro de computação paralela e distribuída e das interconexões das redes de computadores, objetivando-se cobrir algumas necessidades técnicas que um modelo de comunicação, baseado só em barramentos, não pode atingir dentro de um custo razoável.

1.2 Comparação entre barramentos e NoCs

O sistema de barramentos é composto por elementos usados em diversos dispositivos SoC como canais, *buffers*, *switches* e unidades de controle que facilitam o transporte de dados entre blocos IP (PASRICHA; DUTT, 2008). Um canal é composto por fios (*wires*) e corresponde ao meio de propagação de sinais de dados, controle, endereço, sinalização e arbitragem, sendo a largura do barramento determinada pela quantidade de fios. Podem-se encontrar dois tipos de componentes, os ativos chamados mestres, responsáveis de iniciar a comunicação, e os passivos ou escravos que respondem às demandas de comunicação.

Um barramento pode ser do tipo compartilhado (Figura 1.2.a), que funciona como um decodificador de endereços (pode ser centralizado ou distribuído) e que controla um multiplexador encarregado da seleção do mestre, caso o escravo não possa responder, a transferência sofre atraso. Este tipo de arbitragem cria atrasos porque numa transação, os sinais de endereço e leitura/escrita do mestre são enviados a todos os escravos, gerando contenção no barramento, além de maior consumo de energia. Para reduzir os atrasos e a contenção, o barramento segmentado (Figura 1.2.b) foi concebido, formado por barramentos interconectados por pontes; estes poderiam ser FIFOS, caso os protocolos fossem iguais. Neste esquema, tanto a arbitragem como a alocação ótima de componentes são complexas. Frente às limitações anteriores, surgem os *crossbars* multiponto (Figura 1.2.c) que resultam da conexão direta de todos os componentes, porém os custos de implementação elevam-se e tornam-se impraticáveis para o projeto . Pode-se ter também uma estrutura de *crossbars multistage* (Figura 1.2.d), onde todos os componentes estão conectados, mas o acesso é controlado por *switches*. Se, por um lado, os *crossbars* resolvem os problemas relacionados à largura de banda, as limitações nos barramentos ganham relevância ante as dificuldades de escalabilidade, pela presença de um maior número de processadores e núcleos dentro dos MPSoCs. Ficam evidenciados gargalos, com piores atrasos por arbitragem, maior consumo de energia (em proporção direita ao tamanho do *footprint* na área) e mais dificuldades de integração de blocos com frequências variadas (PEH; JERGER, 2009). Além dos

problemas anteriores mencionados,com a integração abaixo de *22 nm*, aumenta-se o risco de erro gerado pelo ruído elétrico, interferências eletromagnéticas e os funcionamentos defeituosos por *crosstalk* nas linhas de interconexão (FLYNN; HUNG; RUDD, 1999).

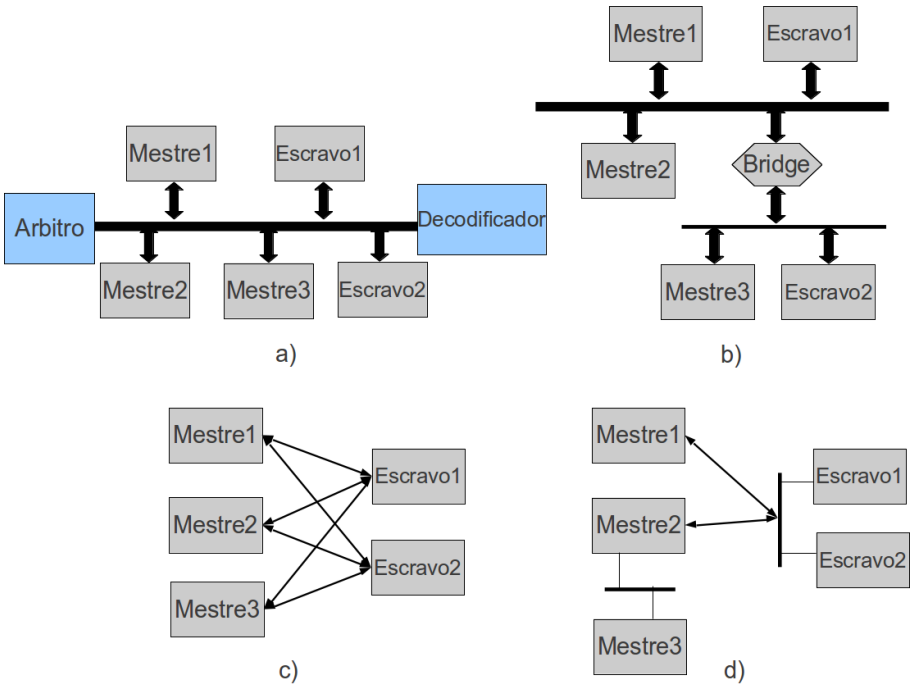


Figura 1.2: Classificação dos barramentos: (a) Barramento compartilhado com arbitragem centralizada. (b) Barramento segmentado. (c) Crossbar multiponto. (d) Crossbar multistage

As redes intrachip surgem pela necessidade de se alcançar níveis adequados de escalabilidade e de se usar estruturas ordenadas e eficientes de interconexão nos MPSoCs. Diferente de um barramento que adota transferências por circuito, as NoCs transmitem mensagens em pacotes de dados em *pipeline*, direcionadas a partir de um nó origem[5] a um nó destino. Outra diferença importante é que as NoCs são sistemas de comunicação distribuídos, com roteadores e enlaces ao longo de todo o chip, multiplexando os distintos fluxos de dados num mesmo enlace; trabalham com roteamento baseado em algoritmos, diferente de um barramento que enfoca a decodificação para ocupação dos canais, resultando em mais eficiência na utilização das interconexões e reduzindo a área ocupada, o leva à diminuição do consumo de energia e da quantidade de atrasos.

Uma rede NoC geralmente realiza a comunicação dos nós do MPSoC através de duas ligações que formam um enlace ou canal bidirecional, via um componente comum chamado de roteador. A Figura 1.3 mostra os componentes dentro de uma rede dispostos em topologia grelha 3x3; a estrutura interna de um roteador é composta por *buffers* na entrada, um elemento de controle ou árbitro, e um *crossbar* como matriz de chaveamento. Observa-se, neste exemplo de rede, que o nó processante está sempre ligado a um roteador responsável de encaminhar as mensagens que o primeiro gera ou recebe. Neste caso, é um bloco IP de um elemento de processamento (EP) com cache hierárquica L1 e L2.

[5]Um nó é qualquer componente ou conjunto destes que está ligado a rede.

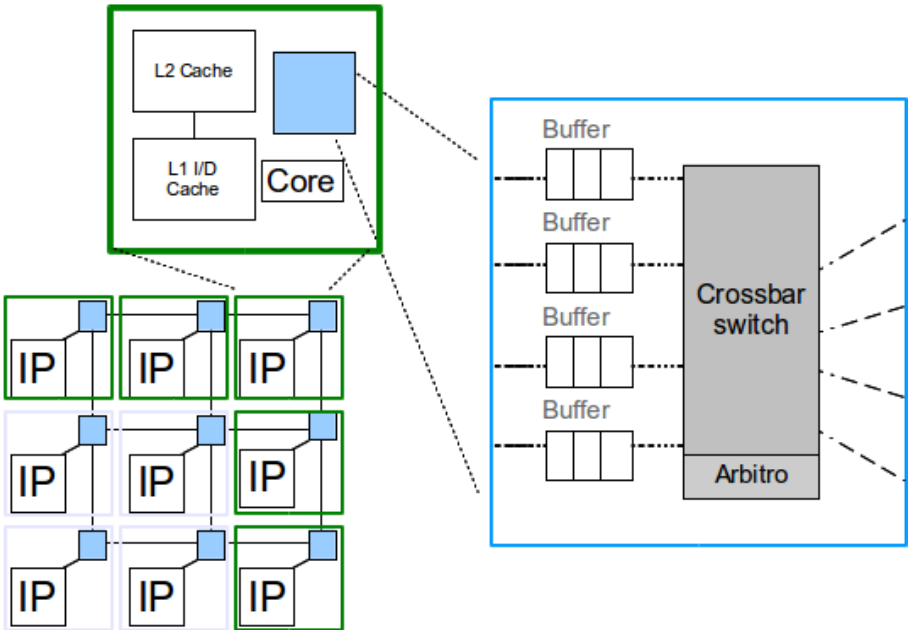


Figura 1.3: Roteador de uma rede Intrachip de 3x3 grelha.

1.3 Motivação

Na simulação lógica convencional de MPSoCs, os modelos RTL são aqueles que primordialmente definem o hardware; este é representado, numa linguagem de descrição de hardware (HDL), como VHDL e Verilog, e adequados para síntese. Para a execução do software embarcado, é necessária a descrição completa do(s) processador(es); dada a complexidade destes modelos, na simulação do sistema, é somente factível uma breve observação do comportamento total do sistema devido aos elevados tempos de simulação envolvidos. Desta forma, o uso da descrição RTL faz com que o desenvolvimento do sistema hardware-software acabe por tomar uma grande parte do tempo total do projeto, apesar de a simulação fornecer alta fidelidade na observação do comportamento do sistema em relação à implementação real.

Uma das primeiras tentativas para reduzir o impacto da simulação no tempo de projeto foi a criação de plataformas que permitissem a co-verificação de *HW/SW*. Em uma plataforma de co-verificação, o hardware continua sendo modelado em RTL, entretanto, modelos de processadores apresentam uma descrição menos detalhada; exemplos disto são os simuladores de conjuntos de instruções, ISSs (do inglês, *Instruction Set Simulator*), que possuem alta precisão nas instruções por ciclo de execução. Numa plataforma deste tipo, o hardware geral está conectado a um simulador lógico e um depurador (*debugger*) simbólico que liga o software da aplicação com o processador ISS para simulação.

Em paralelo aos ISSs, o modelo de precisão de ciclos , CA, (do inglês *cycle-accurate*), foi adotado, para o qual são descritas algumas das partes de hardware por modelos definidos em linguagens como SystemC[6] (OSCI, 2011a). Esta descrição é sensível aos eventos, às variações de valores de variáveis ou sinais, etc. de cada ciclo de relógio, mas é menos preciso do que o modelo

[6]SystemC é uma biblioteca de classes C++ que fornece construções orientadas à modelagem de hardware que uma linguagem tradicional não possui.

RTL[7]. O modelo CA garante o protocolo de comunicação dos sinais do modelo descrito (em ciclos de relógio), permitindo a verificação, mas não corresponde a uma descrição que permita a síntese do projeto em andamento. Dada a complexidade do tratamento em ciclo de relógio no modelo CA, a velocidade de simulação do sistema não é muito diferente dos tempos idealizados nas descrições RTL em VHDL ou Verilog.

Com o crescimento contínuo na complexidade do projeto MPSoC, a metodologia ESL surgiu para aumentar a produtividade, resultando em mais blocos projetados em um menor intervalo de tempo. Para tornar o desenvolvimento de projetos MPSoC mais gerenciáveis, a metodologia ESL realiza-os por meio de refinamentos sucessivos, em etapas que correspondem a gerar diferentes modelos de descrição de hardware e software, em diferentes níveis de abstração. A técnica de abstração consiste na remoção de detalhes nos modelos, frente aos requisitos de determinada etapa (GRöTKER et al., 2002). Na Figura 1.1 é observado nas quatro primeiras etapas: os sistemas são modelados a partir do nível mais alto de abstração por algoritmos; na etapa seguinte, já incorporam a noção de hardware ou software no modelo de arquitetura; depois consideram a estrutura de comunicação necessário ao modelo anterior; e finalmente, chega-se a um modelo de implementação em RTL que representa com fidelidade detalhes como a especificação de pinos e a operação por ciclos de relógio.

Originalmente, a modelagem no nível de transações (TLM) foi concebida para conceituar a estrutura de comunicação incorporada sobre o padrão IEEE 1666 do SystemC. Atualmente os modelos TLM podem ser usados nas descrições de hardware dos elementos processantes, EPs, do sistema (SCHIRNER; GERSTLAUER; DOMER, 2010; BOMBIERI; FUMMI; PRAVADELLI, 2011) através das características herdadas da linguagem SystemC [8]. A metodologia ESL considera a simulação destes modelos de hardware mantendo o equilíbrio entre velocidade, precisão e esforço mínimo; permitindo um rápido e confiável desenvolvimento do sistema alvo.

Em TLM, cada componente do sistema é modelado por um conjunto finito de estados e uma série de comportamentos concorrentes, em uma estrutura chamada módulo (GHENASSIA, 2005). Cada módulo possui variáveis que representam os estados internos e os comportamentos são modelados por um conjunto de processos concorrentes ou *threads* executados em paralelo. Os módulos comunicam-se entre si por estruturas chamadas canais, que dependendo do projetista, podem representar barramentos, roteadores simples assim como outras estruturas. Os módulos e canais estão ligados uns aos outros por meio de portos de comunicação que facilitam a troca de informação, realizada em grupos, definidos como transações. As transações são originadas nos componentes mestres, dirigidas aos componentes escravos, e possuem diversos tamanhos em relação à quantidade de dados trocados. Em um sistema definido neste modelo não se dá muita importância ao protocolo usado nas transferências; foco é nos dados transferidos e nas localidades de origem e destino.

A integração entre componentes TLM (módulos, canais e portos) conformam uma plataforma TLM do MPSoC projetado, a qual pode ser simulada, executando-se software embarcado mediante uma compilação nativa ou cruzada (em inglês, *cross-compilation*) sobre a descrição de um processador. Tal processador, inclusive, poderia ser um ISS, contudo deveria consistir num

[7] a rigor, o RTL poderia ser incluído também na classe CA

[8] Modelos RTL, CA, TLM e funcionais podem ser descritos com esta linguagem

módulo com um *wrapper* como interface de comunicação. Numa plataforma TLM um módulo possui as seguintes características:

1. Comportamento bit-true do componente que representa.

2. Precisão ao nível de registradores na interface do componente.

3. Operação em baixo sincronismo do sistema, que permite gerir e predizer o seu comportamento.

A sincronização é realizada por intermédio de mecanismos de especificação de eventos, sinais e interrupções, que efetuam chamadas ao *kernel* de simulação para ativar outros módulos. Se na descrição da plataforma existirem pontos de sincronismo em demasia (ou alto sincronismo), estes fariam com que o modelo se aproximasse de um modelo CA ou RTL, obtendo-se no modelo TLM uma redução pequena no tempo de simulação. Portanto, deve-se buscar o baixo sincronismo na maior quantidade de pontos, porém, de tal forma que o modelo não apresente falhas de funcionamento durante a execução por falta destes.

Tendo a garantia de funcionamento eficiente da NoC ter se tornado um aspecto crítico dentro do fluxo do projeto, o interesse na avaliação de seu desempenho manifesta-se em diversas pesquisas em andamento, tendo sido propostas metodologias baseadas em métricas que ajudam o projetista na seleção de topologias de comunicação, alocação de núcleos de hardware e mapeamento de tarefas de software. Alguns trabalhos existentes têm uma abordagem estatística (BOGDAN; MARCULESCU, 2011), outros de simulação (FLóREZ, 2006), assim como existem aqueles com uma abordagem hibrida (LIU et al., 2011).

A falta de ferramentas disponíveis para avaliação e estudo de NoCs, adicionada ao desejo dos projetistas de se concentrarem em um tópico específico, tem os induzido a fazer modificações em ambientes de simulação ou projetá-los inteiramente. Deste modo, apareceu a plataforma Apolo, proposta por (FLóREZ, 2006) e desenvolvida durante sua pesquisa no Grupo de Projeto de Sistemas Eletrônicos Integrados e Software Aplicado (GSEIS) do Laboratório de Microeletrônica (LME) da Universidade de São Paulo (USP). A plataforma é focada na avaliação de desempenho de NoCs em topologias grelha de duas dimensões, toroide, circular ou circular entrelaçada. No *Apolo*, os geradores de tráfego podem inserir na rede, séries sintéticas geradas por um modelo de tráfego pseudo-aleatório (FLóREZ, 2006) ou séries de tráfego autossimilar (FLóREZ et al., 2010). Entretanto, apresenta, como limitante que nenhuma das séries sintéticas geradas corresponde a um ajuste para um modelo do tráfego gerado pela execução de uma aplicação num elemento de processamento. Um gerador de tráfego eficiente deve representar com fidelidade os rastros reais produzidos pelo bloco que representa dentro do sistema, modelando o comportamento e, sobretudo, a interação com os demais blocos (BAILEY; MARTIN; PIZIALI, 2007).

Os trabalhos seminais com foco no tráfego real intrachip usaram dados gravados a partir de uma simulação do sistema completo; tratando-se de uma quantidade significativa de amostras, aumentava-se o tempo de simulação do sistema especificado, convertendo-se numa tarefa tediosa para o projetista. Torna-se mais efetivo, então, que se analise o tráfego intrachip observado nas

simulações, como séries de dados[9], para que possam ser replicadas através de modelos. Seja para gerar resultados de desempenho na análise de arquitetura de sistemas, seja para construir geradores de tráfego sintético, a modelagem de tráfego converteu-se num importante instrumento orientado a se melhorar a compreensão das séries e a identificar semelhanças em grupos específicos de aplicações. Em consequência é importante que as séries de tráfego modeladas correspondam às aplicações ou código executado semelhantes aos que irão integrar a montagem do sistema final.

Os métodos tradicionais para a geração de séries sintéticas baseiam-se no uso do modelo de Poisson (WIKLUND; SATHE; LIU, 2004) ou de cadeias de Markov (KIASARI et al., 2008; YONGHUI et al., 2008). Posteriormente, a partir da detecção da dependência de longa duração (LRD) no tráfego das NoCs numa aplicação de vídeo (decodificação MPEG2), apresentado em (VARATKAR; MARCULESCU, 2004), séries autossimilares foram usadas e na modelagem de tráfego intrachip em diversas pesquisas (SCHERRER; FRABOULET; RISSET, 2009a; PATIñO; CHAU; STRUM, 2010). Passou-se a estudar a influência da presença da LRD em séries geradas pelos elementos de processamento executando aplicações multimídia. Segundo (PARK; WILLINGER, 2000 apud LIMA, 2008), a autossimilaridade de séries é relacionada à preservação da característica de rajadas *(burstness)*, definido como a alternância entre surtos e os períodos de maior suavidade. As propriedades estatísticas do tráfego original, como a LRD e em consequência o *burstness*, não são mantidas ao longo de várias escalas de agregação temporal usando o modelo de Poisson. Os modelos de tráfego tradicionais, como o modelo de Markov, tendo-se em consideração que o modelo de Poisson é uma cadeia de Markov (LIMA, 2008, p.2) particular, e os modelo regressivos, como é o modelo autorregressivo de media móvel (ARMA), só podem capturar a dependência de curta duração do tráfego (SRD). Como modelos que podem capturar a LRD, usam-se o modelo autorregressivo fracionário de médias móveis (ARFIMA) ou o modelo de movimento fracionário Browniano (fBM).

Das pesquisas que estudaram o LRD no tráfego intrachip, o trabalho de (SOTERIOU; WANG; PEH, 2006) apresenta um gerador de tráfego usado na plataforma *Trident*, capturando-se os tráfegos de diversas aplicações para inserí-los em modelos de blocos no nível RTL. As características do tráfego foram avaliadas, obtendo-se o parâmetro de Hurst (H) que é importante para a modelagem de tráfego LRD a partir dos métodos variance-plot e análise R/S [10]. A geração de tráfego com LRD é realizada por intermédio de um método baseado na criação de processos autossimilares conhecidos como ruído gaussiano fracionário (fGN). O trabalho (PATIñO; CHAU; STRUM, 2010) teve como objetivo observar as mudanças no parâmetro de Hurst em relação ao mapeamento de tarefas, portanto não trata da modelagem do tráfego da NoC. Em (FLóREZ et al., 2010), os autores estudam o impacto de LRD nas redes intrachip, entretanto não indicam como foi gerado o tráfego sintético autossimilar usado para estimular a rede que avaliaram. Em (SCHERRER; FRABOULET; RISSET, 2009a), é apresentada uma metodologia prática e detalhada para a captura de tráfego intrachip e a modelagem com métodos conhecidos pelo estudo de teletráfego, como é a estimativa do Hurst pelo método wavelet; também é feita a geração de séries de dados sintéticas por modelos com SRD ou LRD (ARMA e ARFIMA

[9]É a série temporal que representa o tráfego de um enlace da rede, expressado numa contagem de bytes por unidade de tempo (bytes/ciclo).

[10]Vide a Seção 2.2.4 sobre Métodos de estimação do parâmetro H

respectivamente). Em nenhum dos trabalhos anteriormente mencionados, a modelagem TLM foi considerada como solução às limitações dos modelos RTL e CA discutidas anteriormente, no que se refere aos tempos de simulação e modelagem do sistema.

Usar séries sintéticas, porém próximas a tráfegos reais, evitaria dimensionar equivocadamente a rede intrachip e serviriam para testar topologias em diferentes cenários. Além disso, implementar um gerador de tráfego sintético como um bloco de processamento do sistema permitirá o seu reaproveitamento para fazer testes em outras ferramentas destinadas ao estudo de redes intrachip.

1.4 Objetivos

O presente trabalho irá focar no desenvolvimento de uma metodologia para a análise e estudo de tráfego intrachip, que permita modelar séries de dados realistas, semelhantes às produzidas pela troca de informação entre blocos IP de um MPSoC. A presente pesquisa é fundamentada na metodologia de análise de tráfego proposta por (SCHERRER; FRABOULET; RISSET, 2009a), elevando-se a descrição do sistema para a um nível superior de abstração à descrição da plataforma de simulação, para a captura de tráfego pelo uso de um modelo TLM.

Este trabalho prevê o atendimento dos seguintes objetivos específicos:

- Testar e validar a redução de tempos de simulação em modelos TLM em relação a modelos CA.
- Analisar as características estatísticas dos rastros (traces) de tráfego simulado de plataformas descritas no nível CA e TLM, usando-se métodos adequados para o estudo de séries temporais de dados.
- Avaliar a viabilidade de os modelos TLM serem usados na captura de tráfego intrachip comparando-se as características estatísticas deste tráfego em relação ao tráfego do modelo CA.
- Propor uma metodologia para o ajuste das séries de tráfego originais dos modelos TLM via o modelo autorregressivo de médias móveis (ARMA) e o modelo fracionário autorregressivo integrado de médias móveis (ARFIMA), e avaliar os resultados deste ajuste.

Capítulo 2

Fundamentos teóricos

O propósito deste capítulo é introduzir os alguns conceitos relacionados ao nível de modelagem TLM (GHENASSIA, 2005; BOMBIERI; FUMMI; PRAVADELLI, 2008), assim como apresentar os fundamentos relacionados à análise de séries temporais (LIMA, 2008; AUGUSTO, 2009; BERAN, 1994; SHUMWAY; STOFFER, 2005) referentes ao trabalho de pesquisa desta dissertação.

2.1 Modelagem no nível de transações TLM

Segundo (BOMBIERI; FUMMI; PRAVADELLI, 2008) um evento está relacionado às circunstâncias referentes ao sistema modelado, nas quais ocorrem mudanças nos ciclos do relógio, execuções de instruções, chamadas de funções etc., estabelecendo-se a relação entre a granularidade e o nível de abstração do modelo, como representada na Figura 2.1. Observa-se nela que a modelagem é realizada, para os seus domínios de computação e comunicação, em três níveis em cada eixo, os quais correspondem à granularidade dos eventos e à precisão do tempo da sequência de eventos; o cruzamento entre estes domínios define o nível do abstração do sistema modelado. Em (CAI; GAJSKI, 2003), o autor define que o nível mais alto de abstração, que está mais próximo à origem dos eixos, corresponde ao Modelo Funcional, também chamado de Modelo de Especificação; neste nível, não são incluídos detalhes de implementação, portanto é um nível atemporal. O nível seguinte é chamado de Modelo de Montagem de Componentes e possui representações dos EPs (elementos processantes) com anotações de tempos aproximados e a comunicação continua sendo modelada por meio de canais sem noção de tempo. Diferente do modelo anterior, o Modelo de Barramento com Arbitragem já possui tempos aproximados de comunicação. O modelo do sistema corresponde ao nível de Barramento Funcional quando o Modelo de Barramento com Arbitragem possui precisão e realismo no tempo de comunicação. Caso contrário, corresponderá ao Modelo de Computação com Precisão por Ciclos, especificamente um Modelo de Barramento com Arbitragem com anotações de tempos de computação muito realistas e precisos. O nível mais baixo de abstração é o Modelo de Implementação que possui precisão por ciclos em ambos os domínios de comunicação e computação. Os modelos de Montagem de Componentes, Barramento com Arbitragem, Barramento Funcional e Computação com Precisão por Ciclos podem ser classificados como TLM, considerando-se que neste

nível os modelos podem apresentar anotações tempo relacionados aos ciclos de relógio. O nível de menor abstração é o modelo totalmente CA (*Cycle Accurate*) que inclui a modelagem RTL, o qual corresponde ao Modelo de Implementação, sendo o modelo RTL a representação mais fiel do sistema tanto em eventos como em tempo.

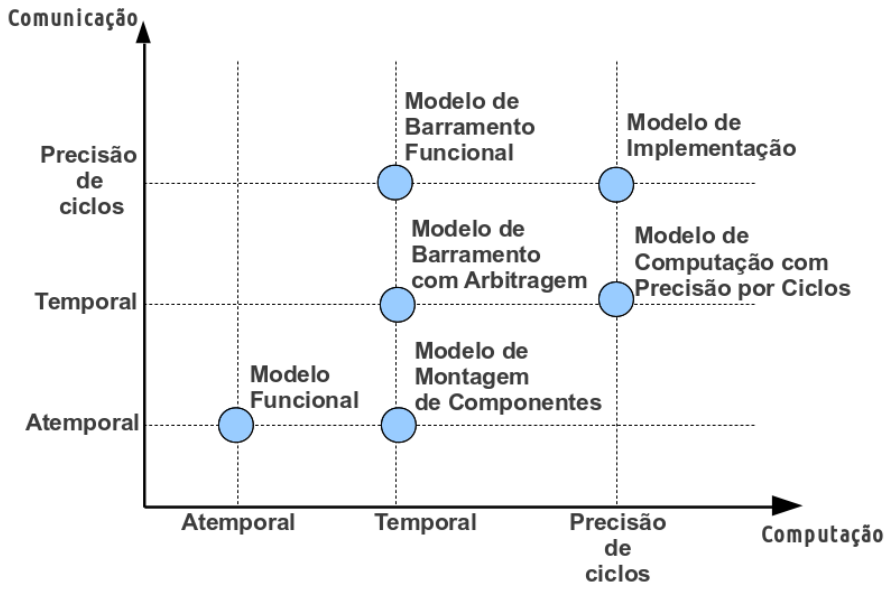


Figura 2.1: Precisão de modelos em relação à granularidade e tempo - Anexado de (CAI; GAJSKI, 2003)

2.1.1 Classificação dos modelos TLM em relação ao tempo

Atemporal TLM-AT

Os modelos completamente atemporais não dependem das informações de tempo dos módulos para produzir respostas, não possuindo anotações de tempos de computação ou de comunicação. Foram concebidos principalmente para a descrição funcional tanto de software como de hardware dentro do fluxo de projeto. Permite o acesso às informações de componentes, como as dos valores em bancos de memória, entretanto não é possível capturar informações relacionadas à arquitetura do sistema no que se refere às regras de arbitragem ou à contenção de recursos. Neste nível, os modelos não possuem relógio, porém, é permitido inserir atrasos funcionais no nível de arquitetura para definir sequências ou ações (por ex. iniciar ou parar eventos) ou dar alguma característica inerente à aplicação (por exemplo, a decodificação de um número de imagens por unidade de tempo) sem afetar o sincronismo entre processos sendo estes modelos chamados de TLM-T ou TLM Temporais. Pode-se classificar ao Modelo de Montagem de Componentes da Figura 2.1 como TLM-A, porque possui anotações de tempo relacionadas as características computacionais mas é atemporal na comunicação.

Temporais TLM-T

Dentro das características deste nível está a execução concorrente de processos independentes dentro do fluxo da simulação, permitindo-lhes todos acessar qualquer dos recursos do sistema como a bancos de memória mencionados na subseção anterior; uma ordem na execução deve ser respeitada, baseando-se nas dependências de sincronização para assegurar um correto funcionamento com a implementação de condições fundamentais, como, por exemplo, evitar a escrita de dados em áreas de memória temporalmente inacessíveis. A Figura 2.2 ilustra o caso de dois processos concorrentes, P_1 e P_2; um atraso funcional é inserido antes da ativação do processo P_{22} e do processo P_{12}. Estas anotações de tempo funcional, Tf, são executadas numa duração de tempo Te dentro do tempo total de execução. Não apresentam, porém, mudanças no sincronismo dos processos dependentes porque continuam sendo ativados pelos mesmos eventos P_{11} e P_{22} respectivamente, estando o início da execução do processo P_{12} ligado ao final do processo P_{22}.

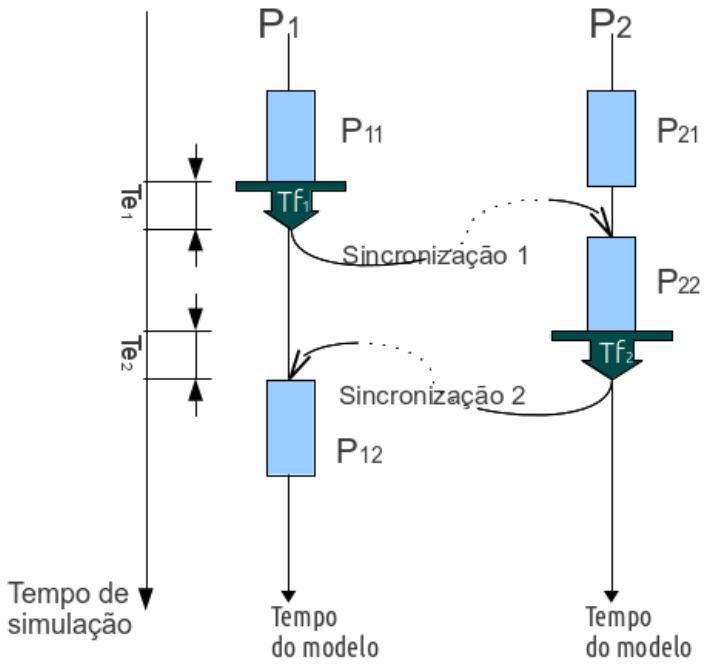


Figura 2.2: Execução de processos concorrentes em modelos TLM - Anexado de (GHENASSIA, 2005)

O modelo TLM-T incorpora anotações de tempo que influenciam na execução dos eventos, feitas a partir da incorporação de atrasos para ativar ou suspender processos. Estas anotações de tempos podem ser de computação, referindo-se ao tempo para realização de cálculos, caracterizando-se como uma função do sistema, e/ou de comunicação, referindo-se ao tempo total consumido para se acessar e transferir dados. Conseguem-se tais anotações pela análise das características da arquitetura dos módulos da plataforma, especificamente dos tempos do modelo comportamental dos componentes baseados no fluxo de controle, podendo ser tomados, por exemplo, os valores de tempo de melhor ou pior caso, ou valores da média. Considerando o exemplo da Figura 2.2 que ilustra o modelo, apresentando dois processos concorrentes onde

P_1 é de computação e P_2 é de comunicação. Diferente do exemplo de processos anterior, há a presença de dois tipos de anotações de tempo funcional T_f , de computação e de comunicação, que serão inseridas antes da ativação dos seus respetivos processos e representam uma anotação mais precisa dos processos. Não apresentam mudanças no sincronismo, mas o tempo de simulação é afetado em comparação ao exemplo anterior considerando-se que os T_f são distintos e mais realistas em relação ao sistema modelado, e também terão T_e diferentes sobre o tempo total de simulação.

