Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Ciência da Computação

Predição da Qualidade de Serviço em Grades Computacionais P2P

Marcus Williams Aquino de Carvalho

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Sistemas Distribuídos

Francisco Vilar Brasileiro
(Orientador)

Campina Grande, Paraíba, Brasil

Ficha Catalográfica Elaborada pela Biblioteca Central da UFCG

C331p Carvalho, Marcus Williams Aquino de
Predição da qualidade de serviço em grades computacionais P2P / Marcus Williams Aquino Carvalho. – Campina Grande: UFCG, 2011.
95 f. : il.

Dissertação (Mestrado em Ciência da Computação) Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.
Orientador: Prof. Dr. Francisco Vilar Brasileiro.

1. Redes de Computadores 2. Sistemas de Processamento Distribuído I. Título

CDU 004.7(043)

"PREDIÇÃO DA QUALIDADE DE SERVIÇO EM GRADES COMPUTACIONAIS P2P"

MARCUS WILLIAMS AQUINO DE CARVALHO

DISSERTAÇÃO APROVADA EM 17.03.2011

FRANCISCO VILAR BRASILEIRO, Ph.D
Orientador(a)

NAZARENO FERREIRA DE ANDRADE, D.Sc
Examinador(a)

ALEXANDRE NOBREGA DUARTE, D.Sc
Examinador(a)

CAMPINA GRANDE - PB

Resumo

As grades computacionais entre pares (P2P) possibilitam a agregação de uma grande quantidade de recursos, espalhados em diferentes domínios administrativos, formando uma infraestrutura de computação de alto desempenho em larga escala com um baixo custo associado. Em uma grade P2P, um nó disponibiliza aos outros participantes do sistema seus recursos quando estes estão ociosos e, em troca, pode usar os recursos ociosos de outros nós, sempre que sua demanda de processamento for superior à capacidade que ele possui localmente. Como se trata de um ambiente colaborativo, mecanismos de incentivo são necessários para motivar os usuários a doarem seus recursos ociosos e promover a colaboração entre os participantes da grade. Porém, mesmo com a utilização de mecanismos de incentivo, as grades P2P ainda possuem como característica a variação significativa, ao longo do tempo, de vários atributos associados à qualidade de serviço (QoS). Os fatos dos recursos não serem dedicados e da demanda não ser conhecida previamente trazem incertezas à QoS oferecida. Se, por um lado, é aceitável a incerteza da QoS das grades P2P em troca dos seus baixos custos operacionais, por outro lado, existem várias ocasiões em que ter alguma estimativa da QoS fornecida é bastante importante. Para que as grades P2P possam ser utilizadas por usuários que possuem aplicações com restrições de tempo, por exemplo, é necessário que se tenha uma estimativa de atributos de qualidade de serviço a serem oferecidos pela grade. Tendo em vista a alta variação em atributos de qualidade de serviço e a falta de estimativas desses atributos para os usuários, neste trabalho são propostos modelos de predição para a capacidade de processamento que estará disponível para um determinado participante da grade P2P em períodos de tempo futuros. Os resultados mostram que os modelos que se baseiam no funcionamento interno do sistema para realizar suas predições (caixa-branca e caixa-cinza) são, no geral, melhores do que modelos de predição que se baseiam apenas em dados históricos (caixa-preta). Para avaliar os modelos de predição, é proposto nesse trabalho um modelo de geração de carga sintética para grades computacionais P2P.

Abstract

Peer-to-peer (P2P) grid systems have been proposed as an economical way to increase the processing capabilities of information technology (IT) infrastructures. In a P2P grid, a peer donates its idle resources to the other peers in the system, and, in exchange, can use the idle resources of other peers when its processing demand surpasses its local computing capacity. For collaborative systems like these, incentive mechanisms are needed in order to make the systems work. However, even using incentive mechanisms, the quality of the service provided by P2P grids varies significantly over time. The facts that the resources are not dedicated to the grid and that the demand is unknown bring uncertainties for QoS attributes. Despite their cost-effectiveness, scheduling of processing demands on IT infrastructures that encompass P2P desktop grids is more difficult. At the root of this difficulty is the fact that the quality of the service provided by P2P desktop grids varies significantly over time. This way, users that execute time constraint applications are compromised by this best-effort behaviour. The research we report in this work tackles the problem of estimating the quality of service of P2P grids. The models proposed are able to estimate total processing capacity that is available for a peer in the system at future periods of time. Our results show that, in general, the prediction models that uses system knowledge to perform the predictions (grey-box and white-box models) outperforms the approaches which use only historical data to apply the predictions (black-box models). In order to evaluate the prediction models, we proposed a synthetic workload generator for P2P grids.

Agradecimentos

Ao meu orientador Fubica, por todos os ensinamentos transmitidos durante esses anos.

Aos meus pais, Rebeca e Marcão, por tudo que me proporcionaram durante toda a minha existência.

A Raquel, minha inspiração tanto no afetivo quanto no profissional.

A vovó Teresa (*in memoriam*), que me incentivou a estudar em Campina Grande, com o objetivo de ter uma formação de maior nível. A vovô Aecio (*in memoriam*), pelo exemplo de profissional e pessoa que foi. A vovó Eliza e vovô Waldemir, pelo enorme carinho e orgulho que me passam.

Aos meus irmãos, Rafael e Nathália, pela convivência pacífica e agradável durante os fins de semana, férias, feriados e afins.

Aos meus outros irmãos (estes por opção), João Arthur e Paulo Ditarso, pela convivência (pseudo)familiar durante os anos de mestrado, tanto dentro de casa quanto fora.

Aos amigos do LSD, pelos momentos de aprendizado e (principalmente) de diversão. Às velhas amizades de João Pessoa e às novas adquiridas em Campina Grande.

Aos professores que tive durante minha formação.

A Dona Inês pelos inúmeros almoços. Ao Portuga pelo X-tudo. A Dona Alba pelo rubacão. A Seu Paraná pelo churrasco. A Tenebras, Alemão e Munhoz pela cerveja gelada. A Dr. Ruy pelo joelho novo.