Na Figura 2.1, pode-se classificar ao Modelo de Barramento com Arbitragem como TLM-T porque apresenta anotações tanto no domínio de computação como de comunicação.Uma eventual representação TLM-T CA ou BCA (TLM *Bit Cycle Accurate*) seria um modelo TLM-T associado a granularidade de computação (Modelo de Computação com Precisão por Ciclos) ou comunicação (Modelo de Barramento Funcional).

2.1.2 Influência dos modelos TLM na modelagem e simulação de sistemas

Segundo as considerações anteriores, se a precisão e fidelidade em representação do modelo são altas, os tempos de simulação serão igualmente altos. A Figura 2.3 mostra a diferença nos tempos de simulação ideais dos modelos RTL, CA e TLM definidos no padrão (OSCI, 2011b). A aceleração mais favorável e destacada é do modelo TLM, sendo mil vezes mais rápido na simulação. Trata-se de uma estimativa do modelo TLM-A contra os modelos CA e RTL; deve-se considerar que os fatores de simulação diminuiriam para o modelo TLM-T. Da mesma forma, a tendência ao decrescimento do esforço dos projetistas na prototipagem do sistema em níveis mais abstratos é representado na linha vermelha; considere-se que as unidades de tempo na modelagem do sistema podem ser horas até meses influenciado por diferentes fatores como redesenho, adição de funções, etc.

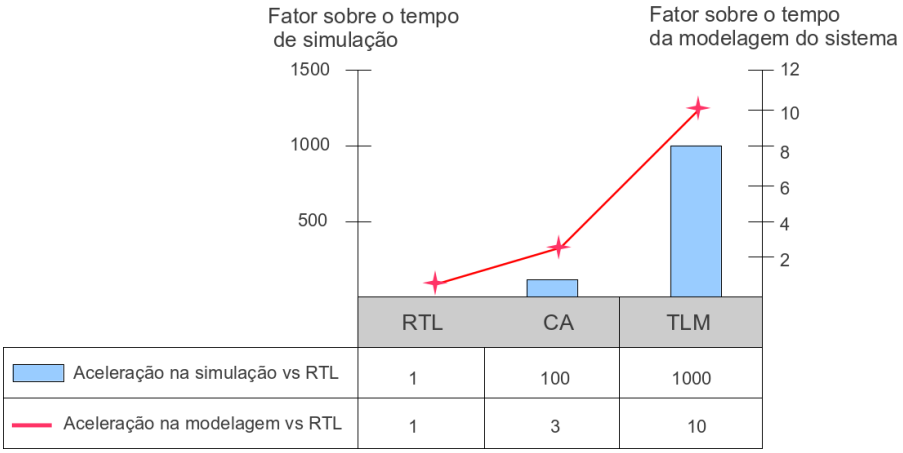


Figura 2.3: Velocidade de modelos em relação aos tempos de simulação - Anexado de (GHENASSIA, 2005)

2.1.3 Abstração e equivalência entre o modelo RTL e o modelo TLM

Em (BOMBIERI; FUMMI; PRAVADELLI, 2008), os autores apresentaram uma definição formal para a equivalência baseada em eventos entre os modelos RTL, referidos como CA, e TLM, referidos como TB (do inglês, *Transaction Based*). Os autores definem que um conjunto de processos que modela o sistema pode ser considerado uma série de eventos. Considerando-se que o modelo do sistema M possui um número n de entradas primárias, $PI =< I_1, ..., I_n >$, e m saídas primárias $PO =< O_1, ..., O_m >$, define-se o universo $\bigcup \Sigma^M$ como a sequência de eventos do modelo cujo comportamento F é definido,

$$F_M : \bigcup \Sigma^I \ \rightarrow \ \bigcup \Sigma^M \tag{2.1}$$

Sendo $\rightarrow$ a relação "ocorre antes de"na relação 2.1, indicando que ante um conjunto de eventos de entrada Σ^I relacionados as entradas primárias PIs o modelo produz um conjunto de eventos de saída Σ^M relacionados as saídas primárias POs. Na Figura 2.4 e na Figura 2.4(a) é apresentado um exemplo de um somador simples como seu respectivo modelo CA e TB. A relação do evento com o nível de abstração é importante já que existem diferenças no protocolo de comunicação; no modelo CA o protocolo é via interpretação de sinais e no modelo TB é mediante funções. Um evento no modelo CA ocorre ao ler-se uma PI ou escrever-se uma PO, enquanto no modelo TB ocorre quando uma transação inicia ou termina. O modelo CA da Figura 2.4 tem três PIs, clk, $dados_{IN}$ e $dados_{EN}$ e duas POs, $resultado_{OUT}$ e $resultado_{EN}$, que indicam a inserção de $dados1$ e $dados2$ para a soma pelo sinal $dados_{IN}$ em cada borda de subida do relógio em que o *flag* $dados_{EN}$ está habilitado. Cada operação de leitura e escrita tem dois eventos incluindo as mudanças sobre o sinal de *flag* e sem considerar o relógio clk, causando um total de oito eventos, dos quais cinco são sucessivos,

$${e_1}^{CA} \ \rightarrow \ {e_3}^{CA} \ \rightarrow \ {e_5}^{CA} \ \rightarrow \ {e_6}^{CA} \ \rightarrow \ {e_8}^{CA} \tag{2.2}$$

e três são concorrentes,

$${e_1}^{CA} \parallel {e_2}^{CA} \ , {e_3}^{CA} \parallel {e_4}^{CA}, {e_6}^{CA} \parallel {e_7}^{CA} \tag{2.3}$$

No modelo TB da Figura 2.4(b) não existem operações de habilitação e não considera-se ciclos de relógio. A leitura de $dados1$ e $dados2$ são operações de escrita, e a escrita do resultado na saída é uma operação de leitura. Cada transação tem dois eventos (inicio e fim) correspondendo um total de seis eventos,

$${e_1}^{TB} \ \rightarrow \ {e_2}^{TB} \ \rightarrow \ {e_4}^{TB} \ \rightarrow \ {e_5}^{TB} \ \rightarrow \ {e_6}^{TB} \tag{2.4}$$

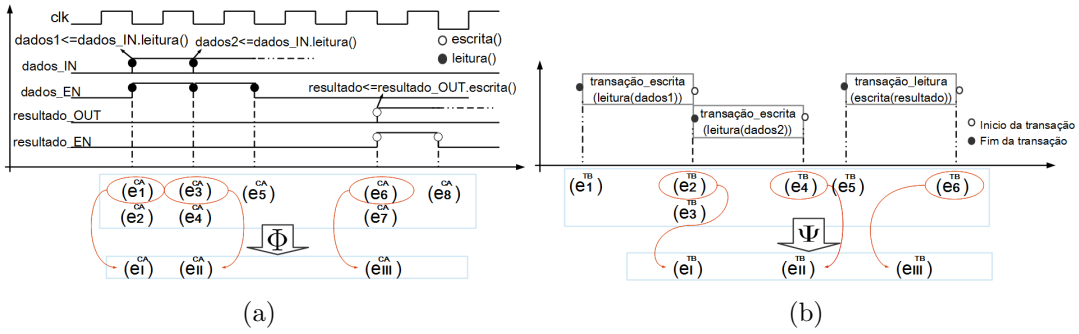
$${e_2}^{TB} \parallel {e_3}^{TB} \tag{2.5}$$

Para analisar a equivalência é usada a função de abstração Φ para o modelo CA e Ψ para o modelo TB definidas por,

$$\Phi \left(\Sigma^{CA}, R\right) = \Sigma^{CA'} \tag{2.6}$$

$$\Psi \left(\Sigma^{TB}, R\right) = \Sigma^{TB'} \tag{2.7}$$

com $R = PI, PO$, onde um evento $e \in \Sigma^{CA}$ é gerado por uma operação de leitura ou escrita num elemento R e não existem eventos e de CA com relação $\parallel$. Igualmente, um evento $e \in \Sigma^{TB}$ é



(a) (b)

Figura 2.4: Somador simples: (a) Modelo CA. (b) Modelo TB - Anexado de (BOMBIERI; FUMMI; PRAVADELLI, 2008).

gerado no fim da transação pela operação de escrita ou leitura sobre um elemento R e não existem e de TB com relação $\|$. Aplicando as funções de abstração sobre as definições 2.2,2.3,2.4,2.5 resulta,

$$\Phi({e_1}^{CA}, \ldots, {e_8}^{CA}) = ({e_1}^{CA}, {e_3}^{CA}, {e_6}^{CA}) = ({e_I}^{CA}, {e_{II}}^{CA}, {e_{III}}^{CA}) \tag{2.8}$$

$$\Phi({e_1}^{TB}, \ldots, {e_6}^{TB}) = ({e_2}^{TB}, {e_4}^{TB}, {e_6}^{TB}) = ({e_I}^{TB}, {e_{II}}^{TB}, {e_{III}}^{TB}) \tag{2.9}$$

O resultado das Equações 2.8 e 2.9 mostram a equivalência entre os modelos em diferentes níveis de abstração do somador acima mencionado.

2.2 Análise e modelagem de séries temporais

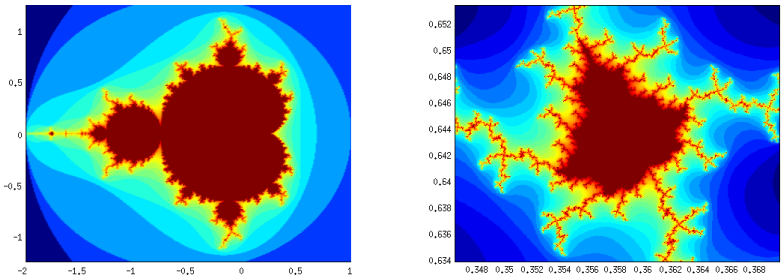
2.2.1 Fractais

Considera-se a um objeto matemático como fractal (do latim *fractus* que significa fracionado), se mantém a sua estrutura em diferentes graus de observação (LELAND et al., 1994 apud LIMA, 2008, p. 29). A Figura 2.5(a) mostra o conjunto de Mandelbrot chamado de "boneco de pão de mel", encontrando-se o mesmo formato em diferentes escalas de uma mesma dimensão, não importa quantas vezes seja expandido ilustrado na Figura 2.5(b). Esta característica corresponde à autossimilaridade do objeto e pode ser observada em diversos fenômenos em um sentido determinístico, como no conjunto de Mandelbrot, e em sentido estatístico, como as series temporais de hidrologia, séries de dados econômicas ou financeiras e o teletráfego de redes.

Os modelos autossimilares a diferença de modelos clássicos como o modelo de Poisson, permitem manter as características de alternância de períodos de surtos e suavidade de uma serie em diversas escalas de tempo (LELAND et al., 1994 apud LIMA, 2008, p. 29). Como se apresenta na Figura 2.6, as séries dos gráficos da esquerda são rastros de tráfego ethernet reais, as séries do meio são de tráfego sintético modelado por Poisson, e as séries da coluna direita são de tráfego simulado por um modelo autossimilar. Observa-se que nas séries do topo da figura, primeira fila, que são mais agregadas[1] (escala temporal=1s) em relação à segunda fila (escala

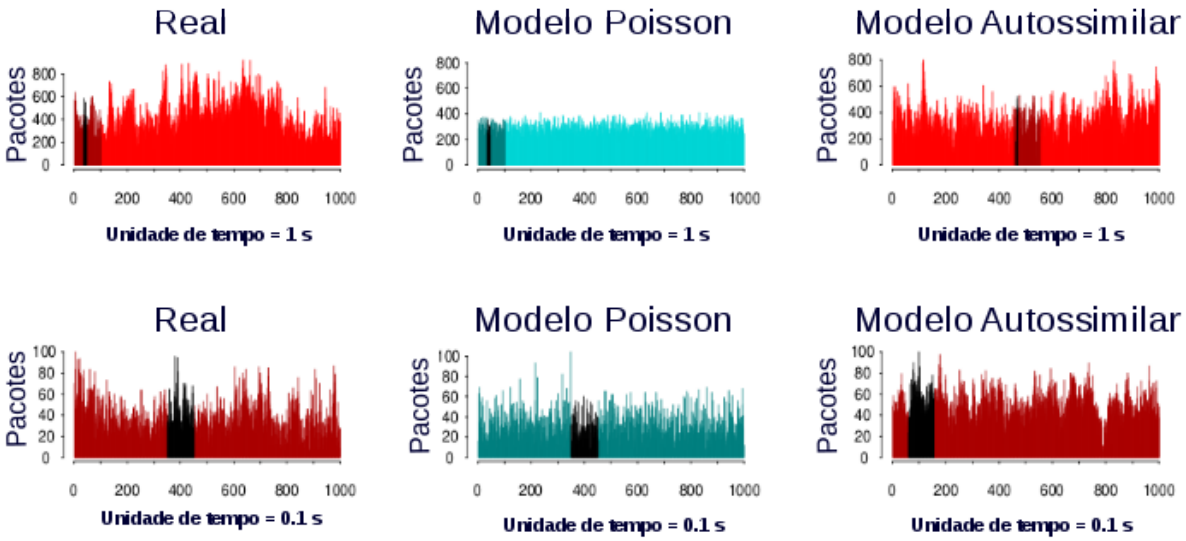
[1]Agregação é uma operação de mudança da escala temporal.

temporal=0.1s), o modelo gerado por Poisson não mantém a impulsividade da série original da segunda fila a diferença do modelo autossimilar, sendo o tráfego Poisson bastante suavizado.



(a) (b)

Figura 2.5: Conjunto de Mandelbrot: (a) Subconjunto no intervalo X[-2,1] e Y[-2,1]. (b) Subconjunto no intervalo X[0.348,0.368] e Y[0.634,0.652] - Gerado pelo código (CETTO, 2011).



(a) (b) (c)

Figura 2.6: Impulsividade de séries de tráfego em diferentes escalas de agregação - Anexado de (LELAND et al., 1994 apud LIMA, 2008): (a) Caso Real. (b) Modelo de Poisson. (c) Modelo Autossimilar.

2.2.2 Dependência de Longa Duração

O tráfego intrachip pode ser descrito como um processo estocástico formado por uma sequência de dados, sobre os quais aplicam-se as definições de séries temporais. Considerando o tráfego intrachip como um processo aleatório ou estocástico x_t de tempo continuo $t \in \mathbb{R}$ ou de tempo discreto $t \in \mathbb{Z}$ e cuja média é $E[x_t] = \mu_t$, a sua covariância será expressa por (SHUMWAY;

STOFFER, 2005):

$$\gamma_x(t+h,t) = E\left[(x_{t+h}-\mu_t)(x_t-\mu_t)\right] \tag{2.10}$$

onde h é o salto ou deslocamento no domínio de tempo conhecido como *lag*. Define-se a função de autocorrelação (FAC) de x_t como,

$$\rho_x(h) = \frac{\gamma_x(t+h,t)}{\sqrt{\gamma_x(t+h,t+h)\gamma_x(t,t)}} = \frac{\gamma(h)}{\gamma(0)} \tag{2.11}$$

No domínio da frequência, a x_t terá a função do periodograma $\hat{P}_x(f)$ definida pela Densidade Espectral de Potencia (DEP) com frequência normalizada em ciclos/amostra $-1/2 < f < 1/2$ como,

$$\hat{P}_x(f) = \frac{1}{n}\left|\sum_{t=1}^{n} x(t)\exp[4\pi^2 \mathrm{i}(t-1)f]\right|^2 \tag{2.12}$$

As séries temporais podem apresentar dependência de curta duração (SRD) e/ou longa duração (LRD). Existe LRD ou memória longa no processo x_t, quando $f \to 0$, a DEP da Equação 2.12 tender ao infinito. Ao cumprir-se esta condição, o processo x_t possui a singularidade $1/f^\alpha$ com $0 < \alpha < 1$ ($\alpha = 2H-1$) portanto pode ser chamado de ruído $1/f^\alpha$. Sendo H o parâmetro de Hurst cujos valores estão definidos entre $1/2 < H < 1$ para processos LRD e $0 < H \leq 1/2$ para processos com SRD respectivamente, quanto mais próximo H estiver de 1 maior é o grau de LRD (LIMA, 2008). Em um processo de ruído $1/f^\alpha$, a função de autocorrelação $\rho_x(h)$, para valores grandes do *lag* h, tem-se um decrescimento para zero extremamente lento do tipo hiperbólico, cumprindo-se em relação à autocovâriancia da FAC $\gamma_\rho > 0$:

$$\lim_{h\to\infty} \frac{\rho_x(h)}{\gamma_\rho h^{-1-\alpha}} = 1 \tag{2.13}$$

Da minimização da Equação 2.13, obtém-se a seguinte equação relacionada à singularidade na origem do espectro quando $f \to 0$,

$$\sum_{h=-\infty}^{\infty} (\rho_x(h)) = \infty \tag{2.14}$$

A Eq. 2.14 indica que as autocorrelações decaem para zero tão lentamente que não são somáveis, o que implica que a dependência estatística entre eventos distantes diminui muito lentamente com o aumento do *lag* h. Este comportamento é contrario ao apresentado em processos SRD, nos quais decresce a autocorrelação para zero de forma exponencial.

Para analisar as propriedades de um processo LRD x_t, por meio de sua agregação, define-se:

$$(X_i)^{(M)} = \frac{1}{M}\sum_{t=M(i-1)+1}^{M_i} x_t \tag{2.15}$$

onde o processo agregado $(X_t)^{(M)}$ de grau M corresponde a uma média móvel de blocos não sobrepostos de x_t de tamanho M. Sendo válida para todo processo LRD a seguinte propriedade com a função de variância σ^2 e c como uma constante:

$$\lim_{M\to\infty} \frac{\sigma^2\ ({X_t}^{(M)})}{M^{(2H-2)}} = c \tag{2.16}$$

o que significa que o numerador tem uma queda lenta quando M for maior, indicando a presença de memória longa nos valores da covariância.

2.2.3 Autossimilaridade

Define-se um processo estocástico autossimilar y_t $t \in \Re$ com parâmetro H, onde $c > 0$ é uma constante e $0 < H < 1$,

$$\{y(t)\} \ = \{c^{-H} y(ct)\} \tag{2.17}$$

Não todos os processos autossimilares são processos LRD (LIMA, 2008). Em (MELLO, 2006), é indicado o caso do movimento Browniano cuja classificação corresponde a um processo autossimilar estacionário de incrementos com parâmetro $H = 1/2$, associado ao ruido Gaussiano branco que não é LRD. Por outro lado, o caso dos processos estacionários exatos de segunda ordem apresentam LRD.

Sendo $\{y_k\}$, com $k \in \mathbb{Z}$, o processo estocástico estacionário em tempo discreto de y_t, define-se um processo autossimilar exato de segunda ordem com parâmetro H no intervalo $1/2 < H < 1$ se a sua autocovariância, com lag h, existe e é definida por:

$$\gamma_{y_k(h)} = \frac{{\sigma_{y_k}}^2}{2} \left[(h+1)^{2H} - 2h^{2H} + h - 1^{2H} \right] \tag{2.18}$$

e satisfaz,

$$\lim_{m \to \infty} \gamma_{Y_k(m)} = [(h+1)^{2H} - 2h^{2H} + h - 1^{2H}] \tag{2.19}$$

em que $\gamma_{Y_k(m)}$ é a função de autocovariância do processo agregado Y_k com nível de agregação m.

A equação 2.19 indica que $\gamma_{Y_k(m)}$ apresenta um decaimento hiperbólico. Portanto, a autossimilaridade de segunda ordem implica em LRD quando $1/2 < H < 1$ e demonstra que as estatísticas de segunda ordem do processo não mudam com a escala de agregação.

2.2.4 Modelagem de séries com dependência de longa duração

Em (LIMA, 2008, p.12), os autores definem que o modelo de uma série temporal x_t é obtido pela estimação de uma função invertível $h(.)$, e corresponde a:

$$x_t = h(\ldots, w_{t-2}, w_{t-1}, w_t, w_{t+1}, w_{t+2}, \ldots) \tag{2.20}$$

onde w_t é um processo Independente e Identicamente Distribuído (IID), denotado por $w_t \sim IID$, sendo também a inovação ou nova informação no instante t da série definida por,

$$g(\ldots, x_{t-2}, x_{t-1}, x_t, x_{t+1}, x_{t+2}, \ldots) = w_t \tag{2.21}$$

sendo $g(.) = h^{-1}(.)$

Entre os modelos paramétricos comportamentais usados na geração de tráfego agregado que cumprem com a Eq. 2.20 estão os modelos lineares Autorregressivos de Média Móvel (ARMA) e

Autorregressivo Fracionário Integrado de Média Móvel (ARFIMA) (LIMA, 2008, p.46). Existem outros métodos não paramêtricos como o modelo fracionário de ruido gaussiano (fGN) que permite a geração de séries sintéticas com LRD. Em (FAN; YAO, 2005), os autores apresentaram uma comparativa entre diversos lineares e não lineares, além de apresentar conceitos de modelos não paramêtricos. Neste trabalho só foram avaliados modelos paramêtricos lineares ARMA e ARFIMA.

Modelo Autorregressivos de Média Móvel (ARMA)

O modelo ARMA é uma generalização dos modelos Autorregressivos (AR), que correspondem aos filtros de tipo FIR (do inglês, *Finite Impulse Response*) ou *all-zero*, com os modelos de Média Móvel (MA) que correspondem aos filtros do tipo IIR (do inglês, *Infinite Impulse Response*) ou *all-pole*; possuem uma função de transferência do tipo IIR (DAS; PAN, 2011).

Um modelo Autorregressivo de ordem p, denotado $AR(p)$, é definido por:

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \ldots + \phi_p x_{t-p} + w_t \tag{2.22}$$

onde x_t é estacionário e w_t é ruído branco gaussiano, para $\phi_j, j \in \{1, \ldots, p\} \subseteq \mathbb{Z}_+$. A Equação 2.22 pode ser reescrita em relação ao operador de atraso para uma amostra B definido por $Bx_t = x_{t-1}$ como

$$\phi(B)x_t = w_t \tag{2.23}$$

sendo

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \ldots - \phi_p B^p \tag{2.24}$$

o operador autorregressivo.

Um modelo de Média Móvel de ordem q denotado $MA(q)$ satisfaz a equação:

$$x_t = w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \ldots + \theta_q w_{t-q} \tag{2.25}$$

onde x_t é estacionário, w_t é ruído branco gaussiano e q são os *lags* da média móvel, com $\theta_i, i \in \{1, \ldots, q\} \subseteq \mathbb{Z}_+$ A Equação 2.25 pode ser reescrita, em relação ao operador B, como

$$x_t = \theta(B)w_t \tag{2.26}$$

definindo-se

$$\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \ldots + \theta_q B^q \tag{2.27}$$

como o operador de média móvel.

Modelo Autorregressivo Fracionário Integrado de Média Móvel (ARFIMA)

Define-se um modelo $ARFIMA(p, d, q)$,

$$\phi(B)\triangle^d x_t = \theta(B)w_t \tag{2.28}$$

onde $\phi(B)$ é o operador da Eq. 2.24, $\theta(B)$ é o operador da Eq. 2.27 e w_t é ruído branco gaussiano. O operador de diferenças $\triangle^d = (1-B)^d$ ou *backshift* pode ter valores fracionários para $-1/2 < d < 1/2$ (HOSKING, 1981 apud DAS; PAN, 2011) e satisfaz,

$$(1-B)^d = \sum_{k=0}^{\infty} \binom{d}{k} (-B)^d \tag{2.29}$$

O modelo $ARFIMA(p,d,q)$ pode ser expresso como:

$$(1 - \sum_{i=1}^{p} \phi_i B^i)(1-B)^d x_t = (1 + \sum_{i=1}^{q} \theta_i B^i) w_t \tag{2.30}$$

2.2.5 Métodos de estimação do parâmetro H

O conceito do parâmetro H foi introduzido no trabalho do hidrologista Hurst (HURST, 1951 apud LIMA, 2008), e representa um índice do grau de LRD numa série temporal. Existem diversos estimadores do parâmetro de Hurst (BERAN, 1994), e cada um deles comporta-se diferentemente ante a presença de SRD e/ou LRD na série analisada como foi avaliado em (SHENG; CHEN; QIU, 2011). Nesta seção serão descritos alguns dos mais conhecidos.

Estimador R/S

O estimador R/S ou Distância sobre Desvio é baseado no estimador de Ajuste de Alcance Reescalado (em inglês, *Rescaled Adjusted Range Estimator*), introduzido em (HURST, 1951). Com ele, o parâmetro H é estimado pela regressão da curva $log - log$ de R/S versus N. O estimador R/S é definido por Q_N:

$$Q_N = \frac{1}{\hat{S}_N} \left[\underset{1 \leq K \leq N}{max} \sum_{j=1}^{K} (x_j - \mu_t) - \underset{1 \leq K \leq N}{min} \sum_{j=1}^{N} (x_j - \mu_t) \right] \tag{2.31}$$

onde $\hat{S}_N$ é o desvio padrão da média μ_t definido como:

$$\hat{S}_N = \sqrt{\frac{1}{N} \sum (x_i - \mu_t)^2} \tag{2.32}$$

e N é o número de amostras da série temporal. Segundo (SHENG; CHEN; QIU, 2011) este método é o menos preciso porque o valor de H é superestimado quando $0 < H \leq 0.6$ e subestimado quando $0.8 \leq H < 1$.

O método do Periodograma

Por este método, proposto em (GEWEKE; PORTER-HUDAK, 1983), é feita a estimativa do parâmetro H a partir do *log* da regressão linear do periodograma no domínio da frequência, calculando-se a função $\hat{P}_x(f)$ da equação 2.12 com *data tapering*, para redução de perda de potência na DEP (AUGUSTO, 2009), e *smoothing* para remoção da variabilidade por picos marcados em $\hat{P}_x(f)$. Em (SHENG; CHEN; QIU, 2011), os autores indicam que este método é influenciado pela presença de ruído estável com sinal-ruído menor que $30dB$, também ruídos impulsivos ou não gaussianos de baixa frequência podem causar mudanças significativas na estimativa.

Estimador de Whittle

O estimador de Whittle, apresentado em (WHITTLE, 1953), precisa da minimização de:

$$Q(\theta) = \int_{-0.5}^{0.5} \frac{\hat{P}_x(f)}{\hat{P}_x(f,\theta)} \mathrm{d}f \tag{2.33}$$

onde $\hat{P}_x(f)$ é o periodograma da equação 2.12 e $\hat{P}_x(f,\theta)$ é o PED teórica do modelo ARFIMA para x_t, e $\theta = [p, d, q]$ são os parâmetros desconhecidos. Este método é estável na presença de ruído impulsivo (SHENG; CHEN; QIU, 2011) pelo que segundo (JEONG; MCNICKLE; PAWLIKOWSKI, 2006) é o método mais preciso.

Estimador Wavelet

Define-se o estimador Wavelet como

$$\hat{H}_{[j1,j2]} = \frac{1}{2}\left[\frac{\sum\limits_{j=j1}^{j2} \varepsilon_j j S_j - \sum\limits_{j=j1}^{j2} \varepsilon_j j \sum\limits_{j=j1}^{j2} \varepsilon_j S_j}{\sum\limits_{j=j1}^{j2} \varepsilon_j \sum\limits_{j=j1}^{j2} \varepsilon_j j^2 - \left(\sum\limits_{j=j1}^{j2} \varepsilon_j j\right)^2} + 1\right] \tag{2.34}$$

em que $\varepsilon_j = \left(N \ln^2 2\right)/2^{j+1}$ e onde,

$$S_j = \log_2\left(\frac{1}{N_j}\sum_{k=0}^{N_j-1} w_{j,k}^2\right) \approx \log_2(\mathbb{P}_j) \tag{2.35}$$

O conjunto S_j define o espectro wavelet entre j_1 e j_2 de x_k com $k = 0, 1, ..N-1$, sendo a série em tempo discreto de x_t, com os coeficientes wavelet w_j, k com $k = 0, 1, ..., N_{j-1}$ e as escalas $j = 1, 2, ...J$ da sua Transformada Wavelet Discreta (DWT). Umas das implementações mais importantes é a introduzida em (ABRY; VEITCH, 1998). Similar ao método anterior, este método é robusto em relação ao ruído impulsivo (SHENG; CHEN; QIU, 2011), mas computacionalmente é mais simples e rápido (JEONG; MCNICKLE; PAWLIKOWSKI, 2006).

Alteração na estimação do H

As séries de tráfego com LRD podem apresentar ruido agregado, elementos com SRD e/ou periodicidade que alteram a estimação do parâmetro H. Em (MARKOVIć; KOCH, 2005) os autores compararam a estimativa de H por diversos métodos, entre os que estão R/S, periodograma e Abry-Veitch, separando os elementos periódicos das séries por métodos baseados em filtragem regressiva, filtragem de média móvel (MA), filtragem wavelet e análise espectral. Os autores ressaltaram a importância da remoção de elementos periódicos e concluíram em que os métodos wavelet e de análise espectral são mais precisos em relação aos outros métodos usados. No estudo proposto por (AUGUSTO, 2009), os autores avaliaram a influência de SRD no cálculo de H pelo estimador wavelet de Abry-Veitch avaliando seu espectro. Os autores propuseram um mecanismo de estimativa de H pela decomposição da série em níveis do espectro usando

sua transformada DWPT (do inglês, *Discrete Wavelet Packet Transform*) para a remoção do elemento SRD. Em (LUDESCHER et al., 2011) os autores confirmaram a influência dos três elementos acima mencionados em séries monofractais, como as analisadas nesta dissertação, e confirmaram a relevância da sua decomposição.

2.3 Decomposição de séries temporais

Segundo a decomposição clássica apresentada em (MAKRIDAKIS; WHEELWRIGHT; HYNDMAN, 1997), uma série x_t pode-se decompor segundo o método aditivo

$$x_t = T_t \ + S_t \ + R_t \tag{2.36}$$

e pelo método multiplicativo como

$$x_t = T_t \ \times S_t \ \times R_t \tag{2.37}$$

onde seus três componentes são identificados como Tendência (*Trend*), Periódico (*Seasonal*) e Restante (*Remainder*) denotados por T_t, S_t e R_t respectivamente.

A Tendência T_t representa variações a longo prazo, chamadas de "mudanças na direção", aumentando ou diminuindo a amplitude da série. O componente Periódico S_t está associado a uma frequência fixa que produz flutuações na série; também pode conter um componente cíclico que não possui uma frequência fixa. O Resto R_t é tudo o que não é nem Periódico nem Tendência, chamado de erro ou componente irregular.

Quando a magnitude do elemento S_t ou T_t não varia muito em relação á magnitude da série, o modelo aditivo da Equação 2.36 pode ser usado; por outro lado, é recomendável usar o modelo multiplicativo da Equação 2.37 quando a magnitude de S_T ou T_t é proporcional á magnitude dá série. A decomposição clássica por médias móveis (MA) pode ser resumida nos seguintes estágios:

1. Obter o componente T_t definido por

$$T_t = \sum_{j=-k}^{k} a_j x_{t+j} \tag{2.38}$$

   Suaviza-se a série com a uma média móvel $m-MA$, se o valor do período m da série fosse par, onde $k = (m-1)/2$. Define-se $a_j = [a_{-k}, \ldots, a_0, \ldots, a_{-k}]$ como os pesos simétricos da média móvel $a_j = 1/m$ cuja somatória é igual a 1. Caso m for ímpar, define-se $a_{-k} = a_k = 1/(2m)$ e os outros pesos da média móvel são calculados com $a_j = 1/(m+1)$.

2. Calcular a série sem o componente Tendência. Define-se para o modelo aditivo e multiplicativo, a partir das Equações 2.36 e 2.37, como $x_t - T_t$ e x_t/T_t respectivamente.

3. Calcular o componente S_t pela estimativa de médias nos intervalos obtidos pela divisão do tamanho da série sem Tendência (do item 2) entre seu período m. O período de S_t será m.

4. Estimar para o modelo aditivo $R_t = x_t - T_t - S_t$, a partir da Equação 2.36, ou $R_t = x_t / T_t / S_t$ para o caso do modelo multiplicativo, com a equação 2.37.

   A decomposição clássica apresenta limitações relacionadas á perda de dados na estimativa de T_t e R_t, além de não ser preciso quando existem mudanças do período no elemento S_t. Existem outros métodos mais robustos, como a decomposição com Loess, chamada de método SDL (do inglês, *Seasonal-Trend Decomposition with Loess*). Existem métodos mais avançados que podem, inclusive, facilitar o análise de séries multifractais, como o método *Multifractal Detrended Fluctuation Analysis* (MF-DFA) (LUDESCHER et al., 2011), mas a avaliação da precisão dos diversos métodos não fará parte do escopo deste trabalho.

Capítulo 3

Trabalhos Relacionados

Os primeiros geradores de tráfego incluídos na simulação de tráfego em NoCs adotaram tráfego sintético gerado por modelos de Poisson (WIKLUND; SATHE; LIU, 2004) e modelos de Markov (KIASARI et al., 2008). Foi a partir do trabalho em (VARATKAR; MARCULESCU, 2004), que se adotou a modelagem de tráfego por processos estocásticos autossimilares para aplicações multimídia; em tal trabalho, os autores propõem uma metodologia para a geração de tráfego sintético de um decodificador de vídeo MPEG2 descrito na ferramenta *Stateflow* do software MATLAB. As capturas de tráfego foram feitas nas entradas dos *buffers* de dois módulos com tarefas distintas. O parâmetro de Hurst (H) foi estimado com os métodos variance-plot, baseado na estimativa pelo gradiente do *log* da variância contra o *log* da escala de tempo, e análise R/S, a partir dos rastros (*traces*) obtidos na troca de informações da estrutura de comunicação, obtendo-se resultados similares. O parâmetro H foi usado posteriormente na geração de tráfego com dependência de longa duração (LRD) por intermédio de um método baseado na criação de processos autossimilares, conhecidos como Ruído Gaussiano Fracionário (fGN) para estimular uma NoC usado na área de teletráfego (JEONG; MCNICKLE; PAWLIKOWSKI, 1999); o parâmetro H foi também usado no dimensionamento dos *buffers* de entrada aos módulos com um modelo de Movimento Fracionário Browniano (FBM) (VéHEL; RIEDI, 1997) sendo que um dos objetivos do trabalho era determinar a influência da LRD na criação de filas.

Os resultados em (VARATKAR; MARCULESCU, 2004) vieram a confirmar a exitosa modelagem de tráfego com memória longa, entretanto já se conhecem atualmente métodos de maior precisão para a estimativa do parâmetro H, os quais incorporam a influência de processos SRD que coexistem nas séries originais de tráfego mas ainda com algumas limitações (AUGUSTO, 2009). Outro aspecto que representa uma limitante no trabalho de Varatkar e Marculescu é a modelagem do decodificador na ferramenta *Stateflow*. Apesar de já existirem trabalhos que integraram a semântica da *StateCharts* com a linguagem TLM (FINDENIG et al., 2010), aquele trabalho não indicou o uso de linguagens orientadas a modelagem no nível de sistema como SystemC e TLM, ou se a estrutura de comunicação intrachip corresponderia a uma modelagem no nível de sistema. Tal fato reforça a motivação desta dissertação de que as características de autossimilaridade estarão presentes ao se usar uma estrutura de comunicação mais abstrata na modelagem do sistema.

Alinhado com as constatações acima, em (SOTERIOU; WANG; PEH, 2006) os autores

introduzem a ferramenta Trident para geração de tráfego em redes intrachip. O Trident trabalha a partir de múltiplas capturas de tráfego em ambientes de simulação de três arquiteturas de multiprocessadores onde executaram diferentes aplicações, incluindo-se compressão de dados e codificação e decodificação de imagens e vídeo. O parâmetro de Hurst foi estimado pela análise R/S enquanto as séries de tráfego sintético foram geradas de forma similar à do trabalho proposto em (VARATKAR; MARCULESCU, 2004). Os autores estimaram a distribuição espacial de saltos (entre roteadores), calculada pela probabilidade de recepção e consumo de pacotes dos nós para diversas topologias (anéis, grelhas 2D e 3D e *fat tree*) em relação às distâncias entre os nós. A distribuição de injeção de pacotes na rede por cada roteador com os dados obtidos na simulação foi também estimada, e posteriormente modelados com uma distribuição Gaussiana a partir do desvio padrão, tendo sido usada para inserir o tráfego sintético em padrões como hot spot. Este trabalho (SOTERIOU; WANG; PEH, 2006) possui limitações similares ao trabalho de (VARATKAR; MARCULESCU, 2004) devido à utilização dos mesmos métodos para o análise das séries de tráfego. Ademais, as simulações foram feitas com modelos nível CA o que requereu um tempo de implementação e simulação considerável, tendo em conta que foram executadas trinta aplicações.

A plataforma Apolo, introduzida em (FLóREZ, 2006), está baseada na rede Hermes do Grupo de Apoio ao Projeto de Hardware (GAPH-PUCRS) (GAPH, 2011) e incorpora mecanismos de avaliação de desempenho por métricas em modelos TLM de NoCs dispostas nas topologias anel, grelha 2D, toroide ou circular entrelaçada. Neste trabalho foram usados geradores de tráfego sintético com um modelo de Poisson que foi encaminhado pelos roteadores da rede segundo um padrão transposto, *hot spot* ou pseudoaleatório. O principal limitante deste trabalho é que as séries usadas para estimular a rede não podem ser comparadas com séries originadas por alguma aplicação, consistindo a metodologia de uma avaliação de um cenário ideal.

Os trabalhos (SCHERRER; FRABOULET; RISSET, 2006a, 2006b, 2009a, 2009b) propuseram uma modelagem de tráfego a partir da captura em uma plataforma com três elementos (processador, cache e memória) na ferramenta Soclib (SOCLIB, 2011c), no nível CA, isolando-os da rede intrachip para evitar a influência de outros elementos processantes e das características da rede sobre o tráfego. A análise, chamada de Modelo de Geração de Tráfego Multifásico (MPTG), permite dividir em fases as séries de tráfego capturadas no ponto de medição indicado na Figura 3.1, que ilustra a plataforma de simulação, adotado no trabalho em questão, onde foram executadas as aplicações multimídia JPEG, JPEG2000, MJPEG, MP3, e MPEG2. Pode-se observar que a plataforma é composta por um processador MIPS ligado a uma memória cache de dados e instruções, o qual está conectado diretamente a uma memória RAM. Os processadores MIPS na plataforma Soclib são implementados por ISS, com um pipeline de três estágios (SOCLIB, 2011a). Neste trabalho definiram um formalismo das transferências entre a memória e o processador, resumido nos cinco elementos da seguinte sequência: endereço destino, comando de leitura (RD) ou escrita (WR), tamanho de transferência, atraso e tempo entre chegadas de transferência. Os autores capturaram os valores dos sinais na simulação da plataforma por arquivos VCD (em inglês, *Value Change Dump*), e extraíram os dados da transferência associados ao *throughput*[1].

[1] O *throughput* de um enlace é definido como a contagem dos dados ($bits, bytes, flits, pacotes$) por unidade

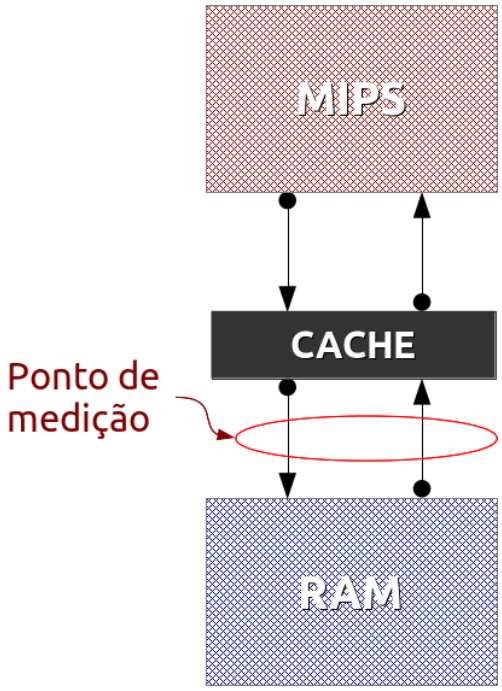


Figura 3.1: Arquitetura da plataforma de simulação - Anexado de (SCHERRER; FRABOULET; RISSET, 2009a)

Os elementos restantes da transação foram usados para identificar, em conjuntos de dados com similaridade entre as suas médias as fases da série através a partir do algoritmo *k-means*. Com as fases identificadas, avaliaram a dependência de longa duração de cada uma, estimando-se o parâmetro de Hurst pelo estimador wavelet. Para modelar as séries só com dependência de curta duração usaram modelos ARMA enquanto para as séries que possuíam dependência de longa duração ajustaram pelos modelos ARFIMA. No trabalho, os autores implementaram um gerador de tráfego CA para inserir as séries modeladas de acordo com diversos parâmetros estabelecidos pelo usuário (por ex. ordem e tempo de duração das fases) dentro de modelos CA de redes tipo DSPIN (PANADES; GREINER; SHEIBANYRAD, 2006) com topologia grelha 2D.

Diferente dos os trabalhos mencionados no começo desta seção, em (SCHERRER; FRABOULET; RISSET, 2009a), é apresentada a análise das séries de tráfego capturado. A Figura 3.2(a) apresenta o tráfego capturado da aplicação MPEG2 expresso como a contagem de flits[2] sendo o seu respectivo espectro em frequência apresentado na Figura 3.3(a). Observa-se que no espectro do tráfego MPEG2 existem picos nos níveis wavelet $j = 5, 9, 11$ e que na avaliação do parâmetro H (gradiente em linha vermelha), encontraram somente dependência de curta duração apesar de apresentar $H = 0.56$; tal situação indicaria, a rigor, a presença de LRD, diferindo dos resultados do trabalho (VARATKAR; MARCULESCU, 2004). Isto poderia ser explicado pelo fato de que o parâmetro H foi estimado para a fase identificada dentro do bloco vermelho da Figura 3.2(a). Na Figura 3.2(b), os rastros da aplicação MP3 são apresentados enquanto o seu

de tempo (ns, ms, s).

[2]Um *flit* é a menor partição de um pacote, presente num enlace de rede, e cuja unidade é geralmente dada em *bits*.

respectivo espectro é mostrado na Figura 3.3(b); observa-se que no tráfego existem mudanças notórias de regime, refletidas sobre a amplitude, as quais são as fases que a metodologia MPTG identifica.

O espectro da aplicação MP3 da Figura 3.3(b) corresponde à fase encerrada no bloco vermelho presente no rastro; este espectro apresenta picos na escala wavelet $j = 4, 6, 8$, o que dificulta a estimativa do parâmetro H representado pela linha vermelha. O tráfego da aplicação MJPEG é apresentado na Figura 3.2(c) e seu espectro na Figura 3.3(c). O espectro apresenta picos nas escalas $j = 6, 11, 13$, indicando um gradiente do estimador wavelet para $H = 0.77$, portando com a presença de LRD. Dentro do rastro apresentado só foi detectada uma fase encerrada num bloco vermelho, mas poderia conter mudanças semelhantes às apresentadas no tráfego da Figura 3.2(b), mas que a escala de observação não permite identificar facilmente. A presença de picos nos resultados anteriores motivou a avaliação de H por diversos métodos na presente dissertação para determinar a sua influência e propor uma metodologia de análise, de tal forma a permitir posteriormente a modelagem das séries originais capturadas.

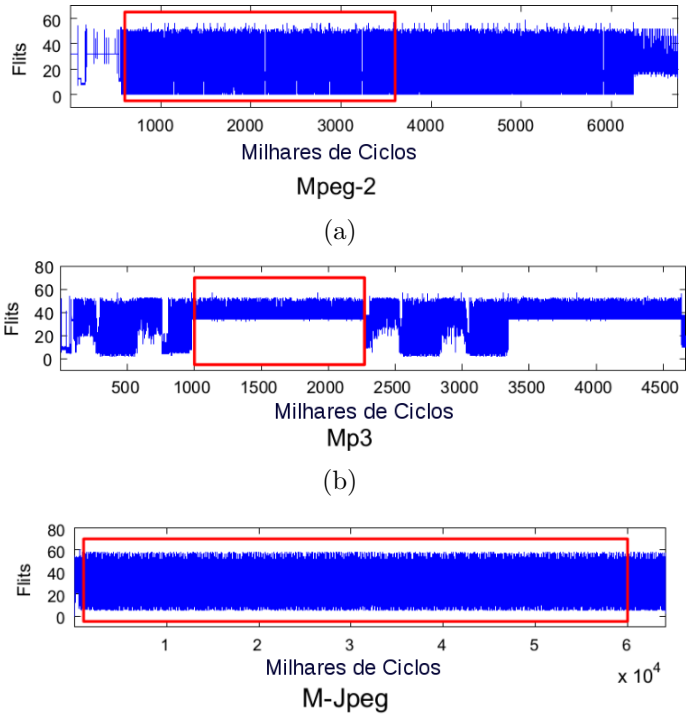


Figura 3.2: Tráfego das aplicações: (a) MPEG-2. (b) MP3. (c) MJPEG - Anexado de (SCHERRER; FRABOULET; RISSET, 2009a).

As experiências com a metodologia MPTG tinham como objetivo criar perfis de tráfego sem a plataforma de simulação da NoC, consumindo baixo tempo de simulação. Entretanto, quando os blocos IP foram substituídos por geradores de tráfego dentro da NoC, não foi observada uma aceleração significativa na simulação (SCHERRER; FRABOULET; RISSET, 2006b). Em (VIAUD; PECHEUX; GREINER, 2006), os autores apresentaram o modelo TLM com Tempo Distribuído (TLM-DT), presente na plataforma Soclib. O modelo TLMDT pode ser considerado

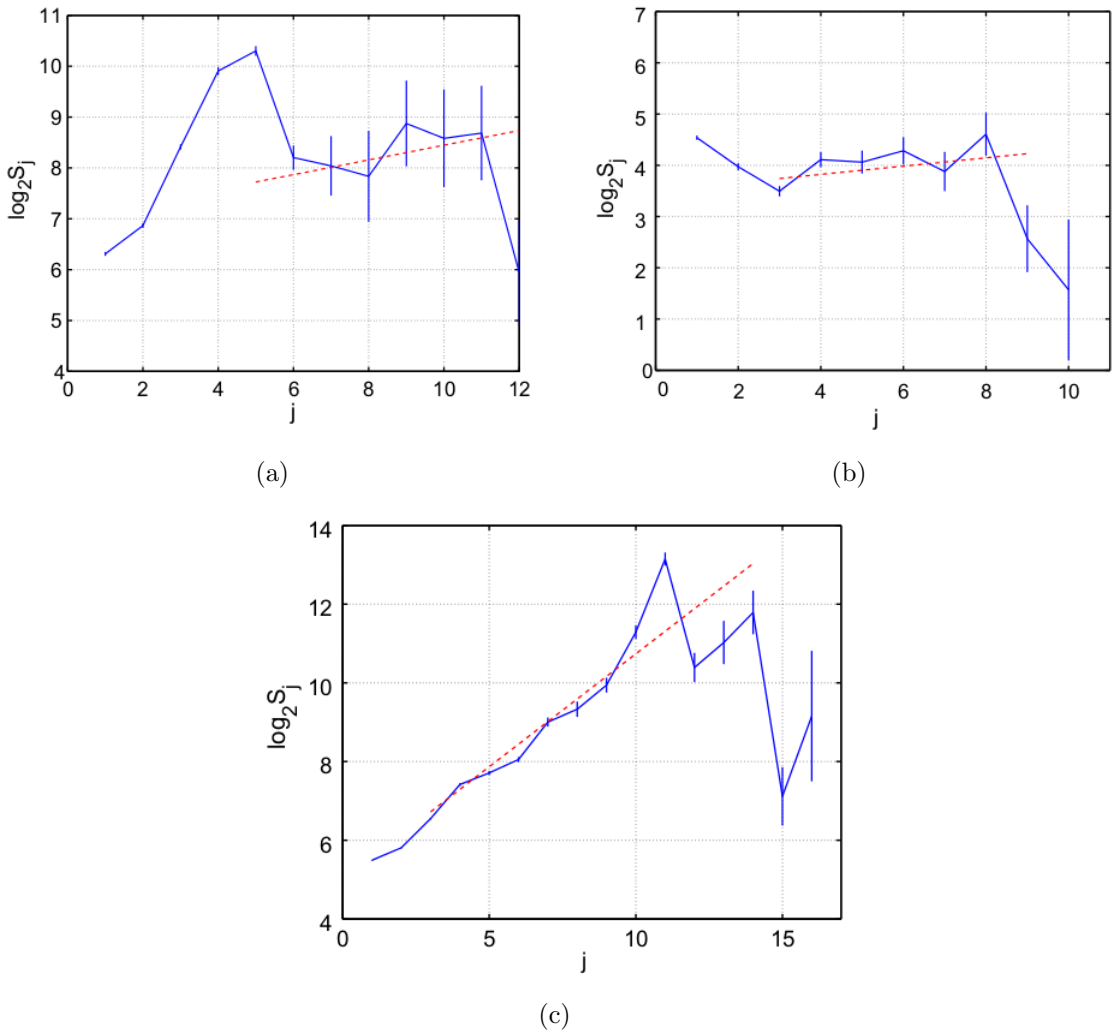


Figura 3.3: Espectro wavelet do tráfego das aplicações: (a) MPEG-2. (b) MP3. (c) MJPEG - Anexado de (SCHERRER; FRABOULET; RISSET, 2009a).

uma variante dentro dos modelos TLM-T com diferenças no estilo da codificação TLM SystemC (linguagem) ou uma extensão do padrão TLM. Esta mudança remove a sincronização do escalonador (*scheduler*) de execução da simulação passando o sincronismo pela anotação sobre as mensagens das transações o que permite ter uma descrição mais detalhada sem incrementar o tempo de simulação. O sistema é modelado como um conjunto de processos executados em paralelo, comunicando-se por canais, passando do paradigma de simulação por tempo global ao paradigma de simulação por tempo distribuído (CHANDY; MISRA, 1979 apud VIAUD; PECHEUX; GREINER, 2006). Os resultados deste trabalho sobre a aceleração na simulação em relação a modelos CA em Soclib demostram que, para um processador, o fator de redução de tempo é de aproximadamente dez vezes, podendo ser maior pela presença de um maior número de processadores. Este resultado evidencia a aceleração na simulação de sistemas pelo uso de modelos TLM e apresenta uma estimativa de erro em relação às transferências. Entretanto, as características estatísticas, como tráfego, não foram avaliadas.

Os trabalhos mencionados anteriormente fortalecem o objetivo desta pesquisa sobre a ava-

liação das características estatísticas de tráfego TLM capturado por simulação, analisando a viabilidade de seu uso na geração de tráfego sintético.

Capítulo 4

Projeto e Implementação da Metodologia

O objetivo principal desta dissertação é mostrar a viabilidade de ajuste do tráfego intrachip obtido por simulação em TLM a modelos de séries autossimilares, como explicitado anteriormente. Neste capítulo apresentamos as ações necessárias para a consecução deste objetivo, fundamentadas na motivação e na teoria propostas nos capítulos anteriores.

4.1 Metodologia de Análise e Ajuste de Tráfego Intrachip em TLM

A metodologia proposta neste trabalho para que o projetista possa analisar e ajustar o tráfego intrachip em TLM a modelos de séries autossimilares é ilustrada na Figura 4.1. A obtenção de séries de tráfego sintéticas pode ser feita a partir dos rastros obtidos por simulação e pode ser resumida em quatro estágios, enumerados a seguir:

1. O projetista deverá montar uma plataforma de simulação no nível TLM composta por um módulo ativo (mestre) e outro passivo (escravo), para capturar o fluxo de dados referente à comunicação entre eles.

2. A captura de tráfego consiste na interpretação das informações coletadas durante as transações de comunicação entre os componentes, na simulação da plataforma, gerando rastros.

3. Os rastros capturados devem ser analisados para se determinar suas características estatísticas.

4. As características avaliadas no estágio anterior serão usadas para o ajuste dos rastros de tráfego a modelos autossimilares.

A implementação de cada um das etapas acima será descrita em seções seguintes.

Figura 4.1: Metodologia de análise e ajuste de tráfego TLM

4.2 Implementação da Metodologia

4.2.1 Especificação da Plataforma De Simulação TLM

A plataforma Soclib

A SocLib (SOCLIB, 2011b) é uma plataforma de código aberto e gratuito concebida para prototipagem de MPSoCs, realizada como projeto da Agência Nacional de Pesquisa da França (ANR) e mantido pelo Laboratório de Informática de Paris (LIP6) (SOCLIB, 2011c). A SocLib incorpora na sua biblioteca descrições modulares de hardware de componentes usados no projeto de sistemas integrados nos níveis de abstração CABA (do inglês *Cycle Accurate/Bit Accurate*) e TLMDT[1]. A lista dos componentes comerciais modelados e presentes em SocLib é apresentada na Tabela 4.1, onde figuram processadores, memórias, barramentos e NoCs, controladores, processadores digitais de sinais entre outros. Os módulos de hardware são representações mais abstratas de modelos reais e sintetizáveis, que visam facilitar o reúso no fluxo do projeto. Os módulos estão descritos na linguagem SystemC (OSCI, 2011a) com a sua biblioteca TLM (OSCI, 2011b) e todos estes componentes trabalham com o protocolo de comunicação VCI (GROUP, 2001). A SocLib também incorpora plataformas de teste e tutoriais para aprendizagem, os quais foram utilizados no desenvolvimento da parte experimental deste trabalho.

Tabela 4.1: Componentes de hardware do Soclib - Anexada de (GREINER; SOCLIB, 2007)

Processadores de Propósito Geral	MIPS32, SPARC V8, ARM-V6T, PowerPC 405, Lattice Mico32, Nios2, Microblaze
Processadores Digitais de Sinais	ST231, TMS320 C62
Barramentos e Crossbars VCI	Generic System Bus, Pibus, Avalon, Token Ring, Local Crossbar
Bridges VCI	PCI, USB, CAN, I2C, Wishbone
NoCs	VCI Generic Micro Network, DSPIN Micro Network
Memórias e controladores	SRAM Embarcada, SDRAM Externa
Coprocessadores dedicados	DWT, DEM, MOD, LDPC, TC1000, TC3000
Utilitários gerais	Controlador de Interrupções, Dispositivo de bloqueio de memória Locks, Controlador DMA, Controlador MWMR, Controlador Frame Buffer, Controlador de uso de disco

[1]A partir de aqui referidos nesta dissertação como CA e TLM

Elementos da plataforma TLM

Nesta dissertação a plataforma SocLib e sua biblioteca de módulos foram usadas para a montagem de um sistema formado pela tripla <*processador, memória cache e memória principal*> para a medição do *throughput* do fluxo de dados de saída do processador, segundo a proposta de (SCHERRER; FRABOULET; RISSET, 2009a). Adicionalmente foi usada uma tela denominada *TTY*, cujo fluxo de entrada não foi capturado, uma vez que teve o propósito exclusivo de monitoramento da execução do código de aplicações.

A Figura 4.2 apresenta a plataforma TLM para simulação, onde o modelo do processador usado é um simulador de conjunto de instruções (do inglês, *instruction set simulator*, ISS) da arquitetura MIPS32. Tal simulador está dentro de um *wrapper* TLM que incorpora na sua descrição a memória *cache* de dados e de instruções, assim como também a interface VCI (SOCLIB, 2011b). O processador está conectado a uma memória RAM por uma microrrede Vgmn (SOCLIB, 2011d), numa configuração de barramento, já que usamos uma ligação direta sob o protocolo de comunicação VCI (GROUP, 2001), ambos descritos no nível TLM. O barramento VCI está formado pelos canais de comunicação Comando e Resposta, associados a atividades de escrita e de leitura, respectivamente; são representados por dois fios que operam por transações. Códigos de aplicações são compilados e inseridos no ISS. Duas aplicações diferentes são adotadas neste trabalho, sendo os detalhes fornecidos na seção de experimentos.

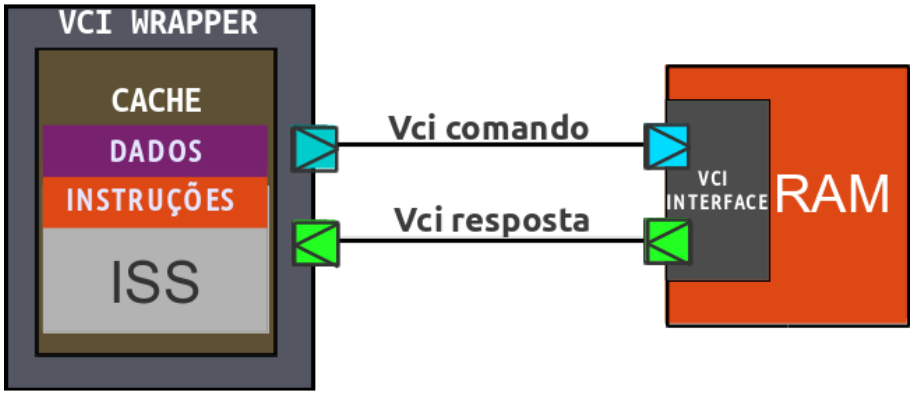


Figura 4.2: Plataforma de simulação no nível TLM

4.2.2 Formalismo de Transferências e Captura do Tráfego Intrachip

A comunicação iniciada do processador para a memória gera uma transferência de saída T_{out} enquanto a comunicação reversa gera uma transferência de entrada T_{in}. Na plataforma TLM da Figura 4.2 definimos um tamanho de transferência mínimo em 4 *bytes* e máximo em 32 *bytes*, adotando esta característica de exemplos nativos do SocLib. Uma transferência maior que o tamanho mínimo implica em um envio consecutivo de dados de tamanho de 4 *bytes*, gerando uma comunicação conhecida como rajada ou *burst*. Os dois canais usados na comunicação só possuem três elementos: o *payload* ou carga da transferência, a fase (*phase*) que sinaliza a habilitação do canal para Comando ou Resposta e o tempo estimado da transferência segundo o paradigma distribuído (VIAUD; PECHEUX; GREINER, 2006; SOCLIB, 2011e).

A transferência de saída entre processador e memória, T_{out}, é formulada, seguindo-se o formalismo de transferências para a estimativa de *throughput* do enlace de saída do processador adotado por (SCHERRER; FRABOULET; RISSET, 2009a) e segundo a teoria da Seção 2.1.3, da seguinte maneira:
T_{out} = {Tipo de transferência (Comando ou Resposta), Endereço do *payload*, Tamanho do *payload*, Dados do *payload*},
associado ao tempo (ns) em que ocorre. Neste nível indicamos que uma transferência é igual a uma transação.

A Figura 4.3 ilustra o formalismo adotado para as transações como uma função no domínio de ciclos de relógio. Cada ciclo, tem um período de 1 ns, indicando que para cada bloco T_{out} do canal de Comando, com duração de n ciclos ou ns, existe uma transação de resposta T_{in} no canal de Resposta com m ciclos ou ns de duração. O uso da escala de 1 ns em TLM é apenas referencial, pois a definição de uma escala de tempo absoluto é necessária em SystemC; o foco deve ser no ciclo de relógios, já que permite a medição das instruções do processador. O formalismo é ilustrado na Figura 4.3(a) com duas transações T_{out1} e T_{out2}, com um tempo de chegada i, onde $i \geq 0$, e existe uma resposta T_{in1} associada a T_{out1}. No gráfico da Figura 4.3(b), com a informação capturada das transações, definimos o *throughput* como a contagem de bytes no tamanho de pacotes e/ou *payload* por unidade de tempo (ns). A partir deste fluxo de dados capturados estabelecemos uma janela de observação T_{obs}. O gráfico da Figura 4.3(c) ilustra o tráfego intrachip (em bytes) por uma unidade chamada *bins* que representa a mudança na escala temporal pela *agregação* dos dados em T_{obs}, que é usado como fator do tempo (ns); estima-se o valor do tráfego num instante pela agregação ou somatória dos valores do tamanho do pacote e/ou *payload* no intervalo da janela de observação anterior. Em analogia às séries temporais, x_t representa as séries de tráfego intrachip capturado referidos como rastros (*traces*) originais a partir dos quais as análises são feitas.

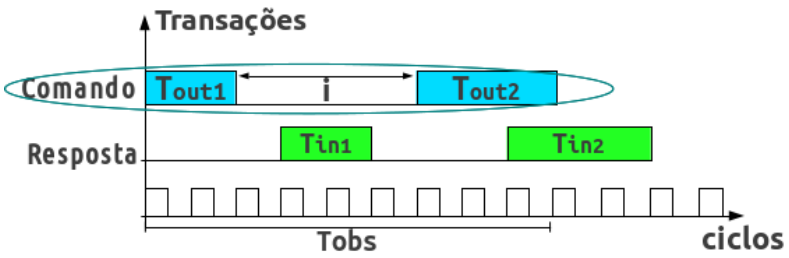
4.3 Análise de Tráfego Intrachip em TLM

Para atender os objetivos fixados no início do trabalho, avaliamos os rastros originais TLM segundo as características estatísticas definidas na Seção 2.2, para posteriormente analisar os resultados. A linguagem R (FOUNDATION, 2012), amplamente usada na computação estatística, foi usada para o desenvolvimento de *scripts* para os diversos cálculos necessários à análise. As operações dos *scripts*, são representadas como um fluxo de forma sintética na Figura 4.4. Os cálculos referem-se aos seguintes itens:

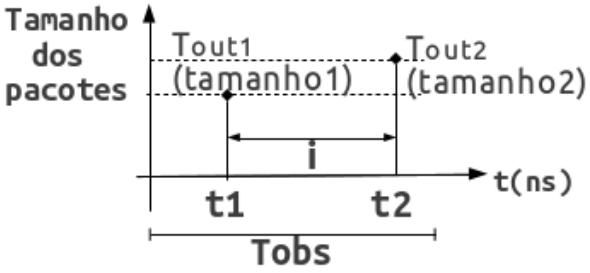
1. A função de autocorrelação (FAC), para estabelecer o grau de semelhança entre as observações dos traces e identificar pela observação do comportamento registrado em gráfico, a queda hiperbólica, associada à dependência de curta duração (SRD), ou exponencial, associada à dependência de longa duração (LRD).