Agradeço também à Hewlett-Packard (HP) pela bolsa recebida através dos seus projetos e, mantendo a tradição, agradeço ao povo brasileiro que ajudou financiar minha formação.

Conteúdo

Lista de Símbolos

ℓ	*Nó (ou peer) local da grade P2P*
$\mathcal{G}$	*Conjunto de nós que compõem a grade P2P*
p	*Nó (ou peer) da grade*
b_d^c	*Balanço do crédito de favores que o nó p_c associa ao nó p_d*
$\mathcal{M}_i$	*Conjunto de máquinas pertencentes ao nó i da grade*
$m_{i,j}$	*Capacidade de processamento da j-ésima máquina do nó i*
$\mathcal{A}_i^t$	*Conjunto de alocações das máquinas do nó doador i no turno t para o nó local*
$a_{i,j}^t$	*Alocação da máquina j do nó i para o nó local ℓ no turno t*
t_c	*Turno atual, ou turno inicial da janela de predição*
t_p	*Turno final da janela de predição*
Δ	*Janela de predição*
$\mathcal{C}(\Delta)$	*Capacidade total de processamento que o nó local irá obter em Δ*
$\mathcal{G}_c(t)$	*Conjunto dos nós da grade $\mathcal{G}$ que estão consumindo recursos no turno t*
$\mathcal{G}_d(t)$	*Conjunto dos nós da grade $\mathcal{G}$ que estão doando recursos no turno t*
$B_d(t)$	*Somatório do crédito que p_d associa aos consumidores no início do turno t*
$r_d(t)$	*Total de recursos do nó p_d disponíveis à grade no turno t*
$R_d(t)$	*Estimativa de recursos disponíveis do nó p_d para o nó local no turno t*
$\Psi(\Delta)$	*Estimativa da capacidade de processamento que o nó local obterá da grade em Δ*
Q	*Coeficiente de Qualidade de Ajuste*
n	*Quantidade de nós na grade*
u	*Quantidade de usuários por nó da grade*
r	*Quantidade de recursos por nó da grade*
OR	*Proporção de tarefas submetidas que estão alocadas em alguma máquina*
ξ	*Erro na predição*

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

A computação está cada vez mais presente em todas as áreas de atuação, tanto na academia quanto na indústria. A capacidade computacional continua crescendo, como já previa a Lei de Moore, e os seus custos estão sendo reduzidos. Por outro lado, as aplicações estão demandando cada vez mais poder computacional para realizar suas tarefas. Algumas aplicações fazem parte de um grupo chamado *computação de alto desempenho*, que está bastante presente na área de pesquisa que utiliza infra-estruturas de Tecnologia da Informação (TI) para fazer ciência, denominada *e-Science* [41]. Para atingir seus requisitos, essas aplicações geralmente são paralelizadas, ou seja, são divididas em várias partes que serão executadas paralelamente em infra-estruturas computacionais de alto desempenho.

Em meados dos anos 90, surgiu o conceito de grades computacionais [31], que consiste em uma infra-estrutura de computação em larga escala que permite o compartilhamento de recursos[1] entre entidades de diferentes domínios administrativos. Esses recursos compartilhados geralmente são heterogêneos e podem ser de diversos tipos, como: clusters, máquinas desktop, instrumentos digitais, etc. Quando uma grade é composta apenas por recursos dedicados, ela é chamada grade de serviço. Um outro tipo de grade existente é a grade oportunista, que propõe o compartilhamento da capacidade ociosa de processamento de recursos de diversas instituições. Este tipo de grade computacional tornou-se bastante conhecido através da chamada computação voluntária, que tem como projeto mais famoso o SETI@home

[1] Durante todo o documento, as palavras "recursos" e "máquinas" são usadas de maneira intercambiável.

[36]. Na computação voluntária, os participantes doam processamento de suas máquinas, quando estas estão ociosas, para a execução de aplicações de terceiros, geralmente associadas a algum projeto científico com um forte apelo, que serve de incentivo para os voluntários doarem o processamento de suas máquinas. Porém, neste modelo de grade, os voluntários não são capazes de executar suas tarefas, apenas sendo capazes de doar seus recursos.

Um dos grandes problemas enfrentados por vários tipos de grade computacional é a dificuldade na negociação entre as diversas entidades envolvidas para que a grade seja composta e, posteriormente, administrada. Geralmente, esta negociação envolve pessoas e contratos, tornando o processo lento e complexo, limitando a escala do sistema [4]. As grades computacionais entre pares (ou P2P, do inglês *peer-to-peer*) surgiram como uma alternativa que facilita a implantação e gerência das grades. Neste modelo, cada participante do sistema pode atuar tanto como provedor quanto consumidor de recursos. O OurGrid [17] é um exemplo de middleware para grades computacionais P2P abertas, em que não há negociação humana para a entrada de novos participantes no sistema.

Como as grades P2P se baseiam no comportamento de reciprocidade entre seus participantes, mecanismos de incentivo são usados para motivar os usuários a doarem seus recursos ociosos e promover a colaboração entre os pares[2] do sistema. O OurGrid utiliza como mecanismo de incentivo um esquema de reputação autônomo chamado Rede de Favores – ou *Network of Favors* (NoF) [4].

A NoF é um mecanismo de incentivo para a doação de recursos em grades computacionais P2P baseado na reciprocidade entre os pares da grade. Cada nó da grade decide para quem doar seus recursos baseado no passado de interações com os outros nós. Os nós associam um crédito a cada um dos seus pares, com os quais eles interagiram anteriormente, que é atualizado a cada troca de favores[3]. Quando há doação de recursos entre dois nós, o nó consumidor incrementa o crédito que ele associa ao nó doador, e o nó doador decrementa o crédito que ele associa ao nó consumidor. Nos momentos em que a grade P2P está com uma alta contenção de recursos, ou seja, a demanda é maior que oferta, os nós doares precisam escolher para quais nós consumidores os seus recursos serão destinados. A ideia na NoF

[2]Chamamos de pares (ou *peers*) os participantes (ou nós) de um sistema P2P.