2. O periodograma, para determinar a presença ou ausência da singularidade $1/f^{\alpha}$, cuja existência é associada à LRD.

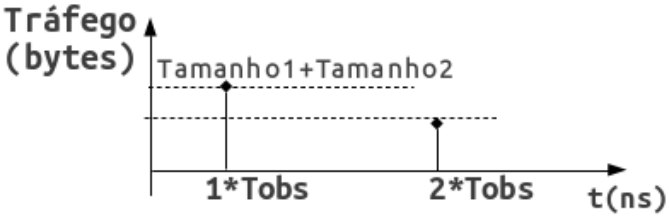
3. O parâmetro H como indicador da dependência de SRD ou LRD, obtido pelos métodos:



(a)



(b)



(c)

Figura 4.3: Formalismo adotado. (a) Transações. (b) *Throughput*. (c) Tráfego.

(a) Range over Deviation (R/S).

(b) Periodograma.

(c) Whittle.

(d) Estimador Wavelet de Abry-Veitch.

4. A representação em níveis de energia das séries pelo espectro Wavelet usado no estimador de Abry-vetich proposto em (ABRY; VEITCH, 1998), procurando entender possíveis inconsistências nos resultados do parâmetro H com os outros métodos, evidenciadas pela presença de picos, associados a elementos SRD como é discutido em profundidade em (AUGUSTO, 2009). O espectro representa a quantidade de energia nos instantes de tempo $2^j k$, onde j é o nível da sua escala obtida e $k \in \mathbb{Z}$.

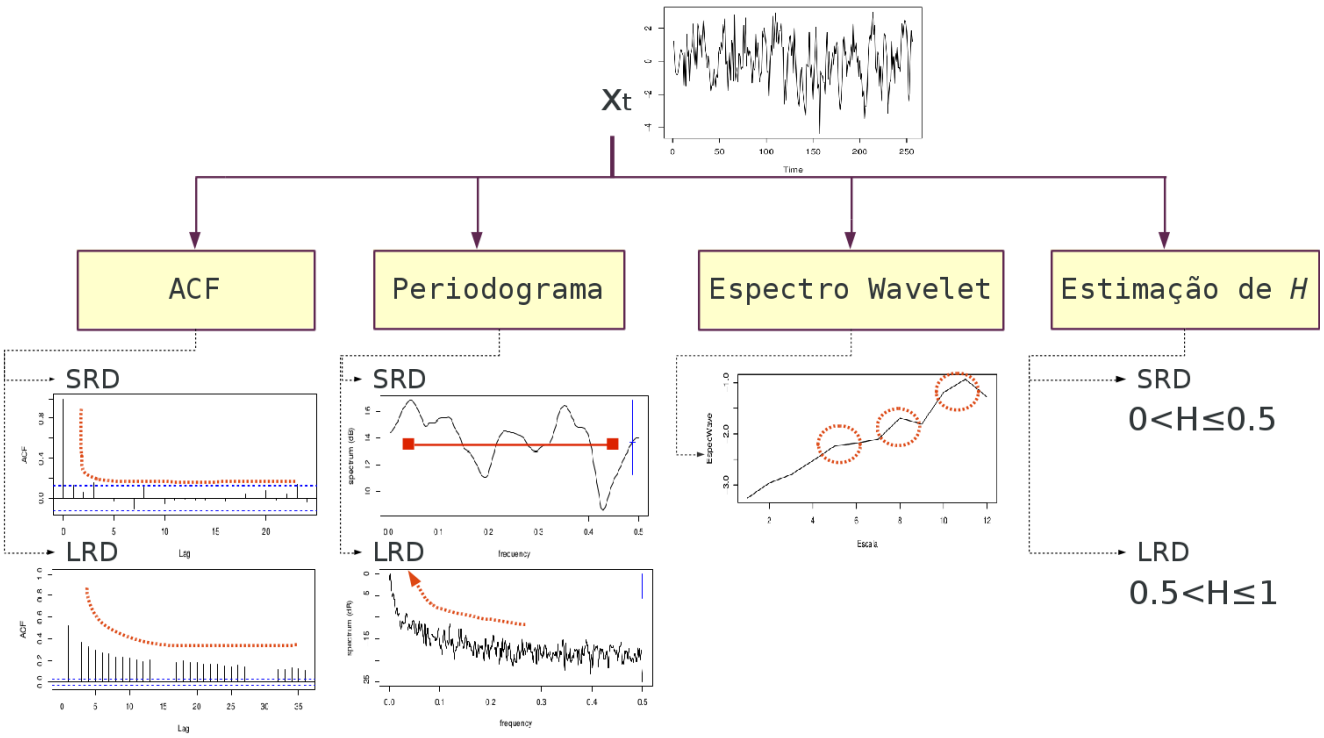


Figura 4.4: Metodologia de análise estatística do tráfego

4.4 Modelagem de Tráfego TLM como Série Autossimilar

Nos resultados dos espectros wavelet, pode ocorrer a presença de picos, o que indicaria que a determinação direta do parâmetro de Hurst poderia ficar distorcida pela existência de componentes periódicos (geradores dos picos). Nesta dissertação, é proposta para este cenário, a implementação da metodologia apresentada na Figura 4.5 para o ajuste mais preciso do tráfego intrachip TLM original a modelos autossimilares para geração de séries sintéticas. A estratégia consiste na decomposição das séries originais pela identificação de seus elementos periódicos originadores dos picos observados na análise espectral. Assumimos que a frequência dos picos é a frequência do elemento periódico da série. Este procedimento é apresentado na Figura 4.5 e foi implementado como *script* na linguagem R.

1. A primeira etapa do procedimento mostrada na Figura 4.5 consiste na decomposição da série original x_t por médias móveis (MA) em três componentes denominados como Tendência T_t, Periódico S_t (*Seasonal*) e Resto R_t (*Remainder*), de acordo com a formulação da Seção 2.3, onde a Equação 2.36 corresponde ao método aditivo e a Equação 2.37 ao método multiplicativo. Para a decomposição da série de forma automática, pode-se usar o comando *decompose*, incorporado na linguagem R (MEYER, 2012).

2. Avalia-se estatisticamente cada um dos componentes T_t, S_t e R_t como indicado na Fig. 4.4 para ajustar um modelo para cada um deles. Pode-se ajustar um modelo *ARMA(p,q)* e um modelo *ARFIMA(p,d,q)* para os componentes com SRD e LRD respectivamente. Depois

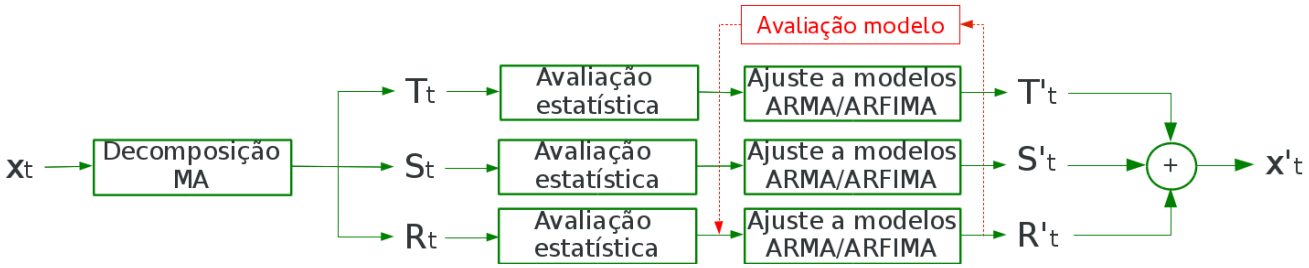


Figura 4.5: Decomposição e ajuste das séries de tráfego intrachip TLM

de cada ajuste avaliam-se os resíduos do modelo por FAC procurado baixa autocorrelação e periodograma observando uma resposta em frequência quase plana. Também, observa-se o histograma e o *Quantile-Quantile plot* normalizado, procurando-se uma resposta do tipo ruído Gaussiano. Na Fig. 4.6 apresenta-se a avaliação de ruido gaussiano que considera-se um caso ideal de resíduos onde o $qq-plot$ e histograma tem distribuição normal, Fig. 4.6(a) e 4.6(b) respectivamente. As Fig. 4.6(c) e 4.6(d), da FAC e periodograma, apresentam autocorrelação dos dados muito baixa e uma resposta na frequência estável.

Pode-se obter um ajuste "manual"para cada componente, ou no caso "automático", usar o comando *auto.arima* e *arfima* do pacote *forecast* (HYNDMAN, 2012), que ajusta modelos ARMA e ARFIMA respectivamente.

3. Com os modelos ajustados, geram-se séries sintéticas Tendência' T'_t, Periódico' S'_t e Restante' R'_t. Pode-se usar o comando *fracdiff.sim* e *simulate.Arima* do pacote *fracdiff* (MAECHLER, 2012) e *forecast* respectivamente, para gerar séries sintéticas a partir dos coeficientes dos modelos ajustados. A série sintética total x'_t é a reconstrução das Equações 2.36 e 2.37 com estas séries sintéticas.

4. Finalmente, avalia-se estatisticamente x'_t e pode-se compará-lo com os resultados da avaliação de x_t.

4.5 Formulação CA para Validação

Para a validação da metodologia descrita na seção anterior, os resultados obtidos podem ser comparados com resultados de referência. Para isto, pode-se usar o tráfego gerado através da simulação de um sistema com aplicações equivalentes ao modelo TLM, porém descrito no nível CA. As características estatísticas obtidas pelo modelo CA podem ser capturadas de forma idêntica àquelas apresentadas anteriormente e utilizadas para comparação.

A plataforma no nível CA implementada contém as mesmas aplicações do nível TLM. A plataforma CA da Figura 4.7 apresenta o processador MIPS32 com sua memória *cache* ligados à memória RAM pelo barramento VCI, implementado por diversos sinais (GREINER; SOCLIB, 2007) *bit-accurate*. Todos estes componentes estão descritos no nível CA; o processador MIPS32 usado é modelado como o ISS usado na simulação em TLM, porém ligado a uma estrutura CA de memória *cache* de dados e instruções por um *wrapper* ISS (para compatibilizar a granularidade

dos sinais). A memória *cache* está embarcada dentro de um *wrapper* que implementa sua interface de comunicação VCI, diferentemente da plataforma TLM, onde o *wrapper* contém ambas a memória *cache* e a interface VCI. Devido ao alto nível de abstração para a descrição TLM do barramento, não existe neste caso o conceito de sinais, diferentemente da plataforma CA; a Figura 4.8 apresenta o barramento VCI, de acordo as definições do padrão VCI (GROUP, 2001), formado por:

- $COMP = \{$*Command, Address, Packet Length, Source ID, Thread ID*$\}$, - sinais compartilhados por seus canais

- $CMD = \{$*Valid, Acknowledge, End of Package, Packet Lenght, Packet ID, Write Data*$\}$, - o canal de saída do processador ou canal de Comando

- $RSP = \{$Error, Read Data, Packet ID, End of Package, Valid, Acknowledge$\}$ - o canal de saída da memória ou canal de Resposta.

Pelo protocolo definido para o padrão VCI, o elemento base da comunicação a ser transferido é chamado de célula, cujo tamanho mínimo foi definido em 4 *bytes*. Uma célula é transferida por leitura ou escrita via um canal, de acordo com o valor de Comando (CMD), no ciclo em que *Valid* e *Acknowledge* têm valor $'1'$; são usados CMDVAL e CMDACK, ou RSPVAL e RSPACK, para o canal de Comando e Resposta, respectivamente. O sinal *Source* (SRC) define o componente de origem da transferência que, nestas plataformas, pode ser a memória ou o processador. O tamanho de uma transferência é definido por *Packet Lenght* (PLEN) e seu destino, se for o caso de escrita sobre a memória, por *Address* (ADDR). Esta também define a sua origem, no caso de leitura da memória. Sendo o tamanho máximo de PLEN 32 *bytes*, fica determinado o número de rajadas ou *bursts* de tamanho mínimo. Cada canal possui a informação do pacote, consistindo de: um número de identificação, *Packet ID* (PKTID), e os dados transferidos por WDATA e RDATA para escrita e leitura, respectivamente. Ao se finalizar a transferência, o sinal *End of Package* (EOP) sinaliza tal evento . O protocolo VCI possui outras funcionalidades, como identificar transferências em processos *multi-thread* pelo *Thread ID* (TRDID), erros de leitura (RSPERROR), tamanho restante do pacote, etc.

Para a obtenção do tráfego nos canais de comunicação, define-se o formalismo de transferência
$T_{out} = \{$*Command* (CMD), *Address* (ADDR), *Packet lenght* (PLEN), *Write Data* (WDATA)$\}$ associado ao ciclo, porém expresso em unidades de tempo (ns) em que ocorre.

A avaliação estatística indicada na análise de tráfego, da forma explicitada nas seções anteriores, fornece um mecanismo quantitativo para detecção de SRD ou LRD, permitindo a comparação entre séries típicas CA e TLM . Pode-se determinar se as características relevantes são mantidas nos diferentes níveis de abstração e quão semelhantes tais características podem estar nas séries.

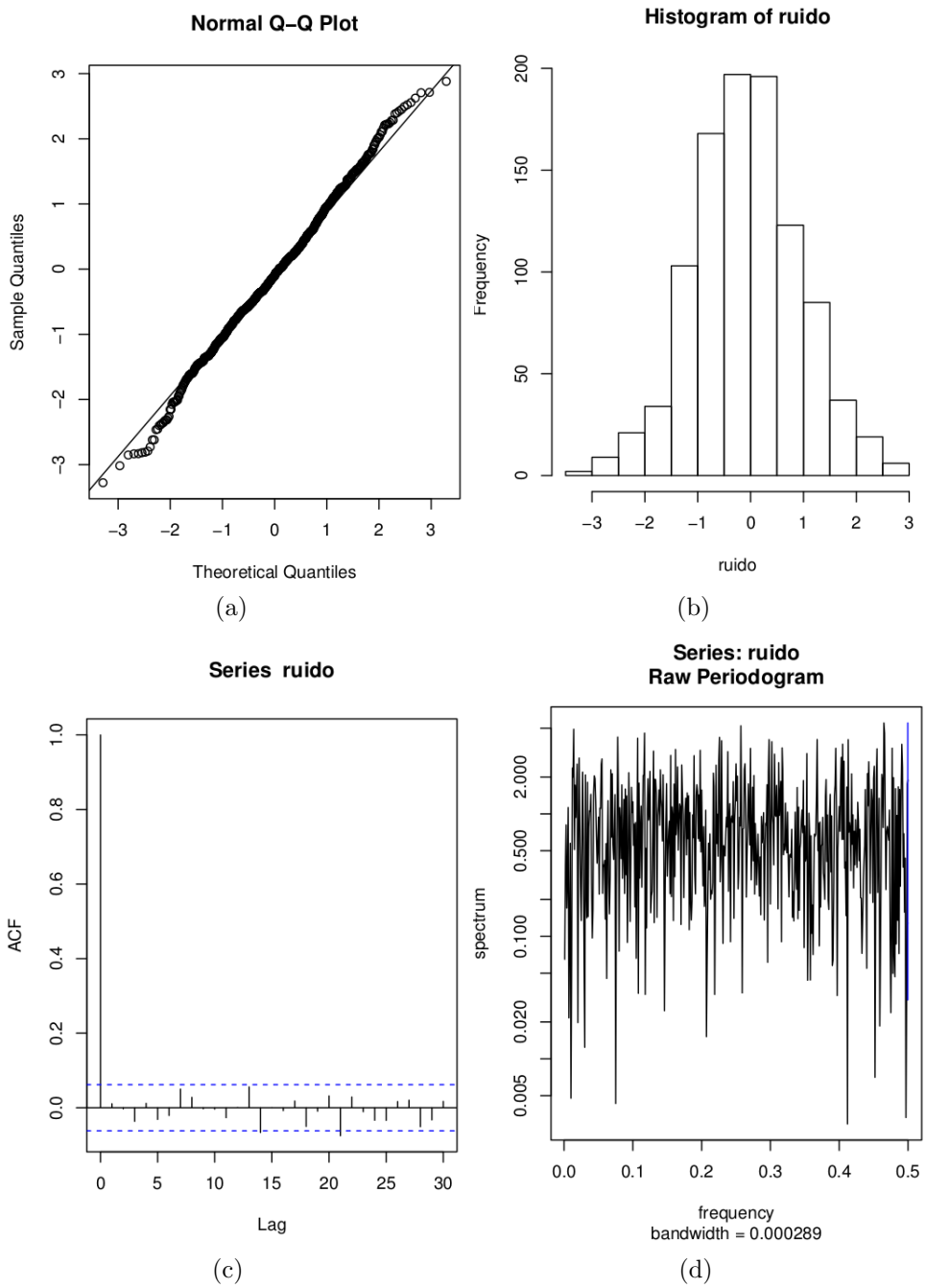


Figura 4.6: Avaliação de ruido gaussiano representado o caso de resíduos ideal. (a) QQplot. (b) Histograma. (c) FAC. (d) Periodograma.

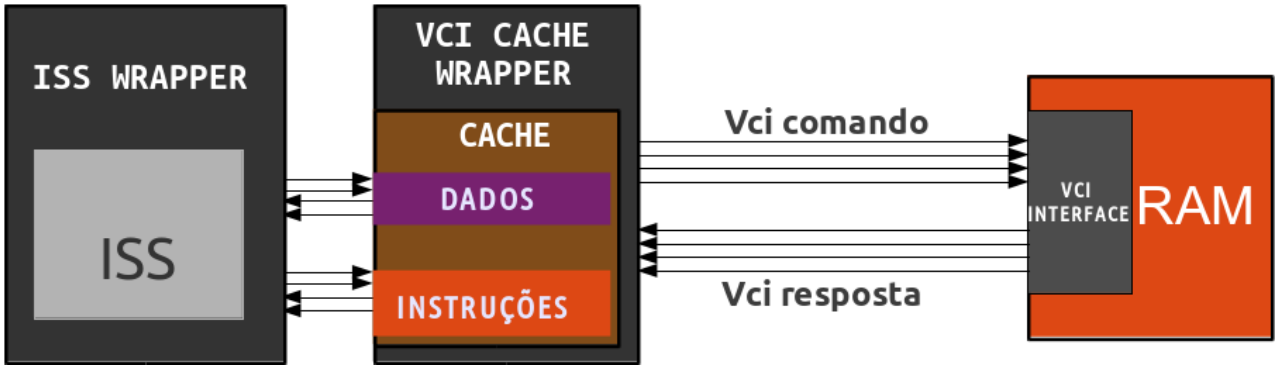


Figura 4.7: Plataforma de simulação no nível CA

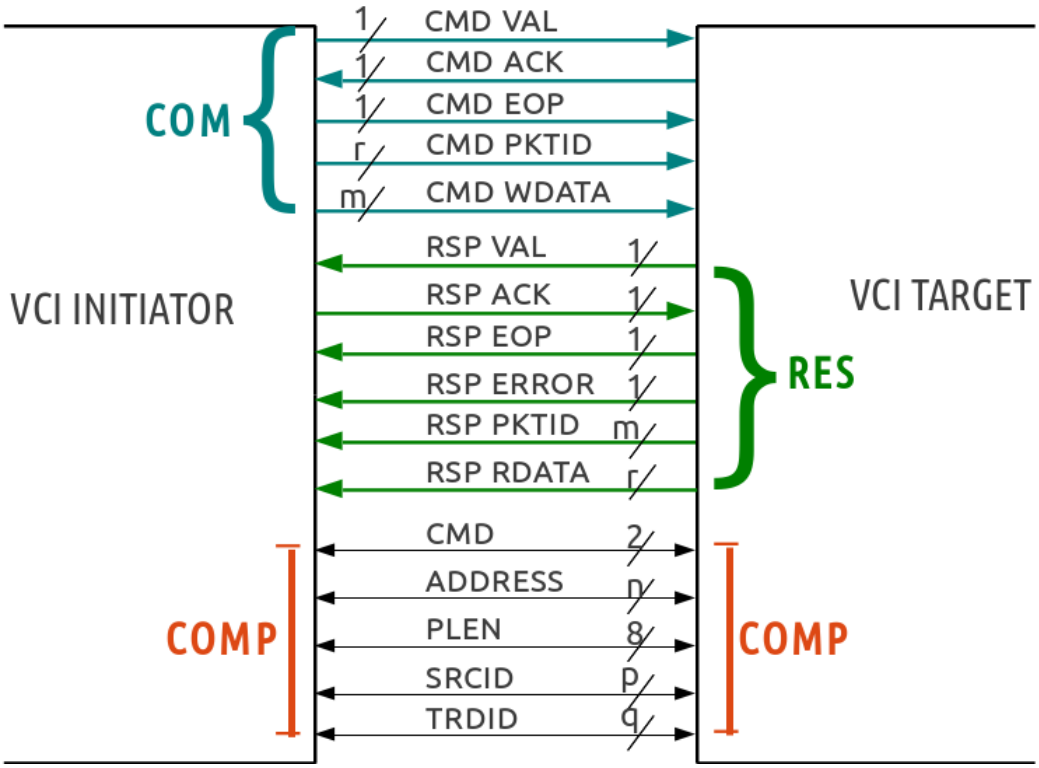


Figura 4.8: Sinais do padrão VCI adotado em Soclib - Anexado de (GREINER; SOCLIB, 2007)

Capítulo 5

Resultados

Neste capítulo são apresentados e discutidos os resultados obtidos como parte experimental da implementação da metodologia descrita em 4.1. Nas primeiras seções, 5.1 e 5.2, são descritas as plataformas de simulação implementadas e os resultados de execução destas comparando-se os tempos obtidos por TLM e CA. Nas seções 5.3 e 5.4 são apresentadas as séries de tráfego obtidas pelo mecanismo de captura e a comparação estatística entre séries TLM e CA, definida em 4.3. Nas seções 5.5 e 5.6, são apresentadas as séries de tráfego estatístico obtidas pelo ajuste direto de modelos sobre as séries e pelo ajuste de modelos sobre as séries decompostas. Os resultados das séries sintéticas das últimas seções foram comparadas com as das séries originais, a partir de parâmetros estatísticos. Todas as simulações e cálculos foram executadas num computador com processador Intel i7 Q720 de 1.60 GHz e 6 GB de memória RAM.

5.1 Aplicações de Teste

Nos experimentos propostos, códigos de duas aplicações foram adotados; para cada uma delas os códigos foram executados para posterior análise do respectivo tráfego capturado. A Figura 5.1 apresenta o diagrama de blocos indicando que os dados para serem processados foram armazenados na memória RAM. Pode-se observar o componente TTY, cuja função é ser um terminal, usado como interface entre o usuário e o processador, para a entrega de mensagens do processador como interrupções, comandos tipo *print*, erros, etc.

Uma das aplicações executadas, disponível em (SIGMALAB, 2012), é denominada de MFT porque é parte de uma implementação de um sistema *Micro Four Thirds*, bastante presente em câmaras fotográficas DSLR (do inglês, *Digital single-lens reflex*). A Figura 5.2 representa o funcionamento da aplicação MFT; dada uma imagem armazenada na memória como uma matriz A, o MFT calcula o valor médio a partir de um pixel em relação aos valores dos pixels circundantes. Na versão desta dissertação, foi adotada uma janela de 3×3 pixels que vai-se trasladando pela matriz A, sendo o pixel original o pixel central. A posição do pixel processado, na matriz B, será a mesma do pixel central e seu valor é o resultado do cálculo MFT. Esta operação produz uma imagem resultante suavizada em relação à original.

A segunda aplicação é uma aplicação nativa do Soclib, o decodificador de vídeo MJPEG (do inglês, *Motion JPEG*). A Fig. 5.3 ilustra o funcionamento do decodificador MJPEG (AUGE et

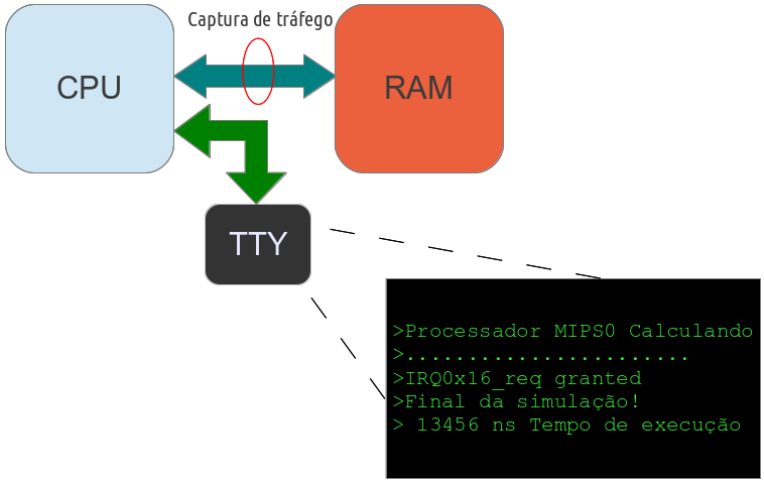


Figura 5.1: Representação em blocos da plataforma com TTY

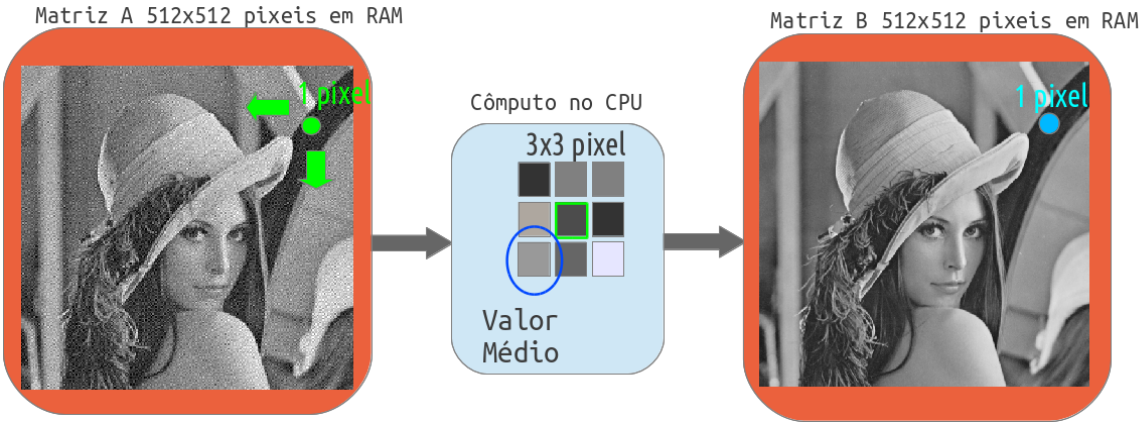


Figura 5.2: Representação em blocos da plataforma com TTY

al., 2005). Neste formato, cada *frame* é uma imagem JPEG comprimida (POYNTON, 2003), que em nosso experimento foi de 256 × 256 pixels. O *stream* (cadeia de dados) ingressa pelo bloco *demux* responsável de passar o fluxo de dados para os blocos seguintes; em seguida,o bloco *VLD* faz uma decodificação de Huffman de longitude variável, os coeficientes são reordenados no bloco *ZZ*, para, então, ingressarem no bloco *IQ* para o processo de quantização inversa. O bloco final é o *IDCT* que faz a transformada inversa discreta do cosseno. O bloco *LIBU* permite passar o resultado para algum periférico ou memória.

5.2 Tempo de Simulação das Plataformas CA e TLM

As aplicações apresentadas na seção anterior foram executadas nas plataformas TLM e CA, na forma descrita na Seção 4.2.1 e 4.5 do capítulo anterior. A aplicação MFT na plataforma CA requereu $2,7 \times 10^6$ ciclos de relógio enquanto que, na plataforma TLM, foram necessários $2,6 \times 10^6$ ciclos referenciais[1] para a execução completa. A aplicação MJPEG, por sua vez,

[1]O *kernel* de simulação do SystemC passa um argumento de relógio à implementação TLM para não perder a sincronização do *scheduler*

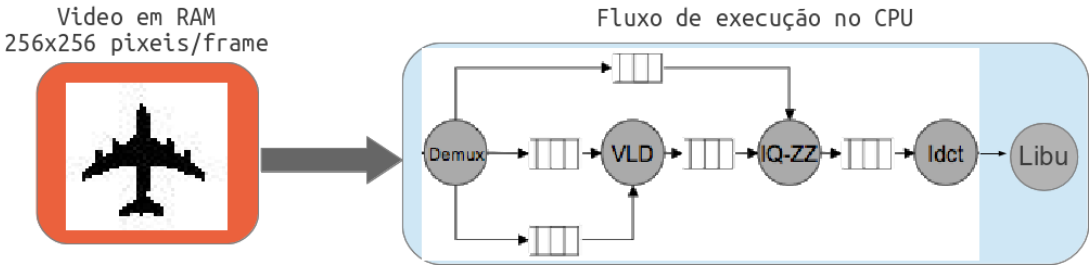


Figura 5.3: Representação em blocos da plataforma com MJPEG

requereu 10×10^6 ciclos de relógio para a sua execução na plataforma CA, e $6,5 \times 10^6$ ciclos referenciais na plataforma TLM para executar o mesmo número de eventos, monitorados pelo componente de visualização TTY das plataformas. A execução do MJPEG foi encerrada ao atingir o processamento do bloco 28 do terceiro *frame*, de um total de 36 blocos por *frame*, tendo sido, então, processados aproximadamente $\approx 2,8$ *frames*.

Os resultados referentes aos tempos de simulação nas plataformas são apresentados na Tabela 5.1. Nas linhas 2 e 3, os tempos de simulação nas plataformas nos níveis TLM e CA são, respectivamente, apresentadas. As colunas 2 e 4 apresentam os resultados de execução direta das aplicações MFT e MJPEG sem captura de rastros, enquanto os resultados das colunas 3 e 5 consideram as aplicações coma inclusão da captura de rastros durante sua simulação. A captura de rastros, correspondente às transferências T_{out}, foi feita no ponto de medição do canal de Comando indicado na Seção 4.2.1 . Desta forma, pedaços extras de código em C++ foram adicionados aos arquivos da descrição em SystemC do barramento para a gravação do fluxo de saída do processador em um arquivo de texto TXT. Na captura da plataforma CA, conforme a Figura 4.7, os valores dos sinais do barramento foram gravados num arquivo VCD (do inglês, *Value Change Dump*) usando comandos em SystemC.

Como esperado, pode-se perceber que a simulação em CA é mais lenta do que o em TLM. A fileira 4 mostra a redução dos tempos de simulação TLM em relação ao modelo CA; paras as aplicações MFT e MJPEG, o tempo de simulação foi, respectivamente, $\times 2,23$ e $\times 6,07$ vezes mais rápido, quando a captura de tráfego é feita. Os tempos na simulação sem captura de rastros são menores porque não incorporam rotinas de escrita em arquivos externos, não alterando ao fluxo de execução do *kernel* SystemC. Por outro lado, o fator de redução é maior, uma vez que não há acréscimo do tempo referente à captura de traces, o qual é aproximadamente igual nos casos CA e TLM.

Como previsto, os tempos de simulação decrescem adotando-se a modelagem das estruturas de comunicação no nível mais abstrato TLM, porém a redução destes tempos poderia ser mais forte se a descrição do processador estivesse no nível TLM ou em alguma outra representação de alto nível, em vez do ISS com *wrappers* presente em Soclib e usado nos nossos experimentos.

5.3 Captura de Tráfego

A simulação permite que se obtenham os valores de cada registrador, no caso TLM, e dos sinais no tempo, no caso CA. Como descrito na seção 4.2.2, o tráfego é obtido a partir do

Tabela 5.1: Tempos de simulação das aplicações MFT e MJPEG

	MFT	MFT Trace	MJPEG	MJPEG Trace
TLM	1466 ms	5807 ms	1598 ms	7116 ms
CA	5428 ms	12969 ms	14516 ms	43225 ms
×fator	×3,70	×2,23	×9,08	×6,07

agrupamento de dados referentes aos registradores ou sinais dentro de um intervalo de tempo, ou seja, um *bin*. Os resultados da simulação são encontrados em arquivos TXT e VCD. Extraímos os dados do T_{out} definidos na Seção 4.2.2 através de um *parser* dos arquivos TXT e VCD, programado no software MATLAB (MATLAB, 2012).

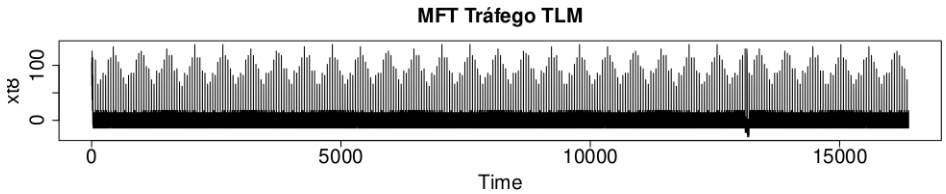
Para obter o tráfego relativo à simulação CA, que denominaremos tráfego CA, definimos janelas de observação de tamanho 100 ciclos dado que 1 ciclo é equivalente a 1 ns como usado em (SCHERRER; FRABOULET; RISSET, 2009a) para ambas aplicações. Mantendo a porcentagem da quantidade de ciclos requerida na simulação, definimos uma janela de tamanho 97 para o tráfego TLM da aplicação MFT e outra, de tamanho 65, para o tráfego TLM da aplicação MJPEG. Obtém-se assim 100.000 amostras para MJPEG e 26.288 para MFT.

As Figuras 5.4(a) e 5.4(b) apresentam as séries tráfego da aplicação MFT no nível TLM com tamanho de $bin = 97$, e CA com tamanho de $bin = 100$, enquanto as Figuras 5.5(a) e 5.5(b) apresentam o tráfego da aplicação MJPEG com tamanho de $bin = 65$ e tamanho de $bin = 100$, respectivamente para os casos TLM e CA. Uma observação visual permite verificar que os tráfegos TLM e CA da aplicação MFT são semelhantes para uma amplitude baixa de aproximadamente 20 bytes/bin. Uma ligeira variação na amplitude presente no tempo aproximado de $t = 12.600$, é manifestada em ambos níveis de abstração. Também é notória uma amplitude mais estável no tráfego CA.

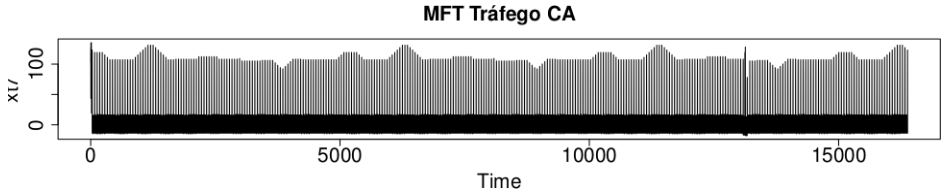
O tráfego da aplicação MFT apresenta uma forte impulsividade, similar em ambos níveis de abstração, mas no caso TLM, a amplitude das amostras de maior valor varia, diferentemente do caso CA que são constantes. No caso da aplicação MJPEG pode-se observar que o intervalo de amplitude é semelhante entre os casos TLM e CA. Para uma melhor visualização dos tráfegos, intervalos de tempo menor são utilizados. A semelhança das séries é evidenciada novamente nas Figuras 5.6(a) e 5.6(b), para a aplicação MFT nos primeiros 256 pontos da série, e nos primeiros 512 pontos do tráfego da aplicação MJPEG, nas Figuras 5.7(a) e 5.7(b). Novamente, pode-se observar que as características de impulsividade da série CA são mantidas no mesmo intervalo na série TLM. Ressalta-se, que o tráfego da aplicação MJPEG apresenta mudanças de regime na amplitude, diferente do comportamento da Figura 3.2(c) apresentada no Capítulo 3, onde é discutida a metodologia de agrupamento por fases.

5.4 Análise de Tráfego

O objetivo principal da análise do tráfego das aplicações MFT e MJPEG é determinar suas características estatísticas. Primeiramente, verificamos se as séries de tráfego TLM e CA apresentam características semelhantes e, depois, os mesmos experimentos são feitos para determinar

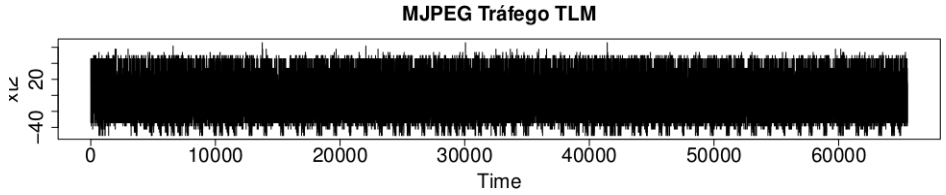


(a)

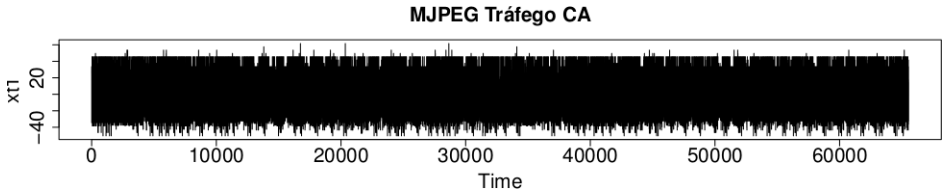


(b)

Figura 5.4: Série de tráfego da aplicação MFT: (a) TLM, $bin = 97$, $T = 16384$. (b) Série de tráfego CA, $bin = 100$, $T = 16384$.



(a)



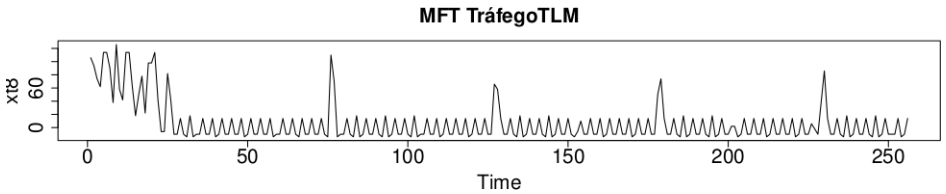
(b)

Figura 5.5: Série de tráfego da aplicação MJPEG: (a) TLM, $bins = 65$, $T = 65536$. (b) CA, $bin = 100$, $T = 65536$.

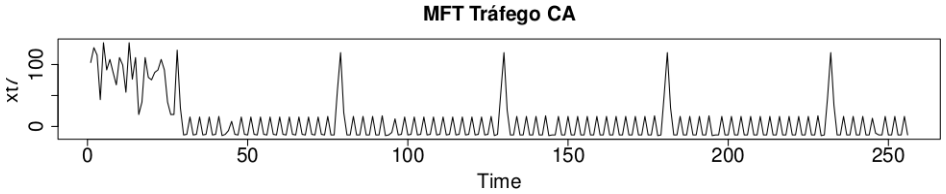
se as características estatísticas se mantém em TLM, caso diferentes taxas de agregação forem usadas. A análise é feita seguindo as técnicas descritas na Seção 4.3.

5.4.1 Comparativo das séries TLM e CA

O conjunto de Figuras 5.8 e 5.9 apresenta os resultados dos cálculos da função de autocorrelação e do periodograma, respectivamente, sobre a série de tráfegos TLM e CA da aplicação MFT. A função de autocorrelação, FAC, da série MFT, na Figura 5.8(a), indica uma baixa autocorrelação entre os elementos, manifestada na presença de múltiplos valores negativos, com uma queda hiperbólica para um $lag = 10$. O periodograma da Figura 5.8(c), com cinco impulsos por cada intervalo de 0,1 da frequência normalizada, mostra a existência de um elemento com periodicidade. Estes resultados indicariam que o tráfego da aplicação MFT possui SRD, pela ausência da singularidade $1/f$, ou seja, sem a tendência ao valor infinito em $freq = 0$ no

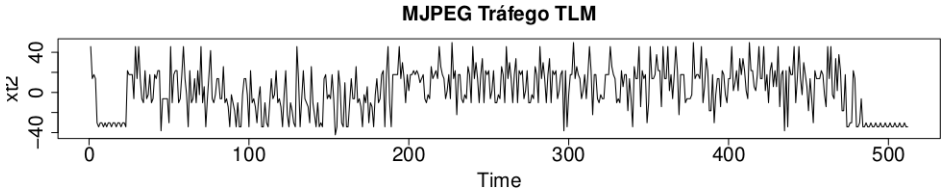


(a)

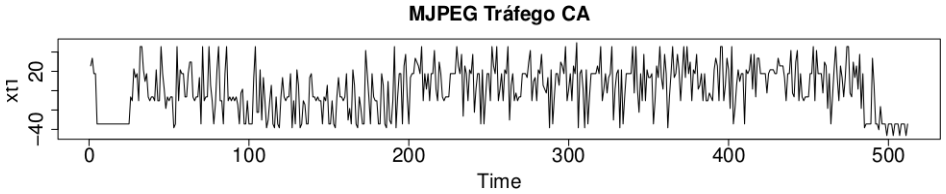


(b)

Figura 5.6: Série de tráfego da aplicação MFT: (a) TLM, $bin = 97$, $T = 256$. (b) Série de tráfego CA, $bin = 100$, $T = 256$.



(a)



(b)

Figura 5.7: Série de tráfego da aplicação MJPEG: (a) TLM, $bin = 65$, $T = 512$. (b) CA, $bin = 100$, $T = 512$.

periodograma, e a baixa autocorrelação da FAC. Comparamos os resultados com os obtidos na análise do tráfego CA, apresentados na segunda coluna da Figura 5.8, indicando semelhança nos testes da autocorrelação e da periodicidade observada no periodograma.

Na primeira coluna da Figura 5.9 são apresentados os resultados da análise estatística sobre o tráfego da aplicação MJPEG. A FAC da série TLM, na Figura 5.9(a), indica alta autocorrelação entre seus elementos, aspecto manifestado numa queda lenta, de comportamento exponencial, para um alto $lag = 30$. O periodograma da Figura 5.9(c) indica uma tendência ao infinito para $freq = 0$; também apresenta uma variação forte, o que indica a existência de um componente periódico. Em conjunto, as curvas indicam que a série possui LRD pela presença da singularidade $1/f$ e alta autocorrelação dos dados. A comparação com os resultados do tráfego MJPEG CA, na segunda coluna da Figura 5.9, mostra a similaridade com as curvas anteriores. A série CA também apresenta alta autocorrelação, com tendência ao infinito em $freq = 0$ no periodograma evidenciando também a presença de LRD.

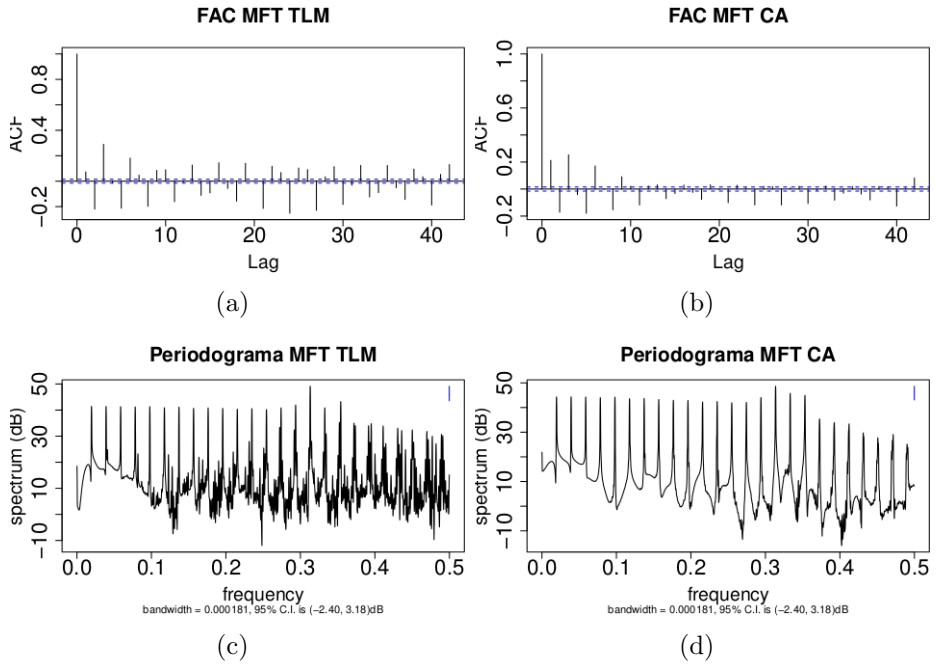


Figura 5.8: Resultados do análise estatístico da aplicação MFT: (a) FAC TLM. (b). FAC CA. (c) Periodograma TLM.(d) Periodograma CA.

5.4.2 Comparativo das séries TLM e CA com teste de agregação

De acordo a teoria da Seção 2.2.3, em uma série temporal com LRD, as características estatísticas típicas devem ser mantidas em todas as escalas de observação. Fizemos um teste para determinar se tais características estatísticas são mantidas no tráfego TLM com diferentes agregações e comparando-as com os resultados do tráfego CA.A primeira agregação foi em 32 pontos e posteriormente sobre esta mesma série realizamos outra agregação em 8 pontos (denominaremos como 8+). As séries agregadas em 32 e 8+, apresentadas nas Figura 5.10 e 5.11 respectivamente, foram preenchidas com zeros (método de *zero-padding*) para obter uma escala de tempo resultado da potência de dois sem alterar os resultados (CASTIGLIONI, 2005). As séries agregadas indicam que o tráfego mantém a impulsividade em diferentes escalas de tempo tanto em tráfego TLM, ilustradas nas Figuras 5.10(a) e 5.11(a), como em tráfego CA, nas Figuras 5.10(b) e 5.11(b).

Os resultados da avaliação estatística sobre a série de tráfego TLM da aplicação MJPEG agregada em 32 pontos são apresentados na primeira coluna da Figura 5.12. Estes indicam a presença de LRD pela alta autocorrelação para um $lag = 35$, na Figura 5.12(a), e a tendência a infinito em $freq = 0$, Figura 5.12(c). Resultados similares foram obtidos com o tráfego CA agregado em 32, apresentados nas Figuras 5.12(b) e 5.12(d). Os resultados estatísticos, depois de uma agregação em 8+, são apresentados na Figura 5.13 e são similares aos obtidos na primeira agregação de 32 apresentados na Figura 5.12.

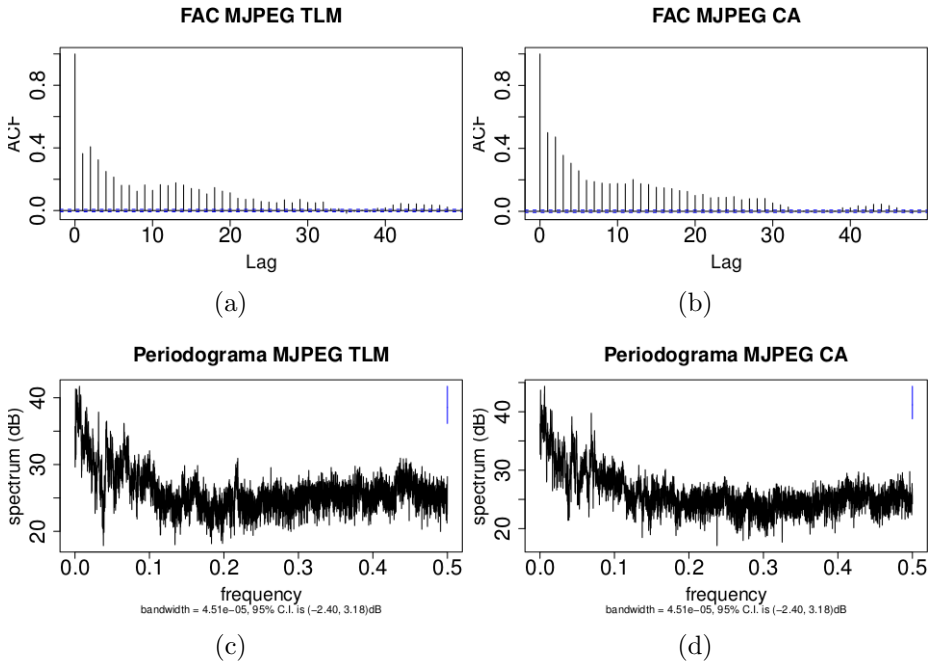


Figura 5.9: Resultados do análise estatístico da aplicação MJPEG: (a) FAC TLM. (b) FAC CA. (c) Periodograma TLM. (d) Periodograma CA.

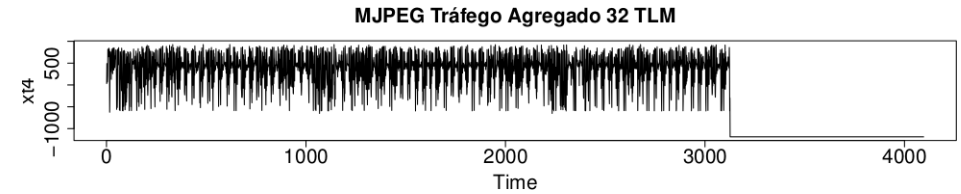
5.4.3 Obtenção do parâmetro H

De acordo com a metodologia descrita na Seção 4.4, fizemos a estimativa do parâmetro H sobre as séries. Os diferentes métodos para a obtenção de H, apresentados anteriormente, foram utilizados inicialmente para o tráfego da aplicação MFT e os resultados, apresentados na Tabela 5.2, mostram incongruência entre os valores obtidos. O valor de H obtido sobre o tráfego TLM tem notórias variações; pelo método wavelet o resultado indica a presença de SRD com $H = 0.151$, porém em todos os outros métodos, há indicação de existência de LRD, com $H > 0.5$. Em relação às estimativas de H no tráfego CA, os resultados são semelhantes, mas estes apresentam inconsistências também pelo fato de apresentar $H > 1$ pelos métodos periodograma e R/S. Os resultados de FAC e periodograma, observados e analisados na seção anterior, indicam a presença de SRD na série, e como foi esperado, os métodos de Wavelet e Whittle foram acertados na determinação do parâmetro H.

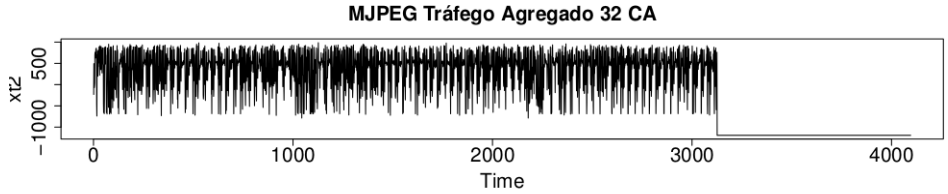
Tabela 5.2: Estimativa do parâmetro H no tráfego da aplicação MFT

Método	TLM	CA
Periodograma	0.860	1.078
R/S	0.992	1.064
Wavelet	0.151	0.214
Whittle	0.512	0.618

Os resultados da estimativa do parâmetro H sobre o tráfego da aplicação MJPEG são apresentados na Tabela 5.3. Os valores de H da série original TLM, na segunda coluna, são próximos e indicam a presença de LRD, assim como os resultados de H nas séries agregadas. Isto é obser-

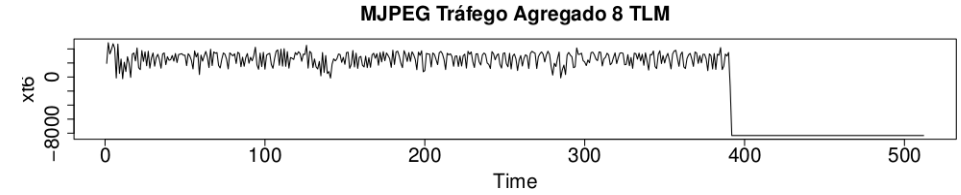


(a)

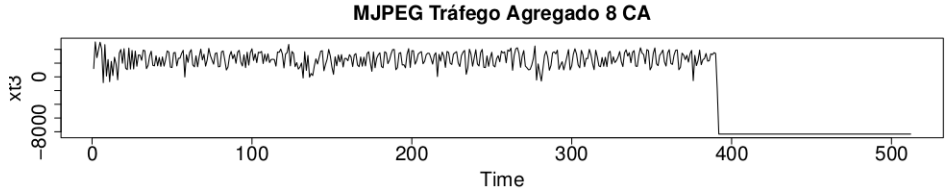


(b)

Figura 5.10: Série de tráfego da aplicação MJPEG agregada em 32: (a) TLM, $bins = 65$, $T = 4096$. (b) CA, $bins = 100$, $T = 4096$.



(a)



(b)

Figura 5.11: Série de tráfego da aplicação MJPEG agregada em 8: (a) TLM, $bin = 65$, $T = 512$. (b) CA, $bin = 100$, $T = 512$.

vável nas colunas quatro e seis, mas existe uma inconsistência com $H > 1$ no resultado com o método do periodograma. Também fica evidenciado um decréscimo no valor de H com o método R/S sendo mais evidente quando as séries são agregadas. Em relação as séries CA, das colunas três, cinco e sete, os resultados são semelhantes. Dados seus resultados na análise da FAC e do periodograma, que indicaram a presença LRD, esta comparação evidencia as deficiências dos métodos de estimação de H por R/S e periodograma, e a robustez dos método Wavelet e Whittle.

5.4.4 Identificação de picos no espectro Wavelet

As inconsistências evidenciadas nos resultados do parâmetro H apresentados na Tab. 5.3 da seção anterior, estão associadas aos picos gerados nos periodogramas, devidos à presença

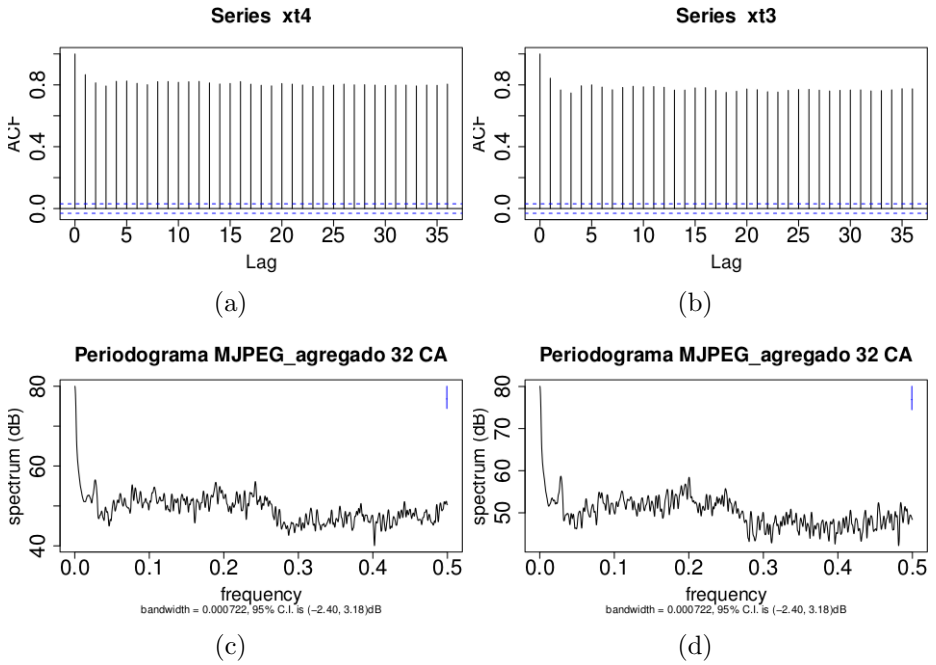


Figura 5.12: Resultados da análise estatística da aplicação MJPEG agregada em 32 pontos: (a) FAC TLM. (b) FAC CA. (c) Periodograma TLM. (d) Periodograma CA.

Tabela 5.3: Estimativa do parâmetro H no tráfego da aplicação MJPEG

	MJPEG Original		MJPEG Agregado em 32		MJPEG Agregado em 8	
	TLM	CA	TLM	CA	TLM	CA
Periodogram	0,821	0,832	0,977	0,990	1,532	1,513
R / S	0,720	0,698	0,674	0,701	0,531	0,575
Wavelet	0,868	0,882	0,716	0,706	0,652	0,563
Whittle	0,810	0,878	0,827	0,814	0,699	0,684

de elementos periódicos. De trabalhos anteriores, sabe-se que que o método wavelet de Abry-Veitch é um dos mais robustos e confiáveis (SHENG; CHEN; QIU, 2011), sendo a estimação do H influenciada pelos picos na medição do gradiente a partir do espectro. Usamos todas as escalas j para observar o nível dos picos, mas é importante ter em consideração que o valor de H mudaria em relação a escala j inicial e final em que seja medido, como foi explorado em (AUGUSTO, 2009). Na Figura 5.14, os espectros wavelet das séries da aplicação MFT são apresentados; em 5.14(a) observa-se o espectro da série TLM com presença de picos nas escalas $j = 2, 6, 8$, apresentando o maior pico na escala wavelet $j = 6$. Tal resultado é semelhante ao da Fig. 5.14(b) com nível aproximado de $2, 8$ no diagrama logarítmico.

Os espectros wavelet observados das séries TLM e CA da aplicação MJPEG indicam picos em $j = 8, 10, 12$ e $j = 7, 10, 12$ respectivamente, mostrados na Fig. 5.15(a) com nível aproximado a 3,8 para o pico da região central. Considerando os resultados da FAC e o periodograma no teste de agregação (que evidenciam a natureza fractal das séries MJPEG, indicando LRD), os resultados dos espectros wavelet, apresentados nas Figura 5.16(c), mostram picos em $j = 5, 7,$

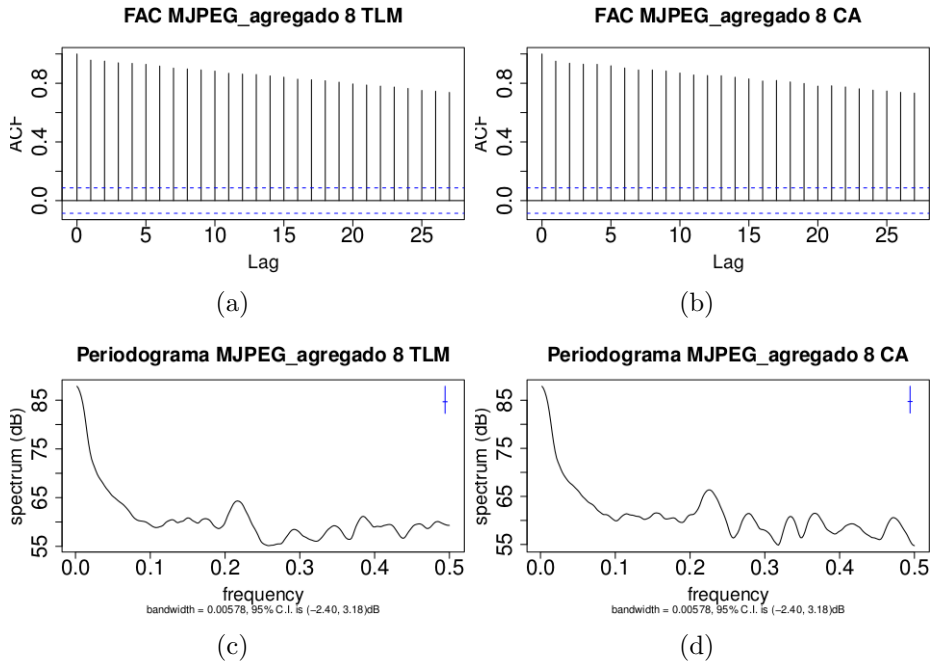


Figura 5.13: Resultados da análise estatística da aplicação MJPEG agregada em 8 pontos depois da primeira agregação de 32: (a) FAC TLM. (b) FAC CA. (c) Periodograma TLM. (d) Periodograma CA.

que foram desfasados pela mudança da escala, indicando assim semelhança entre eles. Considere-se que a constante no final entre $7 < j < 10$, como aparece nas Fig. 5.16(a) e 5.16(c) das séries MJPEG TLM, é produzida pela adição de zeros no final da série, assim como ocorre nas Fig.5.16(b) e 5.16(d).

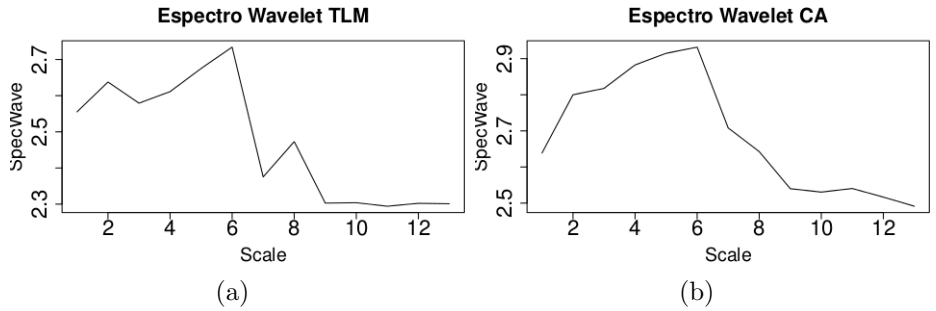
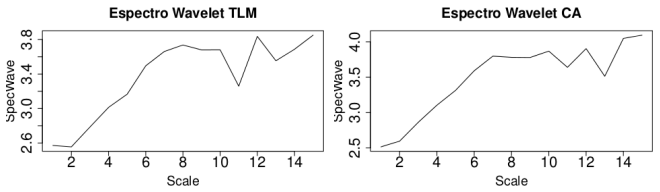


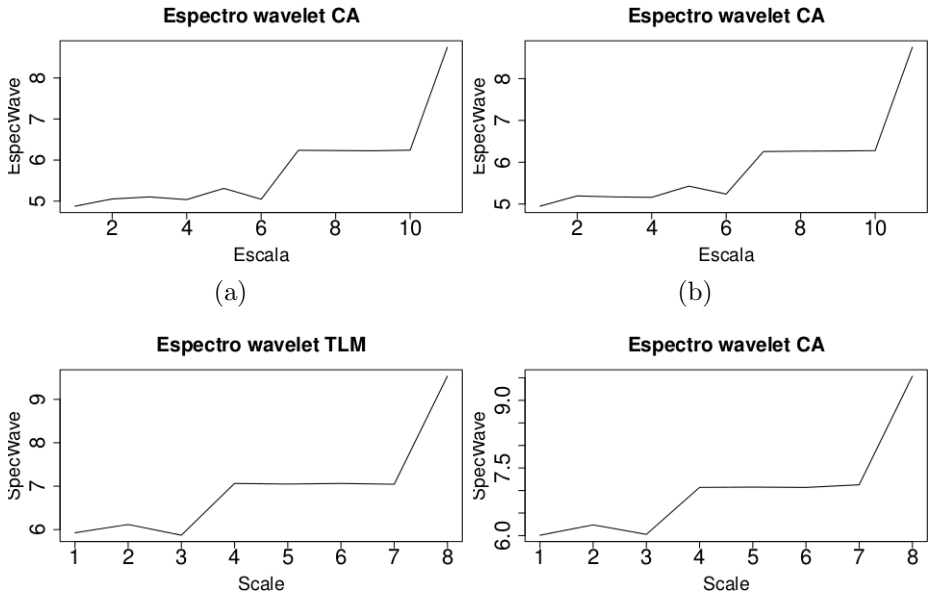
Figura 5.14: Espectro Wavelet da aplicação MFT: (a) TLM. (b) CA.