[3]Chamamos de favores as doações de recursos entre os pares: um nó que doa um recurso, faz um favor para o nó que usa os recursos.

é que a divisão de recursos seja proporcional ao crédito que o nó doador associa aos nós que estão requisitando recursos em um certo instante. Desta forma, os nós que mais doaram no passado e possuem maiores créditos terão preferência quando requisitarem recursos. Atualmente, este é o mecanismo de incentivo utilizado no middleware OurGrid [4].

Mesmo com a utilização de mecanismos de incentivo, as grades P2P ainda possuem como característica a variação significativa, ao longo do tempo, de vários atributos associados à qualidade de serviço (ou QoS, do inglês *Quality of Service*), como a disponibilidade de recursos ou tempo de resposta da execução de tarefas. Os fatos dos recursos não serem dedicados e da demanda não ser conhecida previamente trazem incertezas à QoS oferecida pela grade. Além disso, os nós geralmente competem por recursos, o que faz com que o comportamento dos outros nós também influenciem na QoS. Recursos de um nó que estavam ociosos e estavam sendo doados para a grade podem se tornar indisponíveis a qualquer momento, caso o dono dos recursos volte a utilizá-los. A indisponibilidade de recursos também pode ser causada por novas requisições feitas pelos nós do sistema, fazendo com que recursos que estavam sendo doados para um certo nó sejam desalocados e então realocados para outros nós, de acordo com as políticas de priorização do gerenciamento de recursos e do mecanismo de incentivo.

Se, por um lado, é aceitável a incerteza da QoS das grades P2P em troca dos seus baixos custos operacionais, por outro lado, existem várias ocasiões em que ter alguma estimativa da QoS fornecida é bastante importante. Este caso se aplica, por exemplo, a usuários que precisam executar aplicações que possuem restrições de tempo, tendo prazos para iniciar e/ou finalizar suas execuções. Exemplos deste tipo de aplicação incluem: medicina remota [13]; controle e monitoramento de áreas de risco, instrumentos e tráfego aéreo [40; 29; 56]; processamento e transmissão multimídia [37; 47]; predição de fenômenos naturais [45; 8; 14]; e física nuclear [46; 50].

Um outro exemplo de aplicação que depende de estimativas da QoS da grade P2P foi apresentado por Maciel Jr. et al. [39], que propõe o planejamento de uma infra-estrutura híbrida de TI, composta por máquinas locais dedicadas, recursos computacionais adquiridos de provedores de computação na nuvem (do inglês *cloud computing*) e recursos obtidos em uma grade P2P. Uma das maiores dificuldades no planejamento da capacidade dessa infra-estrutura está na incerteza na quantidade de recursos que podem ser obtidos periodicamente

da grade P2P. Naquele trabalho, conclui-se que uma predição da qualidade de serviço oferecida pela grade P2P é importante para que o planejamento da capacidade dessa infra-estrutura híbrida seja feito de forma eficiente.

Para que as grades P2P possam ser utilizadas por usuários que possuem aplicações com restrições de tempo, é necessário que se tenha uma estimativa da qualidade de serviço a ser oferecida pela grade. Por exemplo, é desejável uma estimativa da capacidade de processamento que estará disponível para um usuário na grade em períodos de tempo futuros. Além disso, uma vez que os recursos sejam obtidos pelo usuário, deseja-se saber por quanto tempo os recursos permanecerão disponíveis para ele.

Tendo em vista a alta variação em atributos de qualidade de serviço e a falta de estimativas desses atributos para os usuários, são apresentados na próxima seção os objetivos deste trabalho, que procuram atacar estes problemas.

1.2 Objetivos

O objetivo geral deste trabalho é propor e avaliar modelos de predição que forneçam estimativas da qualidade de serviço oferecida por uma grade P2P, a partir do conhecimento do sistema em questão e de dados históricos. Mais especificamente, pretende-se estimar a capacidade de processamento que estará disponível para um determinado nó da grade em períodos de tempo futuros, em uma grade P2P que utiliza a Rede de Favores como mecanismo de incentivo.

Para avaliar os modelos de predição propostos, foi adotada como metodologia a simulação de ambientes de grades computacionais P2P. Para simular este tipo de ambiente, é necessário especificar como entrada uma carga de trabalho que descreve o comportamento dos usuários e as características de suas aplicações que irão executar na grade.

Porém, uma dificuldade encontrada em avaliar mecanismos associados a grades P2P é que não há conhecimento de sistemas deste tipo que atingiram uma alta utilização. Apesar da existência de grades computacionais P2P em produção há alguns anos (e.g. Comunidade OurGrid [17]), estas grades não conseguiram ainda atingir uma larga escala de uso, como se espera que este tipo de sistema colaborativo tenha. Consequentemente, não há rastros de sistemas que sejam compatíveis com a escala global que uma grade P2P se propõe a

dar suporte. A falta de registros de cenários "reais" de uso dificulta a avaliação de novas propostas como as de escalonadores, modelos de predição, políticas de alocação de recursos, etc.

Normalmente, duas abordagens são usadas para gerar cenários de carga de trabalho para simulação de ambientes de grades P2P: (i) a varredura de parâmetros de diversos atributos utilizando as distribuições de probabilidade mais conhecidas (e.g. normal, uniforme e exponencial) e (ii) o uso de modelos ou rastros de carga de trabalho extraídos de outros sistemas de computação paralela.

A primeira abordagem consiste em realizar uma varredura de parâmetros em atributos de carga de trabalho relacionados às características de aplicações e da demanda que chega no sistema [3; 43; 7; 18]. O problema dessa abordagem é que, geralmente, as distribuições de probabilidade e os valores dos parâmetros usados não são bem justificados ou validados com dados de sistemas reais, não sabendo então se os cenários são realistas.