A presença dos picos em faixas específicas de frequência, associadas aos períodos detectados na análise do espectro wavelet do tráfego, indicam a presença conjunta de componentes SRD que incidem no resultado de H (AUGUSTO, 2009). As Figuras 5.17(a) e 5.17(b), foram obtidas na implementação do algoritmo do estimador wavelet (VEITCH, 2012) em MATLAB e apresentam os resultados do espectro wavelet do tráfego TLM da aplicação MFT e MJPEG respectivamente. Elas indicam que existe uma mudança no gradiente na estimativa do H considerando-se as escalas de inicio e fim $j = [1 - 12]$, que pode ser considerado uma sobre ou subestimação. Isto porque tais pontos de referência nas escala j podem variar, modificando o resultado em relação



(a) (b)

Figura 5.15: Resultados da análise estatística da aplicação MJPEG: (a) Espectro Wavelet TLM. (b) Espectro Wavelet CA.



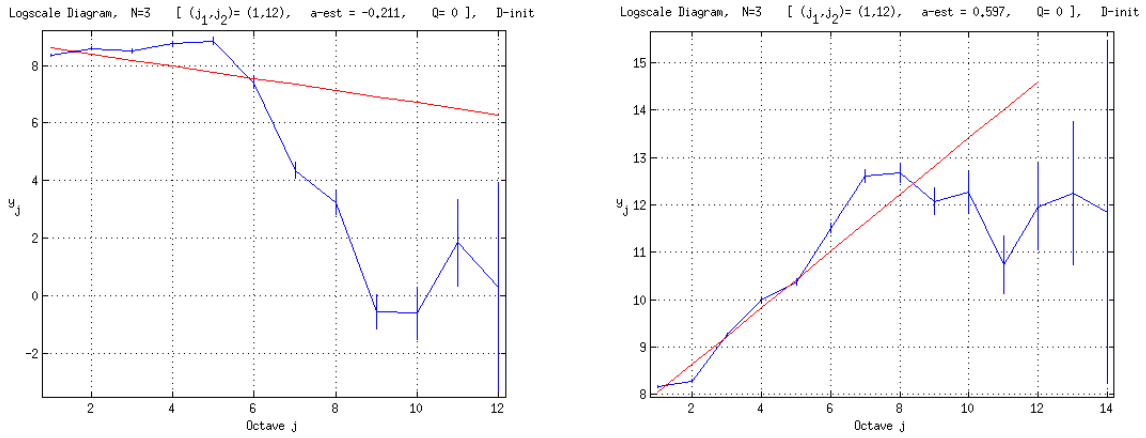
(a) (b)

(c) (d)

Figura 5.16: Resultados da análise estatística da aplicação MJPEG agregada: (a) Espectro Wavelet TLM com agregação de 32 pontos. (b) Espectro Wavelet CA com agregação de 32 pontos. (d) Espectro Wavelet CA com agregação de 8 pontos depois da primeira agregação de 32 pontos. (d) Espectro Wavelet CA com agregação de 8 pontos depois da primeira agregação de 32 pontos.

à amplitude do espectro. Observamos que os picos em $j = 7, 10, 13$ são semelhantes aos picos presentes na Figura 3.3(c) do trabalho de (SCHERRER; FRABOULET; RISSET, 2009a). Estes picos podem ser gerados pelos elementos periódicos das séries que adicionam um componente SRD numa faixa de frequência específica, fazendo com que os resultados do H da série MFT sejam subestimados e com outros métodos, como periodograma e R/S, sejam sobrestimados. Já no caso da série MJPEG, apesar de existir semelhança nos resultados de H, seu valor pode estar sobrestimado.

A partir dos resultados anteriores, identificamos que as características estatísticas do tráfego das aplicações MFT e MJPEG são mantidas no nível TLM. Os resultados de FAC, Periodograma e espectro wavelet apresentaram características semelhantes, sendo o mesmo comportamento, SRD ou LRD, evidenciado em ambos os níveis de abstração das séries, TLM e CA. Consideramos que as séries TLM são apropriadas para modelar as aplicações que representam, sendo então



(a) (b)

Figura 5.17: Espectro wavelet do tráfego TLM obtidos com o algoritmo (VEITCH, 2012): (a) MFT. (b) MJPEG.

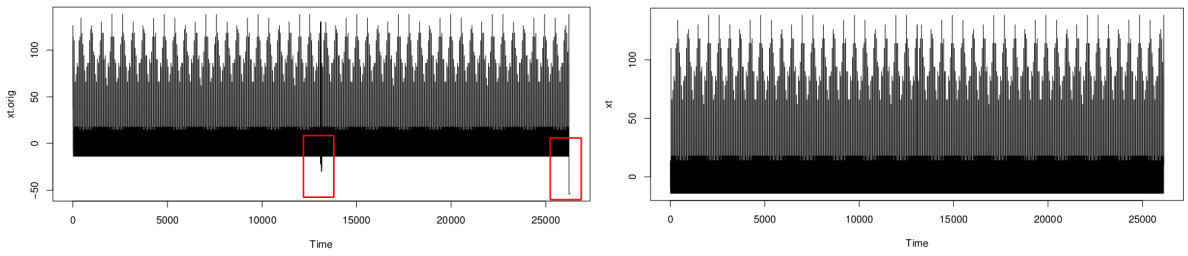
usadas para o ajuste de séries sintéticas, como explicado a seguir.

5.5 Ajuste da séries de tráfego MFT

Nesta seção realizamos o ajuste de parâmetros de séries sintéticas à série original do tráfego MFT com o objetivo de avaliar a qualidade resultante na aplicação das técnicas descritas na seção 4.4.

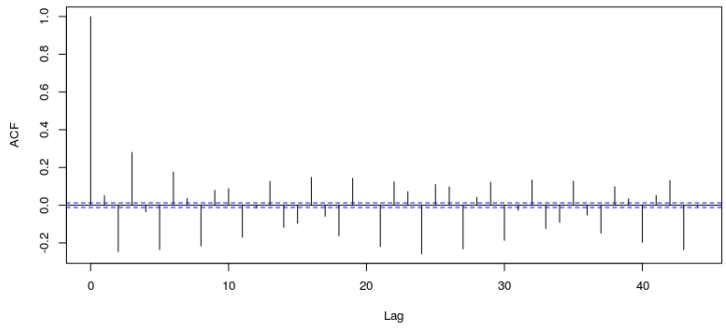
5.5.1 Pré-processamento das Séries para Ajuste

O tráfego inicialmente obtido pela simulação TLM da aplicação MFT sofreu alterações a partir de uma inspeção visual de existência de ruídos. Foram removidos os primeiros 40 dados e outros 122 indicados na Fig. 5.18 da série original que consideramos como transitórios, dada a diferença na amplitude destes dados com os outros que estavam ao seu redor. Os resultados da FAC e periodograma do novo tráfego, nas Figs. 5.19(a) e 5.19(b), são semelhantes aos estimados anteriormente, com baixa autocorrelação dos dados e presença de periodicidade.

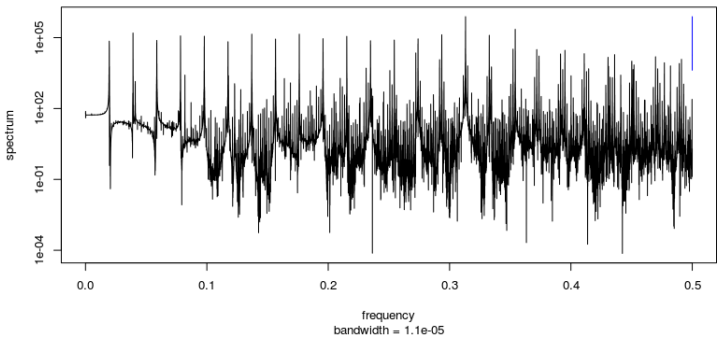


(a) (b)

Figura 5.18: Séries TLM MFT: (a) MFT inicial. (b) MFT sem transitórios.



(a)



(b)

Figura 5.19: Série TLM MFT sem transitórios: (a) FAC. (b) Periodograma.

Por outro lado, como resultante desta remoção, na avaliação do espectro wavelet, os picos foram suavizados, como apresentado na linha vermelha na Fig. 5.20. Pode-se observar as diferenças com a estimação inicial, mostrada pela linha azul, identificando-se o pico maior na escala $j = 5$ com amplitude de $2,6$, e uma queda maior entre as amplitudes 2 e $1,9$ a partir da escala $j = 9$. Portanto, ao nos referirmos à série TLM MFT, original será o caso sem transitórios.

5.5.2 Ajuste sem decomposição

O primeiro método para a geração da série sintética é pelo ajuste direto, ou seja, diretamente sobre a série MFT. Este caso diferencia-se do método realizado com curvas de tráfego decompostas como introduzido na Seção 2.3, sendo então o tráfego denominado como 'sem decomposição'. O ajuste foi feito com o comando *auto.arima* do pacote em R *forecast* (HYNDMAN, 2012), estimando-se automaticamente os coeficientes para um modelo ARMA(5,4). Na Fig. 5.21 são apresentados os resultados da série sintética sem decomposição gerada pelo comando *simulate.arima* do pacote *forecast* gerando a quantidade de amostras $T = 16,384$. Na Fig. 5.21(a) são apresentados os primeiros 512 pontos da série sintética para ilustração. Nas Figuras 5.21(b) e 5.21(c) são apresentados os resultados da FAC e do periodograma deste tráfego, respectivamente. A FAC apresenta semelhança com a série original MFT, quanto à autocorrelação dos dados, com queda para $lag = 10$. Já o periodograma do modelo ARMA(5,4) apresenta diferença com o obtido a partir da série original, ficando ressaltada a presença de um período na frequência normalizada $freq = 0,32$.

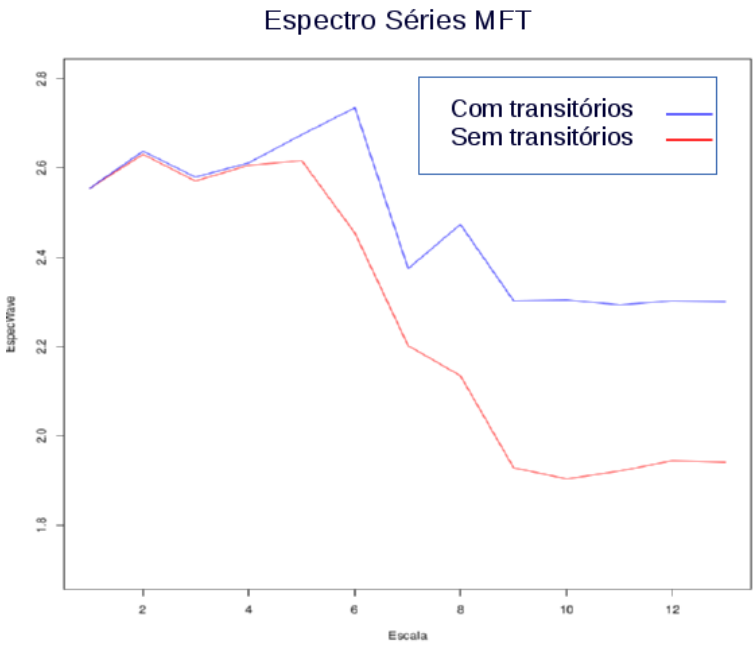


Figura 5.20: Comparação do espectro wavelet da série TLM MFT

A Fig. 5.22 apresenta os resultados da avaliação dos resíduos do modelo ARMA(5,4). As Figuras 5.22(a) e 5.22(b), do FAC e periodograma indicam, respectivamente, uma baixa autocorrelação e uma resposta sem uma tendência significativa, semelhantes aos resultados de ruído gaussiano ideais das Figuras 4.6(a) e 4.6(b), indicando que o modelo pode ser usado para representar a série original. Porém, o histograma e o *qqplot*, das Fig. 5.22(c) e 5.22(d), indicam que a distribuição não é totalmente normal, já que apresenta uma cauda notória no extremo superior do *qqplot*.

5.5.3 Ajuste com decomposição

O segundo método de ajuste a partir da decomposição da curva de tráfego em curvas de Tendência, Periódico e Resto, como indicado na Seção 2.3, foi realizado para comparação. Fizemos a decomposição aditiva da série com o período de 1/32 obtido com a escala $j = 5$ do espectro wavelet da Fig. 5.20. Posteriormente, avaliamos estatisticamente as partes de maneira separada, para, finalmente, ajustar um modelo para cada uma delas. Foram adotados os modelos *ARMA(4,5)*, *ARMA(4,4)* e *ARMA(4,3)*, para Tendência T_t, Periódico S_t e Resto R_t, respectivamente. Os coeficientes dos modelos foram estimados pelo comando *auto.arima* do pacote *forecast*. As séries sintéticas com $T = 16,384$ foram geradas pelo comando *simulate.arima* do pacote *forecast* e as denominamos T'_t, S'_t e R'_t.

Na Fig. 5.23 são apresentados os espectros wavelet das séries sintéticas T'_t, S'_t e R'_t, obtidas de acordo com a modelagem definida previamente, em uma comparação direta com os espectros wavelet dos elementos da decomposição sobre a série original T_t, S_t e R_t. Os resultados do espectro wavelet da Fig. 5.23(b), mostrados pela linha preta, indicam que T'_t não consegue manter o nível de energia a partir da escala $j = 6$, perdendo a representação do elemento periódico original, identificado na linha vermelha. O espectro wavelet na Fig. 5.23(a) de S'_t (linha

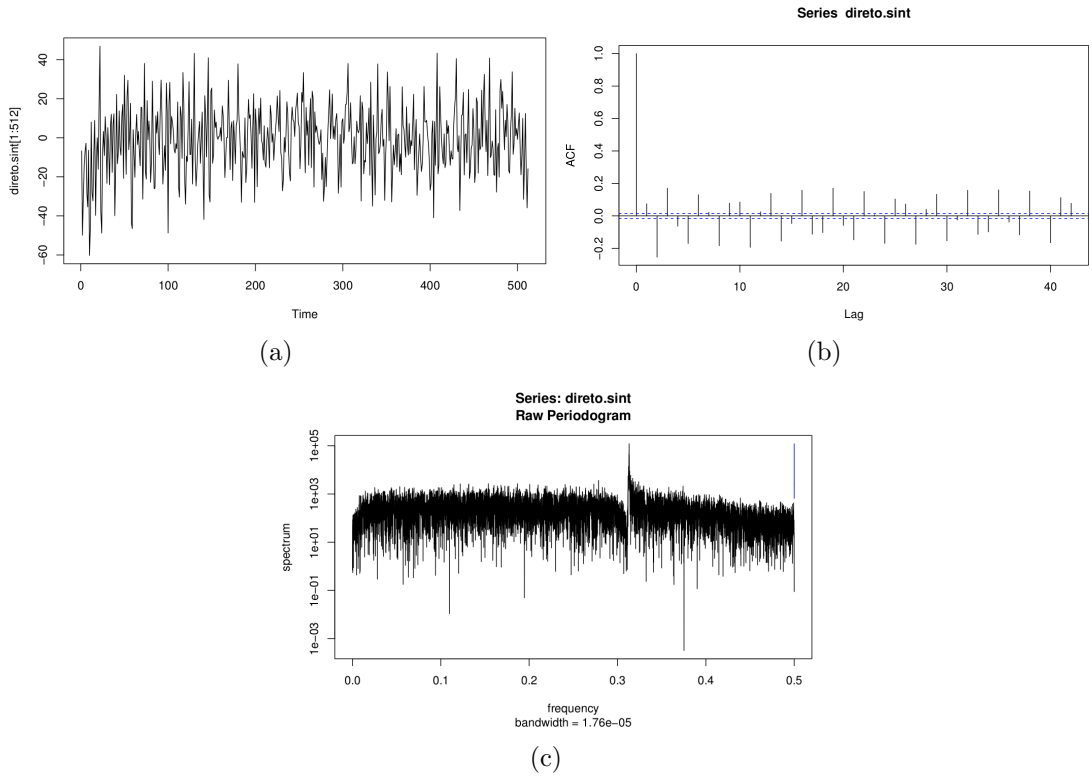


Figura 5.21: Série sintetica TLM MFT por ajuste direto sem transitórios: (a) MFT direto $T = 512$. (b) FAC. (c) Periodograma.

preta) indica uma subestimação na faixa $j = 4$ em comparação ao original (linha vermelha) e uma queda nas faixas seguintes, a partir de $j = 5$. Os resultados do espectro wavelet de R'_t da Fig. 5.23(c) indicam a perda de energia a partir de $j = 10$.

A distribuição dos resíduos de T'_t da Fig. 5.25, apresenta uma distribuição com deformidades em ambos extremos do *qqplot* e não uniforme no histograma, diferentemente dos resíduos do caso ideal da Fig. 4.6, observe-se que na FAC da Fig. 5.24(a) ainda se verifica uma leve autocorrelação. A análise de resíduos de S'_t, apresentada na Fig. 5.24 indica que a distribuição dos resíduos pode se considerar normal como é observado no *qqplot* e histograma. O periodograma dos resíduos da Fig. 5.25(b) indica periodicidade mas a resposta apresenta um comportamento estável, semelhante ao caso ideal da Fig. 4.6. Os resíduos também apresentam uma notória autocorrelação, como observado na Fig. 5.25(a). Os resíduos de R'_t, na Fig. 5.26, apresentam uma distribuição com tendência a ser normal, mas com uma deformidade no extremo superior do *qqplot*; a resposta do periodograma também é estável e os resíduos apresentam uma baixa autocorrelação entre eles.

Para a obtenção da série sintética do tráfego, as séries sintéticas, T'_t, S'_t e R'_t,foram somadas entre si, segundo as definições teóricas da Seção 2.3. Na estimação da FAC do tráfego resultante, da Fig. 5.27(a), evidenciamos uma menor autocorrelação dos dados com uma queda para o $lag = 20$, diferentemente da FAC original da Fig. 5.19(a), o qual tem autocorrelação negativa até $lag = 50$. O periodograma da Fig. 5.27(b) apresenta um forte componente periódico na frequência normalizada $f = 0,32$, o qual não é visível no periodograma da série original da Fig.

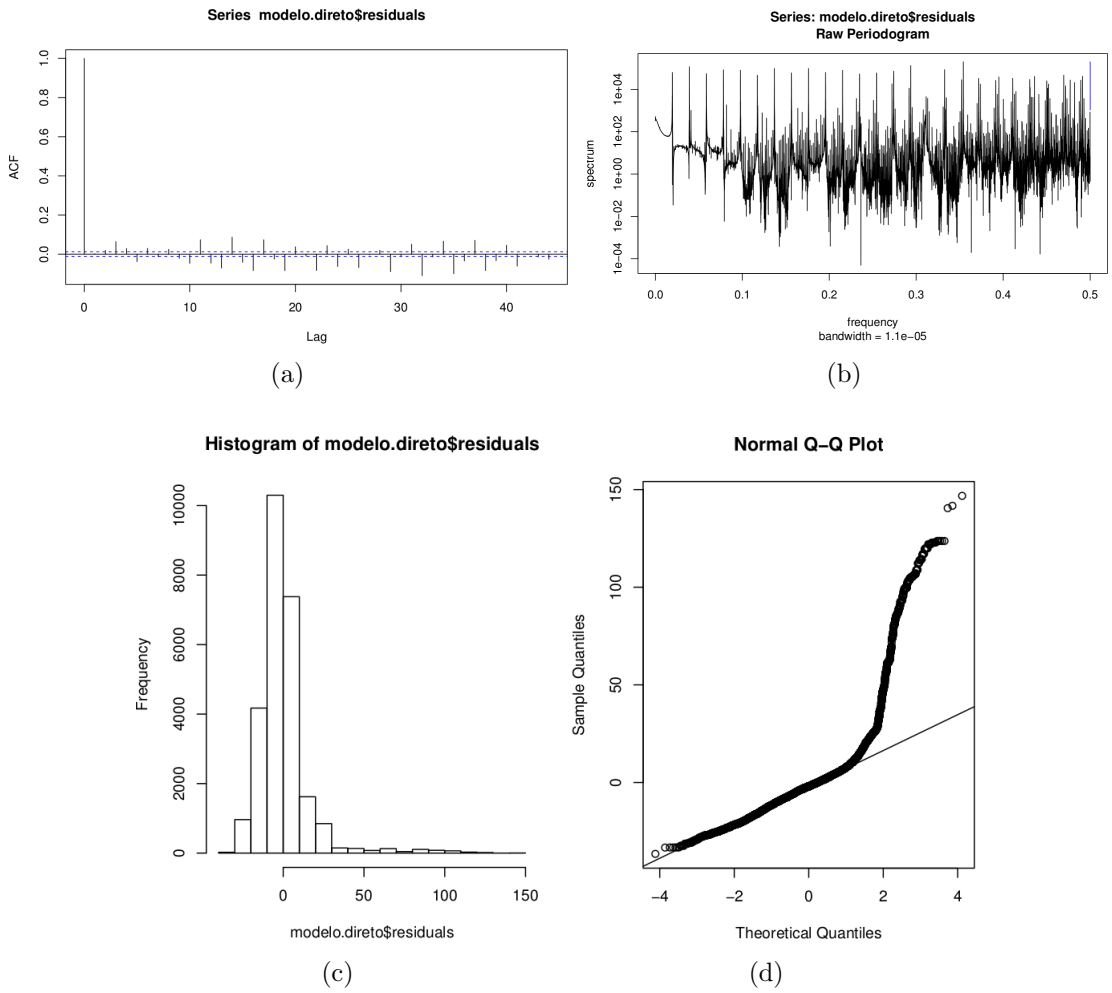


Figura 5.22: Resíduos da série sintética TLM MFT por ajuste direto. (a) FAC: (b) Periodograma. (c) Histograma. (d) *QQplot*.

5.19(b), sendo inclusive maior que o obtido na série por ajuste direto.

5.5.4 Discussão

Para contrastar as séries, calculamos o espectro wavelet apresentado na Fig. 5.28. A figura indica que a série sintética sem decomposição, representada na linha preta, está mais próxima do original, linha vermelha, em comparação a série sintética por decomposição representada na linha azul. Pode-se observar que no caso sem decomposição os espectros wavelet, indicaram maior quantidade de energia nas escalas $j = 2, 5$, semelhante ao espectro da série original; por outro lado, é diferente do caso com decomposição, que apresenta uma queda na faixa de $j = 5$ e um incremento na faixa de $j = 8 - 11$.

Os resultados anteriores indicam que o ajuste direto apresenta resultados com maior semelhança aos resultados originais, devido à semelhança na autocorrelação dos dados e a impulsividade do periodograma na frequência, porém apresenta periodicidade na frequência normalizada do periodograma para $f = 0,31$, é menor em relação ao ajuste com decomposição. Apesar de as rotinas de código da linguagem R ajustarem os modelos anteriores com os modelos mais

apropriados, a análise de resíduos indicam que os modelos tanto do ajuste sem decomposição como dos elementos da decomposição necessitam ser reajustados para uma representação mais fiel dos componentes da série original.

5.6 Ajuste das séries de tráfego MJPEG

Nesta seção realizamos o ajuste de parâmetros de séries sintéticas à série original do tráfego MJPEG com o objetivo de avaliar a qualidade resultante na aplicação das técnicas descritas na seção 4.4.

5.6.1 Pré-processamento das Séries para Ajuste

Diferentemente da aplicação MFT, para a série MJPEG, dada a grande quantidade de dados e a dificuldade para se determinar transitórios por simples inspeção visual, decidimos ajustar a série obtida diretamente da captura da plataforma MJPEG.

5.6.2 Sem decomposição

O primeiro método para a geração da série sintética é pelo ajuste direto, ou seja, diretamente sobre a série MJPEG original, tendo sido usado o comando $fracdiff$ do pacote *fracdiff* (MAECHLER, 2012) e $arfima$ do pacote *forecast*, para ajustar manualmente os coeficientes de um modelo *ARFIMA(4,0.2865943,5)*. O ajuste consistiu em estimar o valor máximo do parâmetro d com um modelo ARFIMA(0,d,0) com *fracdiff*, para posteriormente usá-lo como alcance máximo para um modelo ARFIMA, com número de coeficentes diferentes de zero determinado automaticamente pelo comando *arfima*. Na Fig. 5.29 são apresentados os resultados da série sintética sem decomposição gerada pelo comando *fracdiff.sim* para $T = 65536$. Na Fig. 5.29(a) são apresentados os primeiros 512 pontos da série sintética para ilustração. Nas Figuras 5.29(b) e 5.29(c) são apresentados os resultados da FAC e do periodograma do tráfego sintético, respectivamente. A FAC apresenta grande autocorrelação dos dados, com queda para $lag = 40$, e o periodograma manifesta periodicidade e apresenta a singularidade de $1/f$. Estes resultados sugerem a presença de LRD e são semelhantes aos obtidos com a série original.

A Fig. 5.30 apresenta os resultados da avaliação dos resíduos do modelo ARFIMA(4,0,2865943,5). As Figuras 5.30(a) e 5.30(b), do FAC e periodograma indicam, respectivamente, uma baixa autocorrelação e uma resposta sem uma tendência significativa, semelhantes aos resultados de ruído gaussiano ideais das Figuras 4.6(a) e 4.6(b). Tal situação indica que o modelo pode ser usado para representar a série MJPEG original. O histograma e o *qqplot*, das Fig. 5.30(c) e 5.30(d), indicam que a distribuição é normal já que apresenta um comportamento semelhante às distribuições dos casos ideais 4.6.

5.6.3 Com decomposição

O segundo método de ajuste a partir da decomposição da curva de tráfego em curvas de Tendência, Periódico e Resto, como indicado na Seção 2.3, também foi realizado para a aplicação

MJPEG. Definimos a série com $freq = 256$ em relação à escala $j = 8$ obtida do espectro wavelet para inseri-la no comando *decompose* da linguagem R. Foi realizada a decomposição aditiva da série para posteriormente avaliarmos estatisticamente as partes e finalmente, ajustar um modelo para cada uma delas sendo *ARFIMA(2,0.49,0)*, *ARMA(5,2)* e *ARMA(5,3)* para T_t, S_t e R_t respectivamente. Os coeficentes do modelo para T'_t foram obtidos usando um ajuste manual pelo comando *fracdiff*, e um ajuste automático com *auto.arima* para o caso de S'_t e R'_t. As séries sintéticas com $T = 65536$ foram geradas pelo comando *fracdiff.sim*, do pacote *fracdiff* e *simulate.arima*, do pacote *forecast*, para o caso manual e automático, respectivamente.

Na Fig. 5.31 são apresentados os espectros wavelet das séries sintéticos T'_t, S'_t e R'_t (linhas pretas), comparados aos espectros wavelet dos elementos da decomposição sobre a série original T_t, S_t e R_t (linhas vermelhas). Os resultados do espectro da Fig. 5.31(b) wavelet indicam que T'_t é muito parecido ao espectro do componente original T_t e possui uma ligeira sobrestimação em $j = 10$. O espectro wavelet na Fig. 5.31(a) de S'_t indica uma subestimação, principalmente na faixa $j = 4$ em comparação ao original. Os resultados do espectro wavelet de R'_t da Fig. 5.31(c) indicam a perda de energia em $j = 7$ e não apresenta a forma de campânula, como observado na linha vermelha.

A distribuição dos resíduos de T'_t da Fig.5.33, apresenta uma distribuição uniforme no histograma, semelhante aos resíduos do caso ideal da Fig. 4.6, observando-se que na FAC da Fig. 5.32(a) há uma baixa autocorrelação. A análise de resíduos de S'_t, apresentada na Fig. 5.32, indica que a distribuição dos resíduos pode ser considerado normal como é observado no *qqplot* e histograma. O periodograma dos resíduos da Fig. 5.33(b) indica um comportamento estável, semelhante ao caso ideal da Fig. 4.6. Também, os resíduos não apresentam uma notória autocorrelação como observado na Fig. 5.33(a). Os resíduos de R'_t na Fig. 5.34 apresentam uma distribuição com tendência a ser normal no *qqplot*; também, a resposta do periodograma é estável e os resíduos apresentam uma baixa autocorrelação.

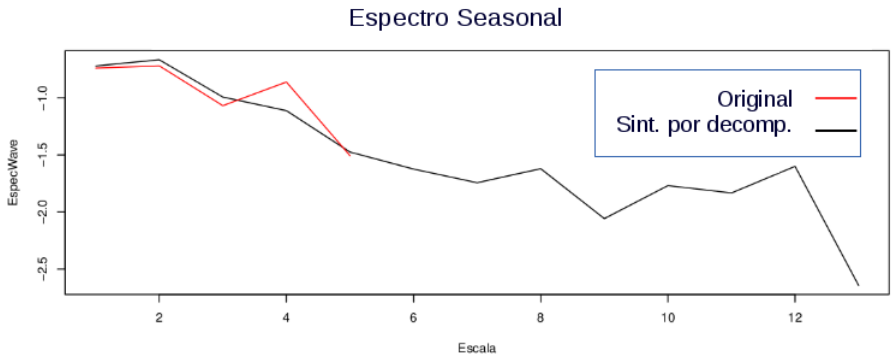
As séries sintéticas foram somadas, segundo as definições teóricas da Seção 2.3. Na estimação da FAC do tráfego somado, da Fig. 5.35(a), evidenciamos menor autocorrelação dos dados, com queda para o $lag = 20$ diferentemente da FAC original da Fig. 5.9(a), o qual apresenta autocorrelação até $lag = 30$. O periodograma da Fig. 5.35(b) apresenta uma ligeira tendência em $f = 0$ pela singularidade $1/f$.

5.6.4 Discussão

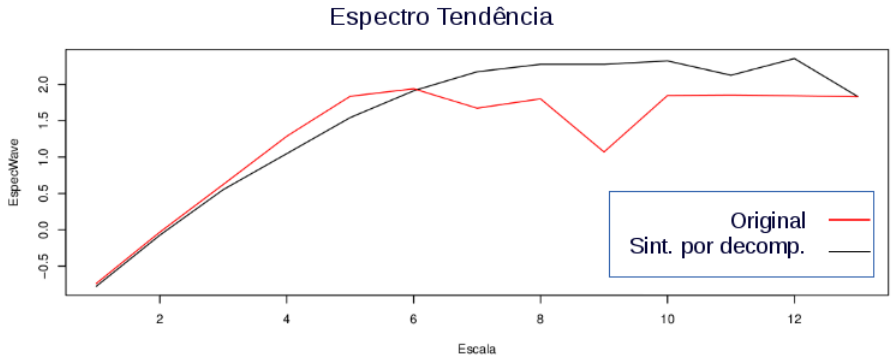
Para contrastar as séries obtidas em métodos diferentes, calculamos o espectro wavelet apresentado na Fig. 5.36; as séries sintéticas, com decomposição e sem decomposição são representadas pelas linhas vermelha e azul respectivamente. A figura indica que a série sintética com decomposição está mais próxima do original, representada na linha preta, em comparação a série sintética sem decomposição. Pode-se observar que no caso com decomposição, os espectros wavelet, indicaram maior quantidade de energia nas escalas $j = 2$ no intervalo $j = 6 - 10$, semelhante ao espectro da série original, porém, diferente do caso sem decomposição que apresenta uma queda na faixa de $j = 2$ e um incremento na faixa de $j = 12$.

Os resultados anteriores indicam que o ajuste com decomposição tem maior semelhança aos resultados originais pela semelhança na autocorrelação dos dados e no espectro wavelet.

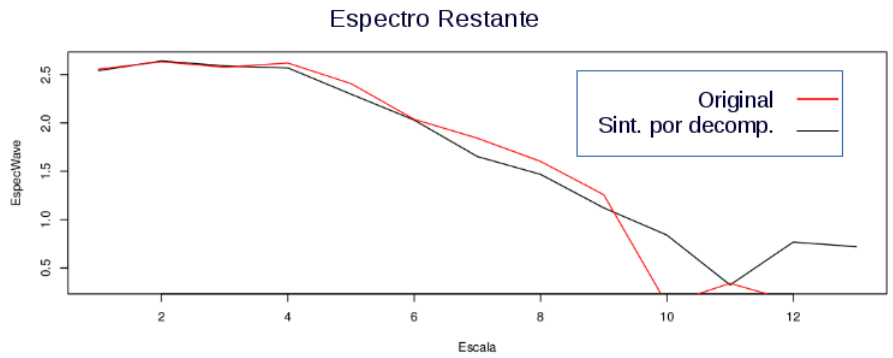
Por outro lado, apresenta baixa impulsividade na frequência normalizada do periodograma e a autocorrelação é baixa. Na estimação posterior do H pelo estimador wavelet, obteve-se um valor de $0,7632489$ indicando LRD, o que é correto. Apesar de as rotinas de código da linguagem R ajustarem os modelos anteriormente avaliados como os modelos apropriados, a análise de resíduos indicam que os modelos tanto do ajuste sem decomposição como dos componentes da decomposição devem ser reajustados para uma representação mais fiel da série original.



(a)

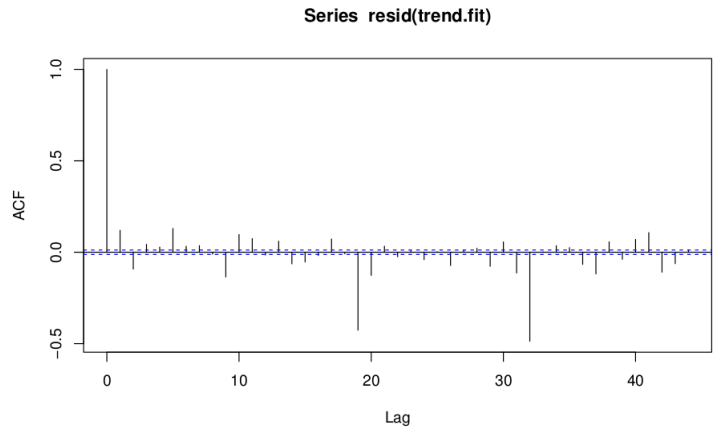


(b)

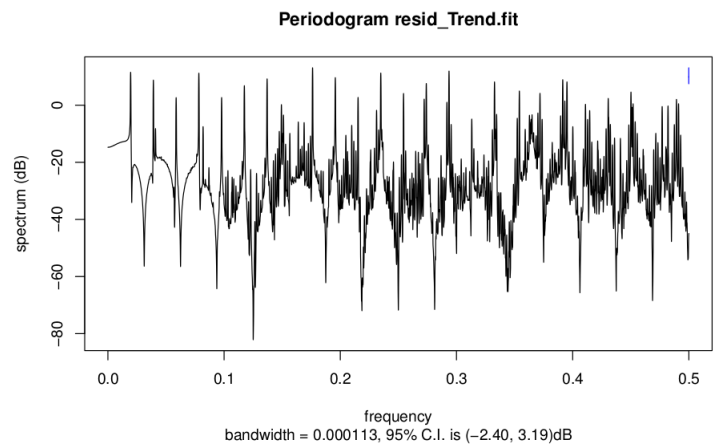


(c)

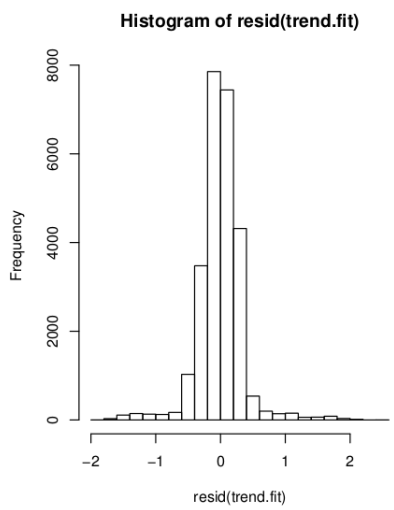
Figura 5.23: Espectros wavelet Original vs Sintética: (a) S_t vs S'_t. (b) T_t vs T'_t. (c) R_t vs R'_t.
.



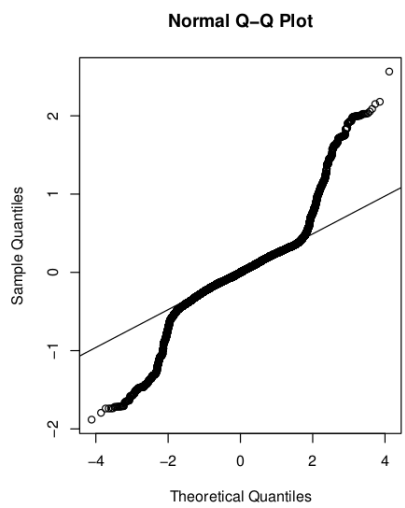
(a)



(b)

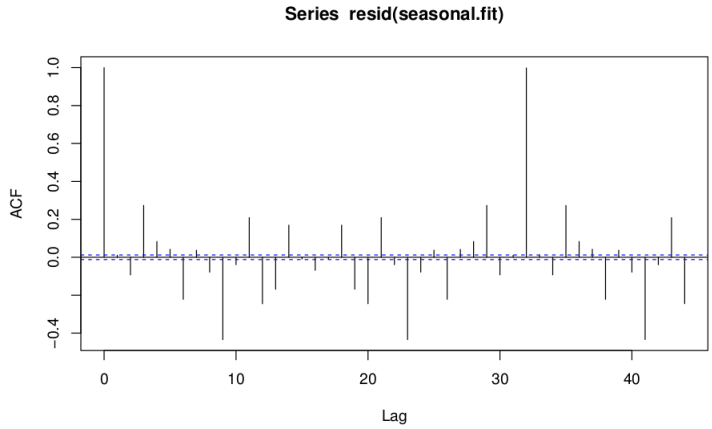


(c)

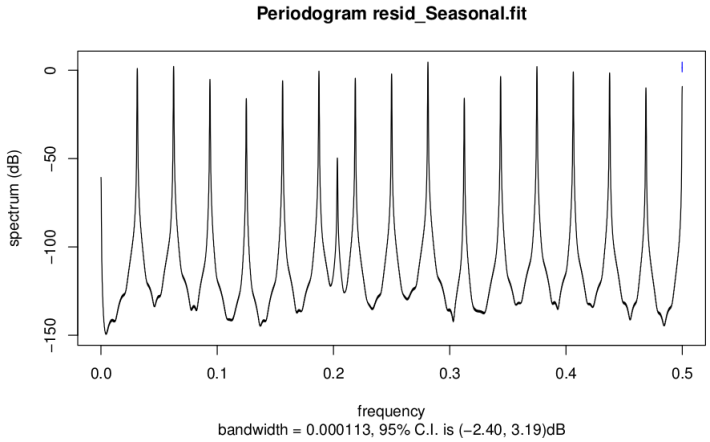


(d)

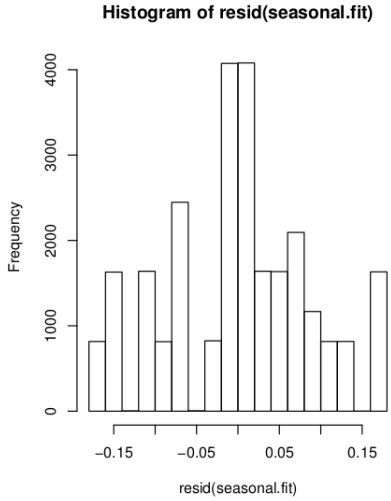
Figura 5.24: Resíduos da série sintética T_t': (a) FAC . (b) Periodograma. (c) Histograma. (d) *QQplot*.



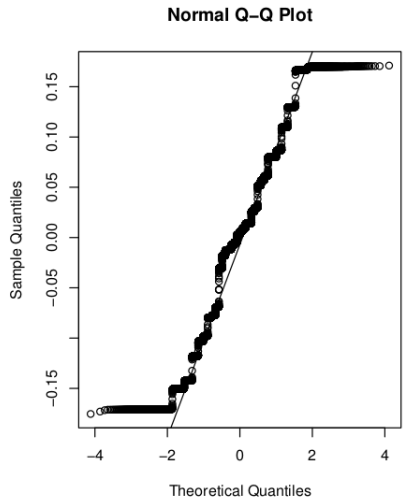
(a)



(b)

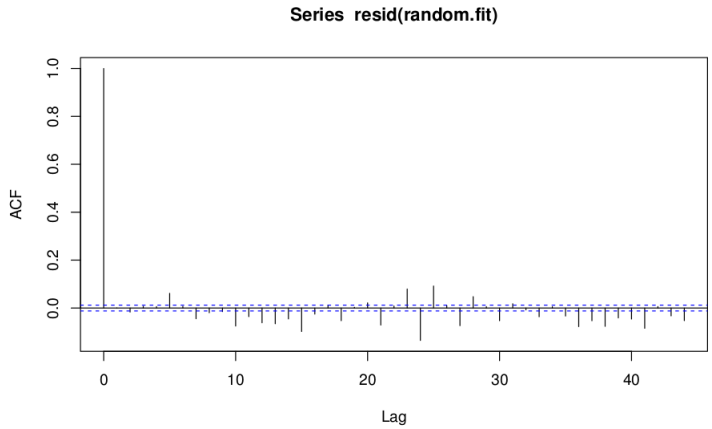


(c)

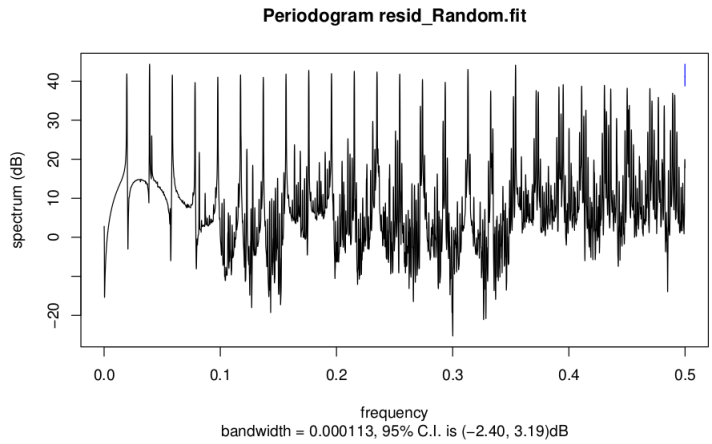


(d)

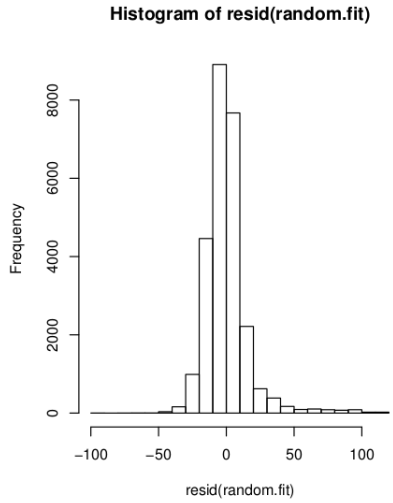
Figura 5.25: Resíduos da série sintética S'_t: (a) FAC. (b) Periodograma. (c) Histograma. (d) *QQplot*.



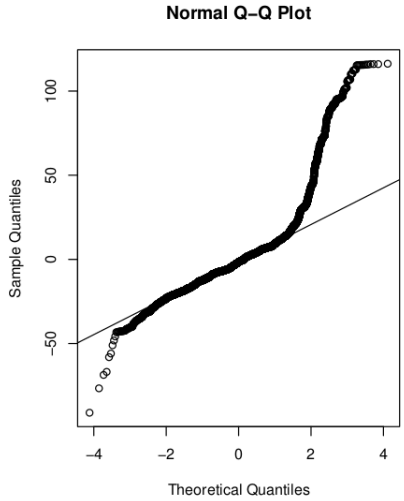
(a)



(b)

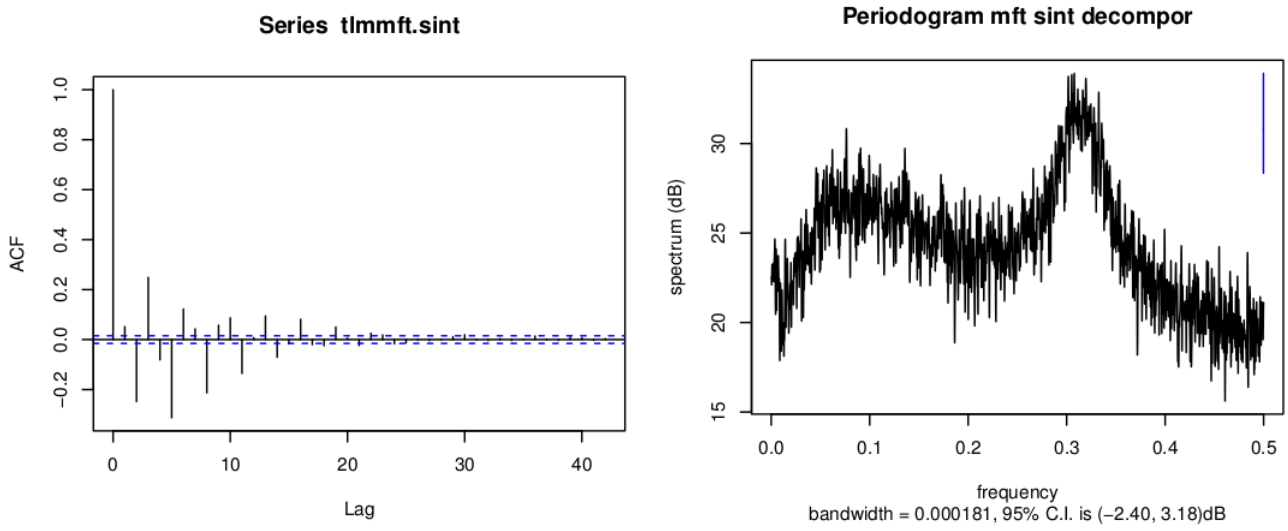


(c)



(d)

Figura 5.26: Resíduos da série sintética R'_t: (a) FAC. (b) Periodograma. (c) Histograma. (d) *QQplot*.



(a) (b)

Figura 5.27: Série sintética TLM MFT com decomposição: (a) FAC. (b) Periodograma.

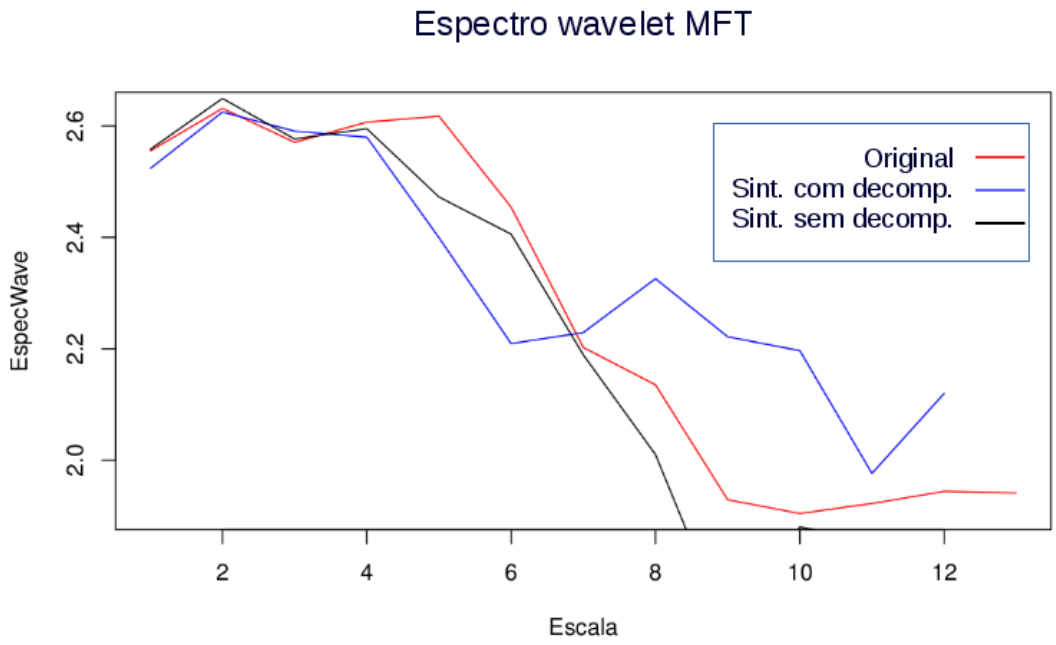
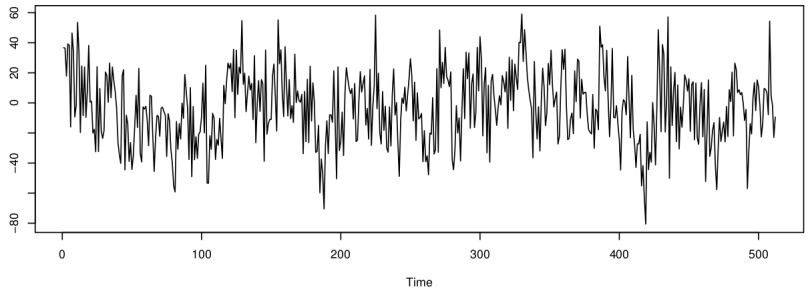
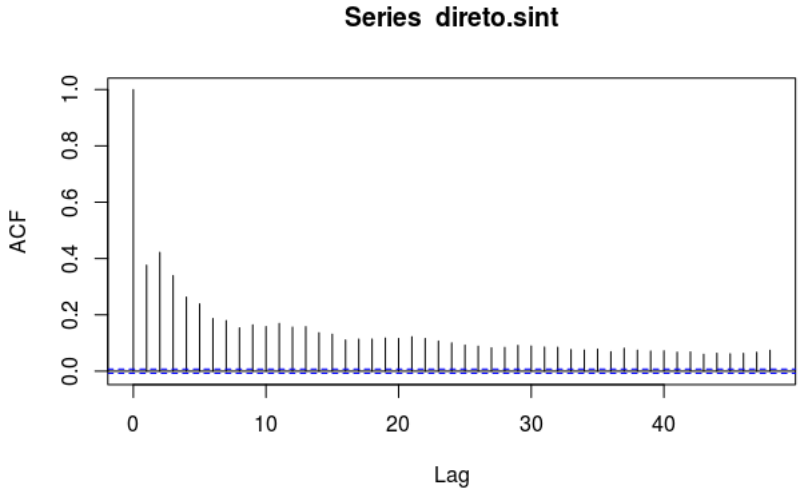


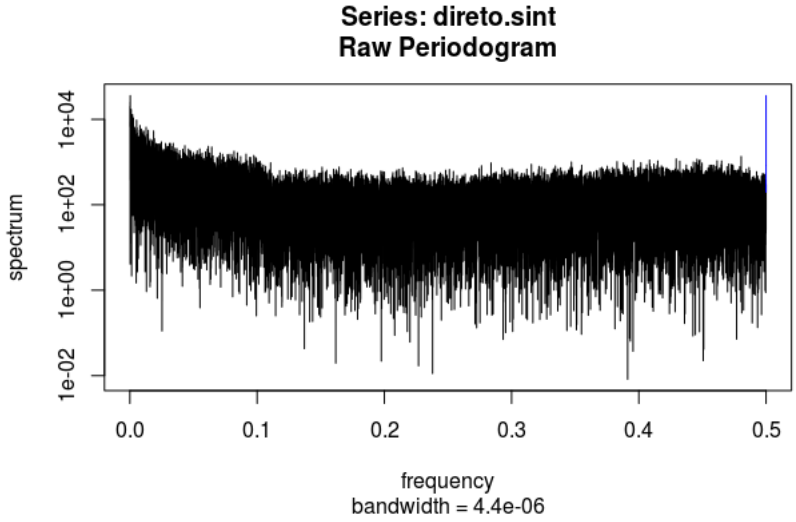
Figura 5.28: Comparação do espectro wavelet da série TLM MFT original com as sintéticas



(a)

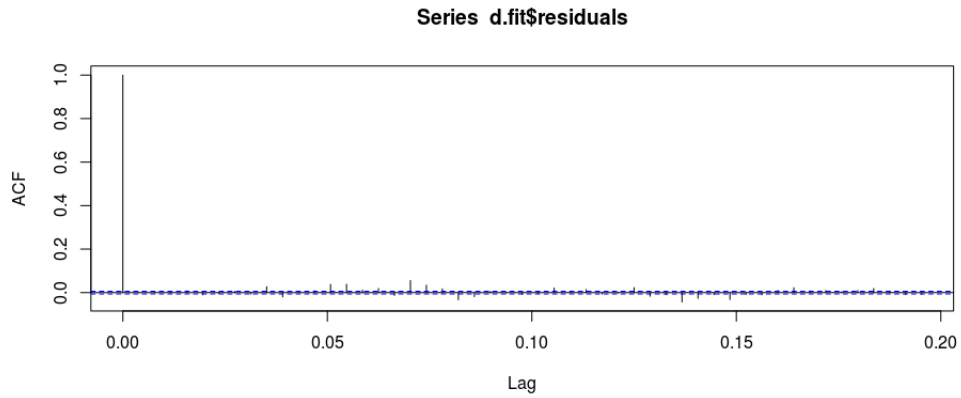


(b)

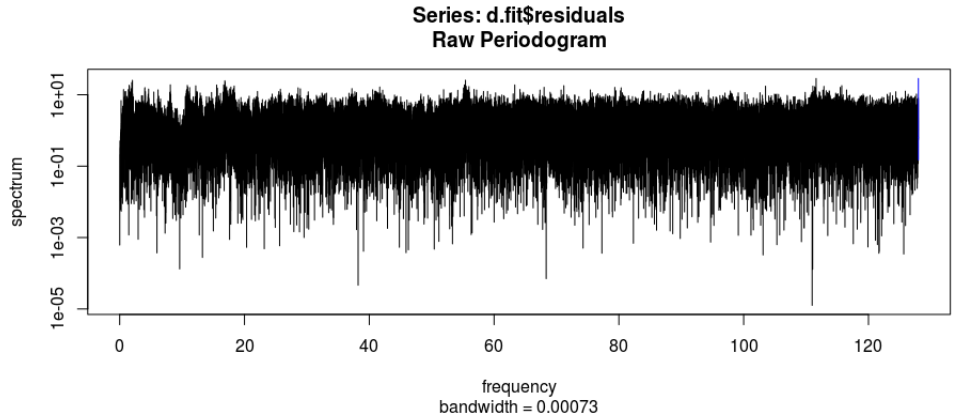


(c)

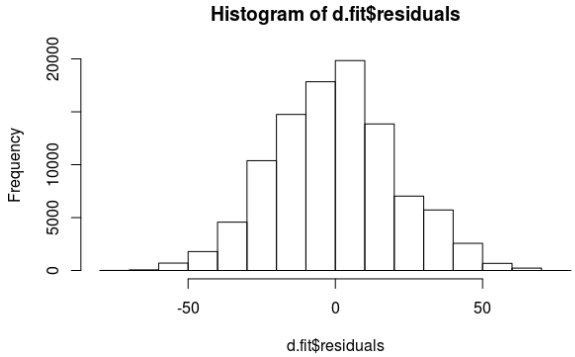
Figura 5.29: Série sintética TLM MJPEG por ajuste direto: (a) MJPEG direto $T = 512$. (b) FAC. (c) Periodograma.



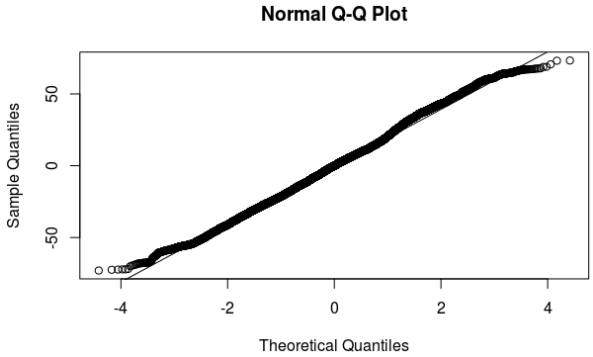
(a)



(b)

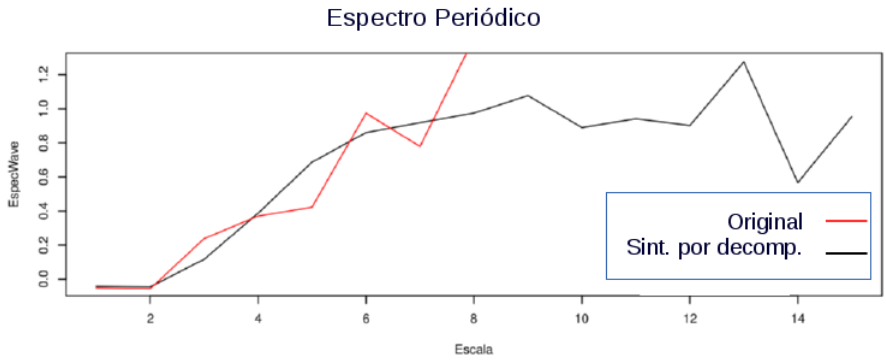


(c)

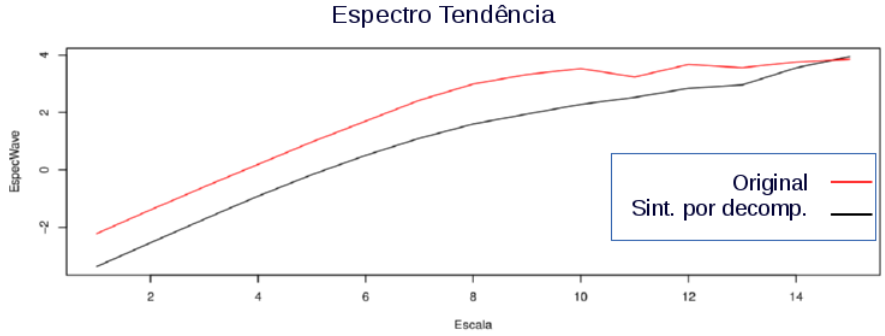


(d)

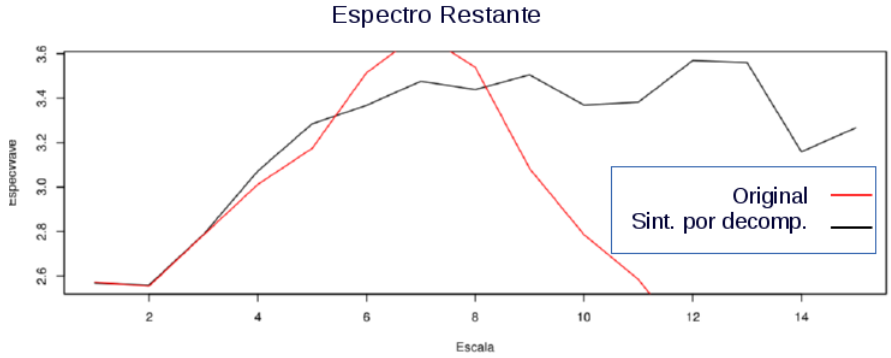
Figura 5.30: Resíduos da série sintética TLM Mjpeg por ajuste direto: (a) FAC. (b) Periodograma. (c) Histograma. (d) *QQplot*.



(a)

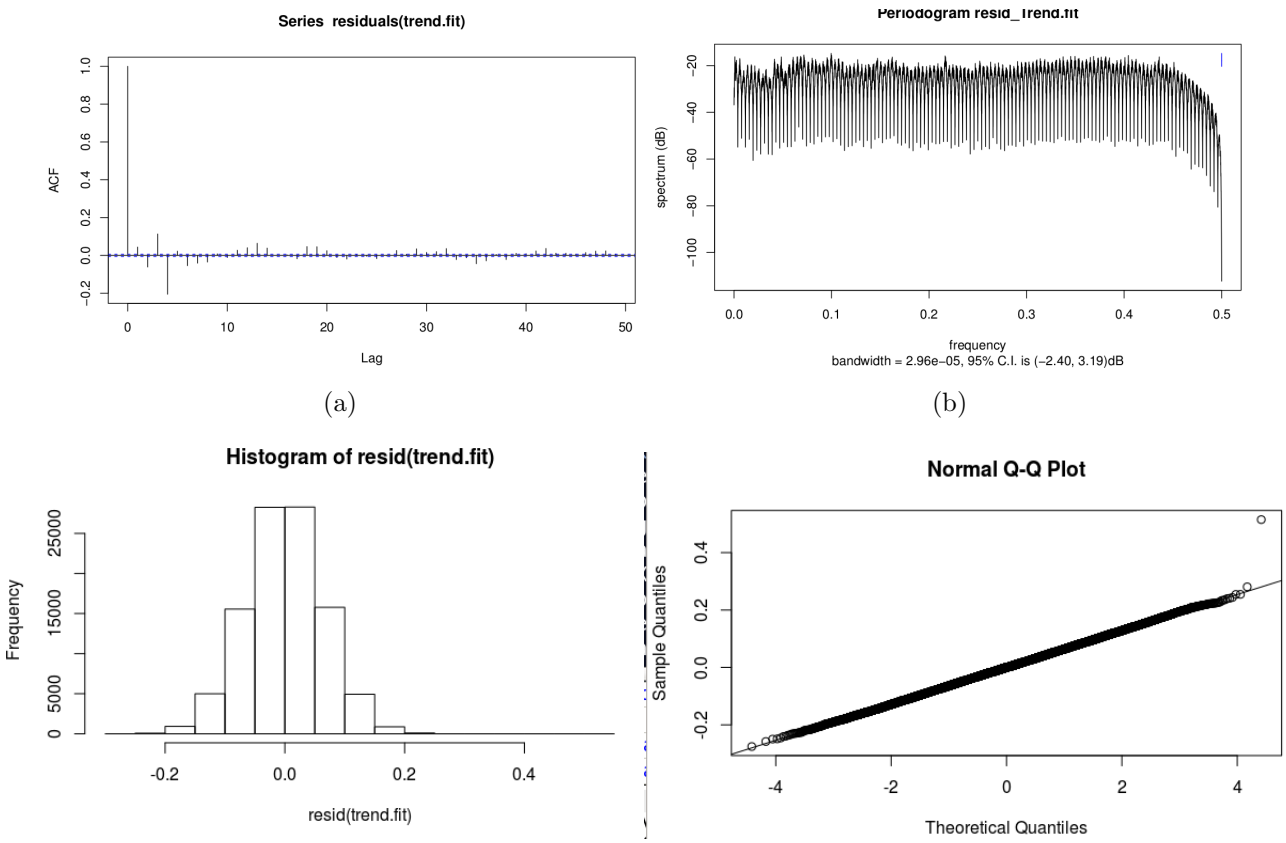


(b)



(c)

Figura 5.31: Espectros wavelet Original vs Sintético: (a) S_t vs S'_t . (b) T_t vs T'_t. (c) R_t vs R'_t.