A outra abordagem consiste em aplicar modelos de carga de trabalho extraídos de rastros de outros sistemas de computação de alto desempenho em produção (e.g. grades de serviço [28], *clusters* [38] ou supercomputadores [16]) disponíveis em repositórios como o Grid Workload Archive (GWA [27; 54]) e o Parallel Workload Archive (PWA [22]) na simulação de cenários de grades P2P [10; 11; 15]. Apesar de existirem semelhanças entre as características desses tipos de infraestrutura de computação, também existem algumas diferenças devido ao tipo dos recursos que compõem cada um desses sistemas e, consequentemente, da qualidade de serviço associada a cada um. Enquanto em *clusters* e grades de serviço as máquinas são dedicadas para executar as aplicações dos usuários, nas grades P2P as máquinas são usadas de maneira oportunista, só ficando disponíveis para os usuários da grade quando os proprietários das máquinas não estiverem usando-as, ou seja, quando as máquinas estiverem ociosas. Isto faz com que o perfil de uso e as características das aplicações de cada sistema sejam diferentes, podendo tornar não representativo o uso direto de rastros ou modelos de outros tipos de sistema para gerar cenários de simulação para grades computacionais P2P.

Levando em consideração os problemas apresentados, os objetivos específicos deste trabalho são:

1. Propor um modelo de predição para a capacidade de processamento disponível para os

nós da grade em períodos de tempo futuros

2. Propor um modelo de geração de carga sintética para avaliação de ambientes de grades P2P

3. Avaliar o modelo de predição utilizando carga sintética gerada pelo modelo proposto em (2), e também utilizando rastros de outros tipos de sistemas de computação paralela

Para o objetivo (1), foi elaborado um modelo de predição para a grade P2P que leva em consideração o conhecimento do sistema e a obtenção de informações do estado da grade. Formalizou-se o problema em questão e também como é feito o gerenciamento de recursos do sistema. A partir disto, definiu-se o modelo de predição baseado nas características do sistema. Também foram utilizados modelos de predição caixa-preta encontrados na literatura, e adaptados ao problema em questão.

Para o objetivo (2), foi elaborado um modelo de geração de carga de trabalho sintética para grades P2P, que é extraído a partir do agrupamento de usuários e aplicações que possuem características similares; e um modelo foi gerado para cada grupo encontrado.

Para o objetivo (3), foi utilizado o modelo de geração proposto em (2) para simular um ambiente de grade P2P realista e avaliar os modelos de predição propostos. Além disso, foram utilizados rastros de outros sistemas de computação paralela existentes, para simular a grade e avaliar as predições em cenários semelhantes a de outros sistemas em produção.

1.3 Estrutura do Documento

Esta dissertação é organizada da seguinte forma: no Capítulo 2 é apresentada a revisão bibliográfica; no Capítulo 3 é apresentado nosso modelo de predição baseado no conhecimento do sistema, para a capacidade de processamento oferecida pela grade para um certo nó do sistema, além de estratégias de predição caixa-preta existentes na literatura, adaptadas ao nosso problema; no Capítulo 4 é apresentado nosso modelo para geração de carga sintética representativa para simulação de ambientes de grades computacionais P2P; no Capítulo 5 é apresentada a avaliação dos modelos de predição, mostrando resultados do erro dos modelos através de simulação com carga de trabalho sintética e com rastros de sistemas existentes; e no Capítulo 6 são apresentadas as conclusões e trabalhos futuros.

Capítulo 2

Revisão Bibliográfica

2.1 Predição da Qualidade de Serviço em Grades P2P

Dos trabalhos encontrados, o atributo de QoS mais analisado anteriormente foi a disponibilidade de recursos das grades computacionais. Este atributo foi explorado em diversos trabalhos, principalmente nos ambientes de grades desktop de larga escala usadas em computação voluntária [6; 32] e grades desktop corporativas [12; 42; 33; 24]. No entanto, nenhum trabalho foi encontrado em predição de disponibilidade de recursos em grades P2P.

Andrzejak et al. propõem um modelo de predição que estima a quantidade de recursos disponíveis em uma grade desktop de computação voluntária em intervalos de tempo futuros [6]. O modelo é implementado como um classificador Naive Bayes e é avaliado através de simulações usando dados empíricos disponíveis em *traces* de uma grade composta por uma grande quantidade de recursos voláteis de computação voluntária.

Kondo et al. usam o algoritmo de agrupamento *k-means* para detectar padrões de disponibilidade entre recursos computacionais de computação voluntária, e aplica os resultados do agrupamento para selecionar recursos com disponibilidade correlacionada para a execução de tarefas dependentes [32].

No entanto, nos modelos de computação voluntária avaliados, os recursos são doados para a grade quando ociosos, mas os usuários que doam seus recursos não são capazes de executar suas aplicações na grade. Os recursos são doados de forma altruísta para que aplicações de alguma(s) entidade(s) sejam executados nesses recursos voluntários. Desta forma, o modelo de predição estima a quantidade de recursos disponíveis na grade como um todo. Em

grades P2P, os nós no sistema podem atuar tanto como provedores quanto como consumidores. Além disso, os consumidores podem ter disponíveis na grade quantidades diferentes de recursos entre eles, de acordo com o mecanismo de incentivo empregado e o histórico de interações deles com os outros participantes da grade. Portanto, modelos de predição de disponibilidade em grades P2P devem considerar a disponibilidade de recursos para cada nó do sistema, assim como levar em consideração o gerenciamento de recursos e o mecanismo de incentivo utilizado na grade.

Nurmi et al. apresentam uma modelagem da disponibilidade de recursos em grades desktop corporativas e também em recursos de computação voluntária de larga escala [42]. Eles defendem que as distribuições hyper-exponencial e Weibull representam bem a disponibilidade de recursos para esses ambientes.

Brevik at al. analisam modelos de predição para a duração da disponibilidade de recursos em grades desktop usando um modelo paramétrico (fazendo ajuste de dados empíricos com a distribuição Weibull) e dois modelos não-paramétricos (*resample* e método binomial), e mostram que os melhores resultados são obtidos com o método não-paramétrico binomial [12].

Kondo et al. caracterizam a disponibilidade de recursos em grades desktop corporativas e classificam a disponibilidade em três tipos [33]:

- *Host availability* - identifica quando um nó está atingível. Possíveis causas de indisponibilidade incluem falha elétrica, defeito, desligamento ou reinicialização de máquina.

- *Task execution availability* - indica quando uma tarefa pode ou não ser executada em um nó, de acordo com a política empregada na máquina a ser usada. Possíveis causas de indisponibilidade incluem uso do teclado/mouse, ou uso da CPU pelo usuário local da máquina acima de um certo limite.

- *CPU availability* - Fração da CPU que pode ser usada por uma aplicação executada na grade desktop. Entre os fatores que a afetam estão a carga de CPU usada pelo sistema e a usada por processos do usuário local.

Os trabalhos anteriores pesquisados que estudam a predição de disponibilidade de recursos em grades desktop cobrem no máximo esses três tipos de disponibilidade. Porém, em

grades P2P outras causas de indisponibilidade também têm que ser levadas em consideração. O mecanismo de incentivo tem que ser considerado para que seja feita uma estimativa da quantidade de recursos que um nó estará apto a receber de cada participante da grade, de acordo com o histórico das interações entre os nós. Além disso, as máquinas podem se tornar indisponíveis caso algum nó com maior prioridade passe a requisitar recursos da grade. Portanto, a demanda na grade também é um fator que deve ser considerado na predição da disponibilidade de recursos em grades P2P. Esses aspectos não são considerados em nenhum dos trabalhos pesquisados, e serão abordados neste trabalho.

2.2 Rede de Favores

A Rede de Favores – ou *Network of Favors* (NoF) [4], proposta por Andrade et al., é um mecanismo de incentivo que usa um esquema de reputação autônomo no qual um nó decide para quem doar seus recursos ociosos baseado no histórico das suas interações com os pares que estão solicitando recursos à grade. A ideia é que os nós que mais doaram recursos no passado para um determinado nó sejam priorizados quando requisitarem recursos desse nó.

Em uma primeira avaliação, foram apresentados por Andrade et. al. indícios de que a solução simples apresentada pela NoF é eficiente ao priorizar os nós que mais colaboram com o sistema. Além disso, mostrou-se que favores tendem a ser retribuídos em algum tempo no futuro, desde que haja demanda [4].

Em um outro trabalho, a NoF é avaliada em uma grade P2P com a presença de *free-riders*[1] e de nós que aplicam ataques de troca de identidade (*whitewash attacks*), no qual *free-riders* saem e entram do sistema com novas identidades para zerar o seu débito com os colaboradores [2]. Os resultados mostraram que, com um técnica simples de limitar o crédito entre os nós somente a valores não negativos, a NoF é eficiente em marginalizar os *free-riders*, fazendo com que colaboradores sejam priorizados, além de também mostrar-se eficaz contra ataques de troca de identidade.

Além disso, Andrade et al. analisaram quais os cenários em que compensa um nó colaborador com o sistema e os cenários em que vale mais a pena ser um *free-rider* [5]. Concluiu-se que os mecanismos de incentivo baseados em reputação são ineficazes em marginalizar

[1] *Free-riders* são nós que não colaboram com o sistema e só consomem recursos.

free-riders quando a oferta de recursos é maior que a demanda. Porém, os colaboradores continuam a ser priorizados. A NoF foi comparada com um modelo de reputação ideal, onde colaboradores são sempre priorizados, e mostrou resultados bem próximos do ótimo para a maioria dos cenários avaliados.

Uma avaliação mais detalhada foi feita posteriormente por Andrade et al., comparando a NoF com um mecanismo de reputação ideal que prioriza os colaboradores e marginaliza *free-riders* [3]. Naquele trabalho, a NoF também foi avaliada com experimentos de medição, utilizando a implementação do mecanismo de incentivo no middleware OurGrid. Os resultados mostraram que a solução simples apresentada pela NoF apresenta resultados próximos aos de um modelo ideal para a maioria dos cenários, mostrando que a NoF é eficaz em priorizar colaboradores.

Porém, nos trabalhos anteriores sobre a NoF, argumenta-se que as aplicações que rodam em grades P2P que usam a NoF como mecanismo de incentivo não devem exigir garantias de qualidade de serviço, já que a grade é *best-effort*. Não são propostos mecanismos para se prever a QoS da grade, apenas constata-se que a QoS varia de acordo com o quanto um nó do sistema colabora. No trabalho proposto, pretende-se fornecer estimativas da qualidade de serviço da grade P2P. Desta forma, o uso da grade P2P poderá ser estendido para aplicações que tenham requisitos fracos de QoS, de modo que os erros da predição dos atributos de QoS sejam toleráveis.

2.3 Geração de Carga de Trabalho Sintética para Grades P2P

Uma característica presente nos geradores de carga de trabalho sintética existentes para ambientes de computação paralela é que a demanda que chega no sistema é proveniente de um único modelo. Geralmente, cada atributo é modelado usando apenas uma distribuição de probabilidade e seus parâmetros associados.

Lublin e Feitelson propuseram um modelo de carga de trabalho para computação de alto desempenho que utiliza atributos como *intervalo entre chegada de jobs no sistema*, *número de processadores alocados por job* e *tempo de execução de jobs* [38]. Esta demanda é modelada como uma caixa-preta, não havendo uma associação da carga do sistema com

os usuários que a geram. Porém, em alguns sistemas, é necessário que a carga de trabalho esteja associada a usuários que a geram, que é o caso da maioria das avaliações feitas em ambientes de grades P2P. Além disso, defende-se que, ao invés de utilizar um único modelo para gerar a carga do sistema, é mais adequado utilizar modelos hierárquicos baseados no comportamento dos usuários do sistema [57; 48; 23]. A ideia é que a carga de trabalho do sistema seja gerada a partir da agregação da demanda de diversos usuários do sistema, modelados separadamente de acordo com o perfil de cada um.