(a) (b) (c) (d)

Figura 5.32: Resíduos da série sintética T'_t: (a) FAC . (b) Periodograma. (c) Histograma. (d) *QQplot*.

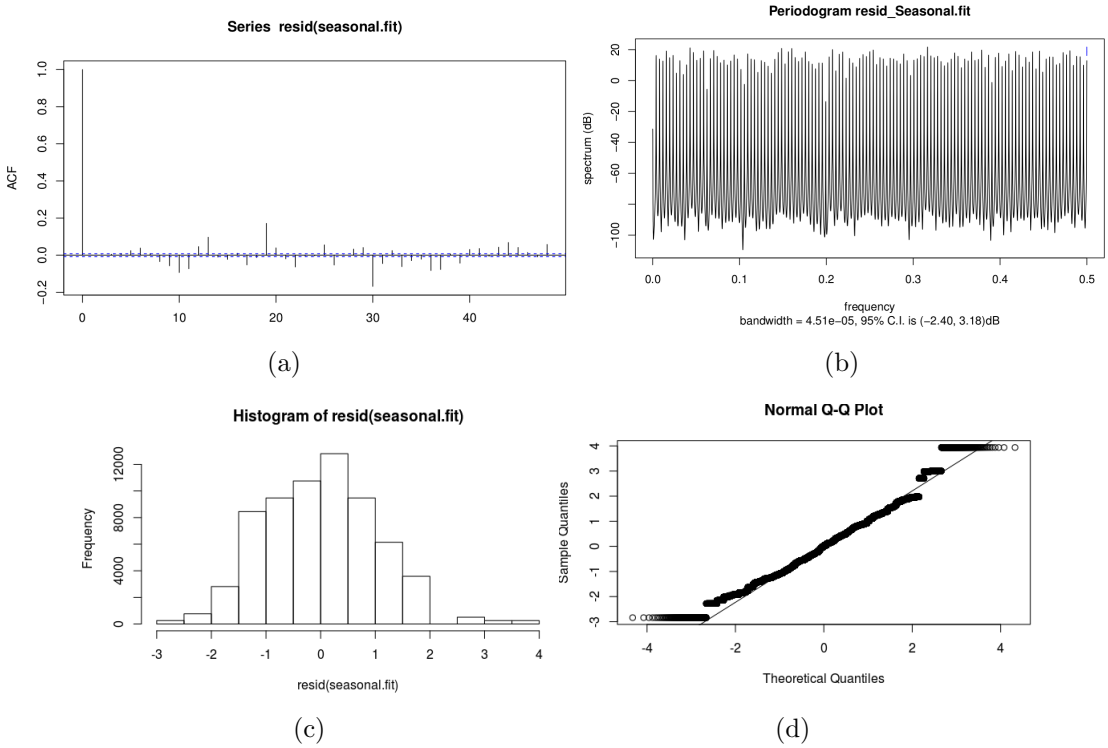


Figura 5.33: Resíduos da série sintética S'_t: (a) FAC. (b) Periodograma. (c) Histograma. (d) *QQplot*.

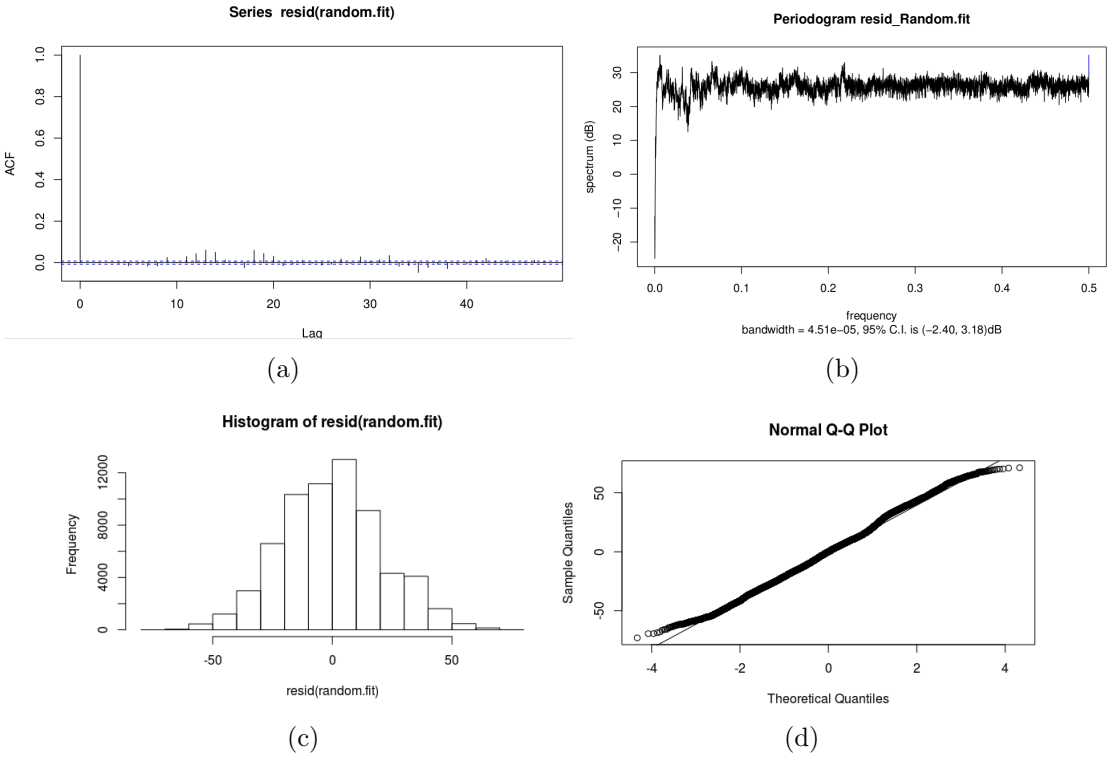
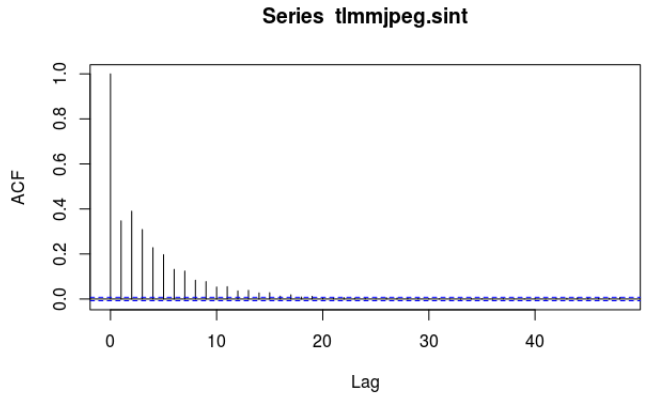
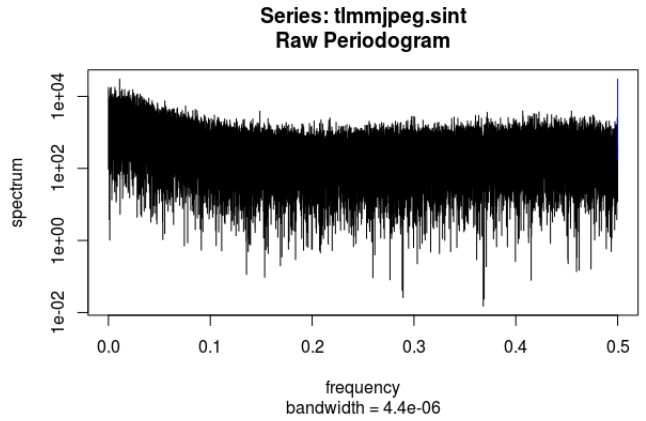


Figura 5.34: Resíduos da série sintética R'_t: (a) FAC. (b) Periodograma. (c) Histograma. (d) *QQplot*.



(a)



(b)

Figura 5.35: Série sintética TLM Mjpeg com decomposição: (a) FAC. (b) Periodograma.

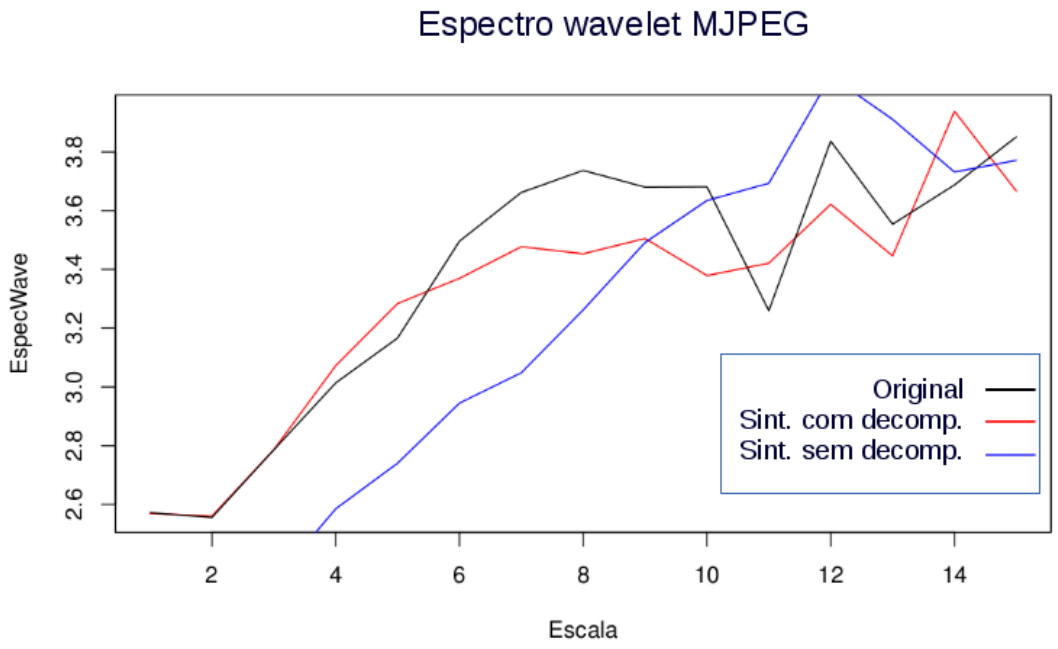


Figura 5.36: Comparação do espectro wavelet da série TLM mjpeg original com as sintéticas

Capítulo 6

Conclusões e Perspectivas

Nesta dissertação abordamos uma metodologia para analisar o tráfego intrachip obtido por simulação no nível de modelagem de hardware TLM baseado na metodologia de (SCHERRER; FRABOULET; RISSET, 2009a) com a qual validamos nossos resultados apresentados no capítulo anterior sobre as séries de tráfego. A partir dos nossos experimentos descritos neste trabalho:

- Observamos as vantagens de simulação e implementação do projeto usando o modelo TLM, permitindo obter resultados semelhantes aos do modelo CA com menor esforço e tempo, validando a redução de tempo na simulação usando modelos de maior abstração.
- Avaliamos as características estatísticas dos rastros de tráfego de plataformas descritas no nível CA e TLM por uma metodologia ágil baseada na determinação de SRD e LRD nas séries, a partir da estimação da função de autocorrelação, periodograma, espectro wavelet e o cálculo do parâmetro de Hurst.
- Evidenciamos e contrastamos as características do tráfego TLM em relação às do tráfego CA pelos cálculos anteriormente mencionados, o que nos permitiu observar que o comportamento relacionado à SRD ou LRD é mantido nas séries TLM.
- Ao evidenciar as características relevantes de SRD e LRD no tráfego TLM, conseguimos elaborar uma metodologia para ajuste de modelos de séries temporais que permite ao projetista ter um maior controle a partir da decomposição da série e o posterior ajuste de modelos paramétricos ARMA e ARFIMA. Porém, esta metodologia requer um estudo mais profundo sobre os modelos de séries com SRD ao evidenciar as limitações dos modelos ARMA usados refletidas na estimação das séries sintéticas apresentadas e discutidas no final do Capítulo 4, o que pode influenciar nos futuros resultados de avaliação das redes NoC.

Sendo o escopo deste trabalho encaminhar o ajuste de séries sintéticas no nível TLM com os modelos ARMA e ARFIMA, os objetivos definidos no começo deste trabalho foram cumpridos. Foi obtida uma metodologia de ajuste de séries por decomposição que pode ser implementada dentro do fluxo de trabalho ESL na parte de exploração de arquiteturas de comunicação levando a facilidade no reúso em posteriores projetos de hardware.

Perspectivas e trabalhos futuros

- Neste trabalho, um modelo simplificado de decomposição foi utilizado, podendo outros métodos mais precisos e modernos serem avaliadas, como os métodos wavelet, para se obter uma melhor identificação de componentes da série total.

- Nos resultados evidenciamos limitações dos modelos ARMA dentro de um ajuste automático, sendo que poderia ser benéfico incorporar alguma metodologia de ajuste manual com estes modelos ou usar outros modelos paramétricos lineares que representem SRD, mas que permitam manter os diversos elementos periódicos da série, quando existirem. Também poderia se considerar outros modelos complexos para geração de séries sintéticas, como indicado na Secão 3.2.

- Poderia se identificar grupos de instruções do programa da aplição e associa-los com um comportamento no tráfego ou *throughput* usando *dummy code* nos processadores; isto permitiria, avaliar o comportamento do código para futuras aplicações *multi-thread*.

- Poderia se avaliar com o mecanismo apresentado neste trabalho o comportamento de tráfego numa implementação sem anotações de tempo na comunicação; caso de uma implementação mais abstrata que reduziria os tempos de simulação.

- Deve-se avaliar o comportamento de uma NoC com o tráfego sintético gerado em comparação ao tráfego original. Isto implica na montagem de uma NoC que permita a medição de diversas métricas de desempenho da rede. Isto requereria a modelagem e desenvolvimento de uma plataforma, considerando-se que a maioria das redes de simulação disponíveis estão no nível RTL ou CA.

Publicações

Os resultados deste trabalho, especificamente da avaliação estatística entre tráfego TLM e CA, foram publicados no trabalho *"Long Range Dependence in Intrachip Transaction Level Traffic"* apresentado no evento *"IV IEEE LATIN AMERICAN SYMPOSIUM ON CIRCUITS AND SYSTEMS (LASCAS)"* realizado de 27 de Fevereiro a 1 Março do 2013 em Cusco, Peru.

Bibliografia

ABRY, P.; VEITCH, D. Wavelet analysis of long-range-dependent traffic. *Information Theory, IEEE Transactions on*, v. 44, n. 1, p. 2 –15, jan 1998. ISSN 0018-9448.

AUGE, I. et al. Platform-based design from parallel c specifications. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, v. 24, n. 12, p. 1811–1826, 2005. ISSN 0278-0070.

AUGUSTO, M. L. *Contribuição para análise de teletráfego com dependência de longa duração*. Tese (Doutorado) — Universidade de São Paulo, 2009.

BAILEY, B.; MARTIN, G.; PIZIALI, A. *ESL design and verification: a prescription for electronic system-level methodology*. 1 st ed.. ed. [S.l.]: Morgan Kaufmann Publishers, 2007. (The Morgan Kaufmann Series in Systems in Silicon).

BERAN, J. *Statistics for Long-Memory Processes*. [S.l.]: Chapman and Hall/CRC, 1994. ISBN 0412049015.

BOGDAN, P.; MARCULESCU, R. Non-stationary traffic analysis and its implications on multicore platform design. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, v. 30, n. 4, p. 508–519, 2011.

BOMBIERI, N.; FUMMI, F.; PRAVADELLI, G. Rtl-tlm equivalence checking based on simulation. In: . Lviv, Ukraine: [s.n.], 2008. p. 214 – 217.

BOMBIERI, N.; FUMMI, F.; PRAVADELLI, G. Automatic abstraction of rtl ips into equivalent tlm descriptions. *Computers, IEEE Transactions on*, v. 60, n. 12, p. 1730 –1743, dec. 2011. ISSN 0018-9340.

CAI, L.; GAJSKI, D. Transaction level modeling: an overview. In: *Proceedings of the 1st IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*. New York, NY, USA: ACM, 2003. (CODES+ISSS '03), p. 19–24. ISBN 1-58113-742-7. Disponível em: <http://doi.acm.org/10.1145/944645.944651>.

CASTIGLIONI, P. Zero padding. In: _____. *Encyclopedia of Biostatistics*. John Wiley & Sons, Ltd, 2005. ISBN 9780470011812. Disponível em: <http://dx.doi.org/10.1002/0470011815.b2a12087>.

CETTO, L. Mandelbrot set vectorized. 2011. Disponível em: <http://www.mathworks.com/matlabcentral/fileexchange/1329-mandelbrot-set-vectorized>.

CHANDY, K.; MISRA, J. Distributed simulation: A case study in design and verification of distributed programs. *Software Engineering, IEEE Transactions on*, SE-5, n. 5, p. 440 – 452, sept. 1979. ISSN 0098-5589.

COPPOLA, M. et al. *Design of Cost-Efficient Interconnect Processing Units: Spidergon STNoC*. [S.l.]: CRC Press, Inc., 2008. 288 p.

DAMARAJU, S. et al. A 22nm ia multi-cpu and gpu system-on-chip. In: *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International*. [S.l.: s.n.], 2012. p. 56–57. ISSN 0193-6530.

DAS, S.; PAN, I. *Fractional Order Signal Processing: Introductory Concepts and Applications (SpringerBriefs in Applied Sciences and Technology)*. 2012. ed. [S.l.]: Springer, 2011. ISBN 9783642231162.

FAN, J.; YAO, Q. *Nonlinear Time Series: Nonparametric and Parametric Methods*. New York: Springer, 2005. (Springer Series in Statistics). ISBN 9780387261423. Disponível em: <http://www.worldcat.org/isbn/0387261427>.

FINDENIG, R. et al. Fast and accurate uml state chart modeling using tlm; control flow abstraction. In: *High Level Design Validation and Test Workshop (HLDVT), 2010 IEEE International*. [S.l.: s.n.], 2010. p. 97 –102. ISSN 1552-6674.

FLYNN, M. J. Some computer organizations and their effectiveness. *Computers, IEEE Transactions on*, C-21, n. 9, p. 948–960, 1972.

FLYNN, M. J.; HUNG, P.; RUDD, K. W. Deep submicron microprocessor design issues. *Micro, IEEE*, v. 19, n. 4, p. 11–22, 1999.

FLóREZ, M. J. S. *Estimativa de desempenho de uma NoC a partir de seu modelo em SystemC - TLM*. Tese (Doutorado) — Universidade de São Paulo, 2006.

FLóREZ, M. J. S. et al. *The LRD traffic impact on the NoC-based SoCs*. [S.l.]: ACM, 2010. 97-102 p.

FOUNDATION, T. R. *The R Project for Statistical Computing*. 2012. Disponível em: <http://www.r-project.org/>.

GAPH. *HERMES*. 2011. Disponível em: <https://corfu.pucrs.br/redmine/projects/hemps/wiki/HERMES>.

GARZON, J. S. E. *Estimativas de desempenho da estrutura de cominicação de SoC a partir de modelos de transações*. Tese (Doutorado) — Universidade de São Paulo, 2008.

GEWEKE, J.; PORTER-HUDAK, S. The estimation and application of long memory time series models. *Journal of Time Series Analysis*, Blackwell Publishing Ltd, v. 4, n. 4, p. 221–238, 1983. ISSN 1467-9892. Disponível em: <http://dx.doi.org/10.1111/j.1467-9892.1983.tb00371.x>.

GHENASSIA, F. (Ed.). *Transaction-Level Modeling with SystemC: TLM Concepts and Applications for Embedded Systems*. 1. ed. [S.l.]: Springer, 2005. 286 p. ISBN 9780387262321.

GREINER, A.; SOCLIB. *SoCLib : Une plate-forme de prototypage virtuel pour systèmes multi-processeurs intégrés sur puce*. 2007. Disponível em: <leom.ec-lyon.fr/fetch/programme/greiner.pdf>.

GROUP, O. D. W. *VSI Alliance Virtual Component Interface Standard*. 2001. Disponível em: <sen.enst.fr/filemanager/active?fid=663>.

GRöTKER, T. et al. *System Design with SystemC*. 1. ed. [S.l.]: Springer, 2002. ISBN 9781402070723.

HOSKING, J. R. M. Fractional differencing. *Biometrika*, v. 68, n. 1, p. 165–176, 1981. Disponível em: <http://biomet.oxfordjournals.org/content/68/1/165.abstract>.

HURST, H. Long-term storage capacity of reservoirs. *Transactions of the American Society of Civil Engineers*, p. 770–808, 1951.

HYNDMAN, R. J. *Package forecast*. 2012. Disponível em: <http://cran.r-project.org/web/packages/forecast/index.html>.

JEONG, H.; MCNICKLE, D.; PAWLIKOWSKI, K. Comparison of various estimators of hurst parameter in simulated fgn. Department of Computer Science and Software Engineering, University of Canterbury, p. 25pp, 2006. Disponível em: <http://hdl.handle.net/10092/3090>.

JEONG, H.-D.; MCNICKLE, D.; PAWLIKOWSKI, K. Fast self-similar teletraffic generation based on fgn and wavelets. In: *Networks, 1999. (ICON '99) Proceedings. IEEE International Conference on*. [S.l.: s.n.], 1999. p. 75 – 82.

KEUTZER, K. et al. System-level design: orthogonalization of concerns and platform-based design. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, v. 19, n. 12, p. 1523–1543, 2000.

KIASARI, A. E. et al. A markovian performance model for networks-on-chip. In: *Parallel, Distributed and Network-Based Processing, 2008. PDP 2008. 16th Euromicro Conference on*. [S.l.: s.n.], 2008. p. 157–164.

LELAND, W. E. et al. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Trans. Netw.*, IEEE Press, Piscataway, NJ, USA, v. 2, n. 1, p. 1–15, fev. 1994. ISSN 1063-6692.

LIMA, A. B. d. *Contribuições à Modelagem de Teletráfego Fractal*. Tese (Doutorado) — Universidade de São Paulo, 2008.

LIU, W. et al. *A NoC Traffic Suite Based on Real Applications*. [S.l.]: IEEE Computer Society, 2011. 66-71 p.

LUDESCHER, J. et al. On spurious and corrupted multifractality: The effects of additive noise, short-term memory and periodic trends. *Physica A: Statistical Mechanics and its Applications*, v. 390, n. 13, p. 2480 – 2490, 2011. ISSN 0378-4371. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0378437111001981>.

MAECHLER, M. *Package fracdiff*. 2012. Disponível em: <http://cran.r-project.org/web/packages/fracdiff/index.html>.

MAKRIDAKIS, S. G.; WHEELWRIGHT, S. C.; HYNDMAN, R. J. *Forecasting: Methods and Applications*. 3. ed. [S.l.]: Wiley, 1997. ISBN 9780471532330.

MARCULESCU, R. On-chip networks: Two sides of the same coin. *Design I& Test of Computers, IEEE*, v. 27, n. 4, p. 80–80, 2010.

MARKOVIć, D.; KOCH, M. Sensitivity of hurst parameter estimation to periodic signals in time series and filtering approaches. *Geophysical Research Letters*, v. 32, n. 17, p. n/a–n/a, 2005. ISSN 1944-8007. Disponível em: <http://dx.doi.org/10.1029/2005GL024069>.

MATLAB. *version 7.14.0 (R2012a)*. Natick, Massachusetts: The MathWorks Inc., 2012.

MELLO, F. L. d. *Estudo e Implementação de um Gerador de Tráfego com Dependência de Longa Duração*. Tese (Doutorado) — São Paulo, 2006.

MEYER, D. *Classical Seasonal Decomposition by Moving Averages*. 2012. Disponível em: <http://finzi.psych.upenn.edu/R/library/stats/html/decompose.html>.

NICOPOULOS, C.; NARAYANAN, V.; DAS, C. *Network-on-Chip Architectures: A Holistic Design Exploration*. Springer London, Limited, 2009. (Lecture Notes in Electrical Engineering, 45). ISBN 9789048130313. Disponível em: <http://books.google.com.br/books?id=5Oi4d5TGdCwC>.

OSCI, O. S. I. Systemc 2.2. 2011. Disponível em: <http://www.accellera.org/downloads/standards/systemc/>.

OSCI, O. S. I. Tlm 2.0.1. 2011. Disponível em: <http://www.accellera.org/downloads/standards/systemc/>.

PANADES, I. M.; GREINER, A.; SHEIBANYRAD, A. A low cost network-on-chip with guaranteed service well suited to the gals approach. In: *Nano-Networks and Workshops, 2006. NanoNet '06. 1st International Conference on*. [S.l.: s.n.], 2006. p. 1 –5.

PARK, K.; WILLINGER, W. *Self-Similar Network Traffic and Performance Evaluation*. 1st. ed. New York, NY, USA: John Wiley & Sons, Inc., 2000. ISBN 0471319740.

PASRICHA, S.; DUTT, N. *On-Chip Communication Architectures: System on Chip Interconnect*. [S.l.]: Morgan Kaufmann Publishers Inc., 2008. 536 p.

PATIñO, G.; CHAU, W. J.; STRUM, M. *On-chip Traffic Charaterization based on a MPSoC Architecture Exploration*. 2010.

PEH, L.-S.; JERGER, N. E. *On-Chip Networks*. [S.l.]: Morgan and Claypool Publishers, 2009. 142 p.

POYNTON, C. *Digital Video and HDTV Algorithms and Interfaces*. 1. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003. ISBN 1558607927.

SCHERRER, A.; FRABOULET, A.; RISSET, T. *Automatic phase detection for stochastic on-chip traffic generation*. [S.l.]: ACM, 2006. 88-93 p.

SCHERRER, A.; FRABOULET, A.; RISSET, T. *A Generic Multi-Phase On-Chip Traffic Generation Environment*. [S.l.]: IEEE Computer Society, 2006. 23-27 p.

SCHERRER, A.; FRABOULET, A.; RISSET, T. Long-range dependence and on-chip processor traffic. *Microprocess. Microsyst.*, v. 33, n. 1, p. 72–80, 2009.

SCHERRER, A.; FRABOULET, A.; RISSET, T. On-chip processor traffic modeling for networks-on-chip design. In: _____. *Network on Chips: Theory and Practice*. [S.l.]: CRC Press, 2009. cap. 4, p. 95–122.

SCHIRNER, G.; GERSTLAUER, A.; DOMER, R. Fast and accurate processor models for efficient mpsoc design. *ACM Transactions on Design Automation of Electronic Systems*, v. 15, n. 2, 2010. ISSN 10844309.

SHENG, H.; CHEN, Y.; QIU, T. On the robustness of hurst estimators. *Signal Processing, IET*, v. 5, n. 2, p. 209 –225, april 2011. ISSN 1751-9675.

SHUMWAY, R. H.; STOFFER, D. S. *Time Series Analysis and Its Applications*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005. ISBN 0387989501.

SIGMALAB. *MFT application code*. 2012. Disponível em: <http://code.google.com/p/sigmav/>.

SOCLIB. *MIPS component documentation*. 2011. Disponível em: <http://www.soclib.fr/trac/dev/wiki/Component/Mips>.

SOCLIB. *Modeling of RISC processors*. 2011. Disponível em: <http://www.soclib.fr/trac/dev/wiki/WritingRules/RISC>.

SOCLIB. *SoCLib Simulation Platform*. 2011. Disponível em: <http://www.soclib.fr/trac/dev/wiki/InstallationNotes>.

SOCLIB. *VciVgmn component documentation*. 2011. Disponível em: <http://www.soclib.fr/trac/dev/wiki/Component/VciVgmn>.

SOCLIB. *Writing TLM2.0-compliant timed SystemC simulation models for SoCLib*. 2011. Disponível em: <http://www.soclib.fr/trac/dev/wiki/WritingRules/Tlmt>.

SOTERIOU, V.; WANG, H.; PEH, L.-S. *A Statistical Traffic Model for On-Chip Interconnection Networks*. [S.l.]: IEEE Computer Society, 2006. 104-116 p.

VARATKAR, G. V.; MARCULESCU, A. R. On-chip traffic modeling and synthesis for mpeg-2 video applications. *IEEE Trans. Very Large Scale Integr. Syst.*, v. 12, n. 1, p. 108–119, 2004.

VEITCH, D. Code for the estimation of scaling exponents. 2012. Disponível em: <http://www.cubinlab.ee.unimelb.edu.au/~darryl/secondorder_code.html>.

VIAUD, E.; PECHEUX, F.; GREINER, A. An efficient tlm/t modeling and simulation environment based on conservative parallel discrete event principles. In: *Design, Automation and Test in Europe, 2006. DATE '06. Proceedings*. [S.l.: s.n.], 2006. v. 1, p. 1 –6.

VéHEL, J. L.; RIEDI, R. Fractional brownian motion and data traffic modeling: The other end of the spectrum. In: *Fractals in Engineering*. [S.l.]: Springer, 1997. p. 185–202.

WHITTLE, P. Estimation and information in stationary time series. *Arkiv för Matematik*, Kluwer Academic Publishers, v. 2, p. 423–434, 1953. ISSN 0004-2080. Disponível em: <http://dx.doi.org/10.1007/BF02590998>.

WIKLUND, D.; SATHE, S.; LIU, D. Network on chip simulations for benchmarking. In: *System-on-Chip for Real-Time Applications, 2004.Proceedings. 4th IEEE International Workshop on*. [S.l.: s.n.], 2004. p. 269–274.

WILTON, S. J. E.; SALEH, R. Programmable logic ip cores in soc design: opportunities and challenges. In: *Custom Integrated Circuits, 2001, IEEE Conference on*. [S.l.: s.n.], 2001. p. 63–66.

YONGHUI, L. et al. Performance modeling of fully adaptive wormhole routing in 2d-mesh network-on-chip with mmpp (2) input traffic. In: *Information Science and Engineering, 2008. ISISE '08. International Symposium on*. [S.l.: s.n.], 2008. v. 2, p. 58–62.