O modelo de carga de trabalho em grades computacionais proposto por Iosup et al. é feito para aplicações do tipo Bag-of-Tasks (BoT), onde um job é composto por um conjunto de tarefas independentes [28]. O modelo é extraído a partir de rastros de grades computacionais de serviço, que estão armazenados no Grid Workloads Archive [27]. Um problema em usar este modelo para uma grade P2P é que as aplicações de grades de serviço geralmente possuem características diferentes de aplicações de grades oportunistas P2P. Um outro problema presente neste modelo é que a carga de trabalho não é alterada à medida que novos usuários são adicionados ao sistema, pois assume-se que a carga gerada pelo sistema é a mesma independente do número de usuários existentes, sendo então essa carga dividida pela quantidade de usuários especificada. Além disso, o comportamento dos usuários e características das aplicações são modelados a partir de uma única função de probabilidade para cada rastro, o que impossibilita uma análise dos diferentes perfis de usuários e aplicações que cada sistema possui.

Neste trabalho, é proposto um modelo de geração de carga de trabalho sintética representativo para grades P2P. O modelo é do tipo hierárquico [23], baseado no comportamento dos usuários, onde usuários e aplicações com características similares são agrupados e vários modelos são criados para cada grupo. Como não existem rastros de grades P2P com larga utilização, utilizou-se para a extração do modelo rastros de outros sistemas de computação paralela, identificando qual tipo de sistema é mais adequado para cada atributo de carga de trabalho.

Capítulo 3

Predição da Qualidade de Serviço em uma Grade P2P

Neste capítulo são descritas as estratégias de predição utilizadas neste trabalho para estimar a capacidade de processamento obtida pelos nós da grade em instantes de tempo futuros. As estratégias de predição apresentadas podem ser divididas em: (i) caixa-branca, quando o modelo de predição é elaborado a partir de conhecimento do funcionamento interno do sistema, (ii) caixa-preta, quando a técnica de predição não conhece o sistema em questão e considera apenas dados históricos ou (ii) caixa-cinza, quando os dois tipos de técnica são utilizadas em conjunto, tanto o conhecimento do comportamento interno do sistema quanto o uso de técnicas estatísticas em dados históricos.

Um modelo de predição caixa-branca é proposto na Seção 3.1, com base no conhecimento interno da grade computacional P2P e da sua gerência de recursos que é guiada pelo mecanismo de incentivo da Rede de Favores. Na Seção 3.2, são apresentadas 4 estratégias de predição caixa-preta, obtidas da literatura e adaptadas ao contexto da predição da capacidade de processamento oferecida pela grade de computação P2P aos nós do sistema. Finalmente, na Seção 3.3 é apresentado um modelo caixa-cinza que combina a técnica caixa-branca com técnicas caixa-preta.

3.1 Estratégia de Predição Caixa-Branca

A estratégia de predição caixa-branca proposta, chamada de *NoF-Based*, utiliza conhecimento do sistema para estimar a capacidade de processamento que um nó da grade é capaz de obter em janelas de tempo futuras. Primeiro apresentamos uma formalização do modelo do sistema, que servirá como base para o modelo de predição apresentado em seguida.

3.1.1 Modelo do Sistema

O modelo de predição proposto neste trabalho foi elaborado para ser utilizado por um nó ℓ participante da grade P2P, chamado de nó local, que deseja estimar a capacidade de processamento que será obtida da grade P2P em períodos de tempo futuros.

Na perspectiva do nó local ℓ, a grade é composta por um conjunto de nós (ou *peers*) remotos $\mathcal{G} = \{p_1, p_2, \cdots, p_{|\mathcal{G}|}\}$, onde $|\mathcal{G}|$ é a quantidade de nós remotos participantes da grade, sem contar com o nó local ℓ. Cada nó pode estar em um dos dois estados possíveis em um certo instante: consumidor (quando este requisita recursos da grade) ou doador (quando este tem recursos disponíveis para doar para os demais participantes da grade). A grade P2P analisada utiliza a Rede de Favores (ou NoF, do inglês *Network of Favors*) como mecanismo de incentivo, que usa informação do histórico das interações entre os nós da grade para decidir como cada recurso é alocado.

Considerando dois nós do sistema, p_d e p_c, definimos b_d^c como o balanço do crédito de favores que o nó p_c associa ao nó p_d, e b_c^d como o balanço do crédito que o nó p_d associa ao nó p_c. Inicialmente, b_d^c e b_c^d são iguais a zero. Ao passo que p_d e p_c interagem, ambos os créditos são atualizados. Quando p_c consome recursos de p_d, o crédito que o consumidor associa ao doador b_d^c aumenta proporcionalmente à quantidade doada de recursos, enquanto o crédito que o doador associa ao consumidor b_c^d diminui de acordo com a mesma quantidade, até um minimo de zero (isto garante que o crédito nunca é negativo, evitando assim o ataque conhecido com "troca de identidade", ou *whitewash*, em sistemas de reputação [3]).

De acordo com a NoF, os nós doadores distribuem seus recursos ociosos de maneira proporcional ao balanço de crédito dos nós que estão requisitando os recursos. Por exemplo, se o nó p_d possui 100 recursos disponíveis, os nós p_{c1}, p_{c2} e p_{c3} são os únicos consumidores e seus créditos na perspectiva de p_d são, respectivamente, $b_{c1}^d = 0$, $b_{c2}^d = 3$ e $b_{c3}^d = 7$,

então os nós p_{c1}, p_{c2} e p_{c3} obterão 0, 30 e 70 recursos, respectivamente. Caso todos os nós consumidores possuam crédito zero, então os recursos são distribuídos de forma igualitária entre eles.

A metodologia de avaliação usada neste trabalho é baseada na execução de simulações de uma grade P2P. No modelo de simulação, o tempo é discretizado em T turnos. No começo de cada turno, os nós que possuem recursos ociosos (provedores) distribuem estes recursos entre os nós consumidores. A contabilidade dos recursos doados é realizada ao final de cada turno, através da atualização dos créditos entre consumidores e provedores, que é proporcional à quantidade doada de recursos em cada interação ocorrida no respectivo turno.

Considerou-se que os jobs submetidos para execução na grade P2P são do tipo *Bag-of-Tasks* (BoT), ou seja, são aplicações paralelas formadas por um conjunto de tarefas independentes [17; 28]. Além disso, considerou-se que cada tarefa utiliza apenas um recurso (máquina) por turno. Cada tarefa possui uma demanda de processamento associada, que corresponde ao número de turnos necessários para executá-la em uma "máquina de referência".

Cada nó do sistema possui um conjunto de máquinas que são disponibilizados para a grade, onde cada uma possui um poder computacional associado. Portanto, o nó que participa da grade P2P funciona como um gerente dos recursos locais disponíveis em um domínio administrativo da grade. O conjunto que representa as máquinas de um nó i da grade é dado por $\mathcal{M}_i = \{m_{i,1}, m_{i,2}, \ldots, m_{i,|M_i|}\}$, onde $\forall i, 1 \leq i \leq |\mathcal{G}|, \forall j, 1 \leq j \leq |\mathcal{M}_i|$, sendo $m_{i,j}$ o poder de processamento da j-ésima máquina do nó i e $|\mathcal{M}_i|$ a quantidade de máquinas que este nó possui. Neste caso, o poder de processamento de uma máquina é representado como uma medida de unidades de processamento por turno.

Em cada turno t, um nó doador aloca suas máquinas para nós consumidores. O conjunto de alocações das máquinas de um nó doador i no turno t para o nó local ℓ é representado por $\mathcal{A}_i^t = \{a_{i,1}^t, a_{i,2}^t, \ldots, a_{i,|\mathcal{M}_i|}^t\}$, com $a_{i,j}^t \in \{0,1\}$ $\forall i, 1 \leq i \leq |\mathcal{G}|, \forall j, 1 \leq j \leq |\mathcal{M}_i|$. Os valores de $a_{i,j}^t$ são definidos da seguinte forma:

$$a_{i,j}^t = \begin{cases} 1 & \text{, se a máquina } j \text{ do nó } i \text{ está alocada para o nó local } \ell \text{ no turno } t \\ 0 & \text{, caso contrário.} \end{cases}$$

A predição é realizada em períodos de tempo do futuro, chamados de janela de predição. O turno inicial da janela é chamado de t_c, que é o turno no qual o nó local ℓ obtém informação

do estado da grade, e t_p é o turno em que a janela de predição termina. Definimos $\Delta = [t_c, t_p]$ como sendo a janela de predição, para a qual ℓ fará a predição.

Portanto, a capacidade total de processamento que o nó local ℓ irá obter da grade na janela de tempo Δ será:

$$\mathcal{C}(\Delta) = \sum_{t=t_c}^{t_p} \sum_{i=1}^{|\mathcal{G}|} \sum_{j=1}^{|\mathcal{M}_i|} m_{i,j} \cdot a_{i,j}^t \tag{3.1}$$

Com isto, pretende-se estimar o poder de processamento $\mathcal{C}(\Delta)$ que o nó local ℓ irá obter da grade P2P durante janelas de tempo $\Delta = [t_c, t_p]$, com informações da grade obtidas no turno t_c.

3.1.2 Modelo de Predição

Com base no problema apresentado, é proposto um modelo de predição para estimar a capacidade de processamento que o nó local é capaz de obter da grade P2P em janelas de tempo futuras. O modelo utiliza informações obtidas da grade para realizar suas estimativas. A descrição do modelo é apresentada a seguir.

Chamamos de $\mathcal{G}_c(t)$ e $\mathcal{G}_d(t)$ os sub-conjuntos dos nós na grade $\mathcal{G}$ que representam, respectivamente, os nós consumidores e os nós doadores no turno t. $B_d(t)$ é o somatório do crédito que o provedor p_d associa a todos os consumidores - exceto o nó local ℓ - no início do turno t, e é definido como:

$$B_d(t) = \sum_{\forall p_c \in \mathcal{G}_c(t)} b_c^d(t)$$

onde $b_c^d(t)$ é o balanço do crédito que o nó p_d associa ao nó p_c no começo do turno t.

Como resultado da NoF, um nó provedor p_d distribui seus recursos disponíveis no turno t, $r_d(t)$, de maneira proporcional ao crédito de cada nó consumidor no sub-conjunto $\mathcal{G}_c(t) \cup \{\ell\}$, a menos que o crédito agregado de todos os consumidores (incluindo ℓ) para o provedor p_d no turno t seja zero, onde neste caso os recursos serão igualmente distribuídos entre todos os nós consumidores. Assim, se o nó local ℓ for um consumidor de recursos no turno t, a quantidade estimada de recursos disponíveis do nó doador p_d para o nó local ℓ no turno t

será:

$$R_d(t) = \begin{cases} \lfloor \frac{b_\ell^d(t)}{B_d(t)+b_\ell^d(t)} \cdot r_d(t) \rfloor & \text{, se } B_d(t) + b_\ell^d(t) > 0 \\ \\ \lfloor \frac{r_d(t)}{|\mathcal{G}_c(t)|+1} \rfloor & \text{, caso contrário.} \end{cases} \tag{3.2}$$

Note que, em cenários de baixa contenção, alguns nós consumidores poderão não requisitar todo o crédito de recursos que eles estarão aptos a receber da grade. Neste caso, haverão recursos sobressalentes a serem redistribuídos entre os outros nós consumidores que necessitam mais do que poderiam receber do sistema. Como os recursos sobressalentes estarão eventualmente disponíveis e poderão ser alocados para o nó local, a Equação 3.2 resulta num limite inferior para a quantidade de recursos disponíveis para ℓ de cada nó no sistema. Além disso, o nó local também pode não utilizar todos os recursos disponibilizados para ele, quando o número de tarefas que o nó local deseja executar for menor do que a quantidade de recursos disponíveis. Desta forma, a quantidade de máquinas que o nó local irá utilizar da grade será limitada pela quantidade de máquinas que ele requisita no momento, ou seja, será limitada pela quantidade de tarefas do nó local a serem executadas. Então, uma estimativa para a quantidade de recursos que o nó local ℓ pode receber de todos os nós da grade no turno t é dada por:

$$R(t) = \begin{cases} \sum_{\forall p_d \in \mathcal{G}_d(t)} R_d(t) & \text{, se } D(t) > \sum_{\forall p_d \in \mathcal{G}_d(t)} R_d(t) \\ \\ D(t) & \text{, caso contrário.} \end{cases} \tag{3.3}$$

onde $D(t)$ é a quantidade de recursos requisitados pelo nó local no turno t.

Podemos estimar a quantidade de recursos disponíveis na janela de predição Δ para o nó local ℓ, com informação obtida no turno t_c, a partir das Equações 3.2 e 3.3. Porém, necessita-se dos valores para os parâmetros dessas equações nos turnos $t \in [t_c, t_p]$, quais sejam: o conjunto de provedores no sistema – $\mathcal{G}_d(t)$; o conjunto de consumidores no sistema – $\mathcal{G}_c(t)$; a quantidade de recursos disponíveis em cada nó doador – $r_d(t), \forall p_d \in \mathcal{G}_d(t)$; o crédito que cada nó doador associa ao nó local – $b_\ell^d(t), \forall p_d \in \mathcal{G}_d(t)$; e o crédito agregado que cada nó doador associa aos nós consumidores – $B_d(t), \forall p_d \in \mathcal{G}_d(t)$.

No contexto das grades computacionais, é comum alguns *middlewares* proverem informações sobre o status da grade através do monitoramento de recursos e serviços de informa-

ção da grade (GIS - *Grid Information Services*) [9] [20] [51] [19]. Em particular, o OurGrid [17], que é um *middleware* para grades computacionais P2P que implementa a NoF como mecanismo de incentivo, provê esta informação através de um GIS distribuído [9] e também em um *website* [44]. Alguns problemas podem surgir dos sistemas de informação em ambientes P2P, como por exemplo: presença de informações não confiáveis ou área de cobertura limitada [1]. Neste trabalho, porém, assumimos que todos os nós da grade reportam informações corretas para os GIS, que por sua vez, alcança toda a cobertura do sistema. Assim, todos os parâmetros necessários são conhecidos para o turno atual t_c, mas não para os outros turnos futuros da janela de predição, i.e. $t \in (t_c, t_p]$.

Os valores dos parâmetros em t_c são utilizados como uma estimativa para os valores dos mesmos parâmetros no intervalo $(t_c, t_p]$. Por simplicidade, no modelo de predição, estimou-se que $B_d(t)$ e $r_d(t)$ são constantes no intervalo de tempo $[t_c, t_p]$ (ou seja, $B_d(t) = B_d(t_c)$ e $r_d(t) = r_d(t_c)$, $\forall t, t_c \leq t \leq t_p$).

Como foi explicado na Seção 3.1.1, quando os recursos doados por um nó são utilizados por outro nó, os balanços dos créditos de favores do consumidor e provedor são atualizados. Assumiu-se que, no turno t_c, o crédito do nó local ℓ em cada nó doador p_d é conhecido, ou seja, $b^d_\ell(t_c), \forall p_d, p_d \in G_d(t_c)$. Assim, uma estimativa da atualização do crédito que o nó doador p_d associa ao nó local ℓ no turno t é dada pela seguinte relação de recorrência, $\forall t, t \geq t_c$, com parada em $b^d_\ell(t_c)$ que é conhecido:

$$b^d_\ell(t) = \begin{cases} 0 & \text{, se } b^d_\ell(t-1) - R^\ell_d(t-1) \leq 0 \\ \\ b^d_\ell(t-1) - R^\ell_d(t-1) & \text{, caso contrário.} \end{cases} \tag{3.4}$$

onde $R^\ell_d(t-1)$ é a quantidade estimada de recursos que p_d doara para ℓ no turno $t-1$.

Porém, como dito anteriormente, pretende-se estimar a capacidade total de processamento que o nó local ℓ irá obter da grade P2P em janelas de tempo. Assumimos que é possível obter da grade informação da média da capacidade de processamento de todas as máquinas que compõe o sistema, definida por:

$$\overline{m} = \frac{\sum_{i=1}^{|\mathcal{G}|} \sum_{j=1}^{|\mathcal{M}_i|} m_{i,j}}{\sum_{i=1}^{|\mathcal{G}|} |\mathcal{M}_i|}$$

Como não é possível saber com antecedência para quais máquinas as tarefas do nó local

ℓ serão alocadas, e por consequência não é possível saber a capacidade de processamento que cada tarefa terá disponível, o modelo de predição considera que cada máquina possui capacidade média de processamento $\overline{m}$.

Com isto, a estimativa da capacidade total de processamento que o nó local ℓ irá obter da grade P2P em um janela de tempo $\Delta = [t_c, t_p]$ será:

$$\Psi(\Delta) = \sum_{t=t_c}^{t_p} R(t) \cdot \overline{m} \tag{3.5}$$

onde $R(t)$ é a estimativa da quantidade de máquinas que o nó local irá obter da grade no turno t, definida pela Equação 3.3, e $\overline{m}$ é a média da capacidade de processamento de todas as máquinas da grade.

Desta forma, a Equação 3.5 é usada para estimar o poder de processamento obtido da grade a partir do turno atual t_c até um tempo futuro t_p, com informações da grade obtidas no turno t_c.

3.2 Estratégias de Predição Caixa-Preta

As estratégias de predição caixa-preta utilizam dados históricos da capacidade de processamento, obtida pelos nós em instantes de tempo do passado, para estimar a capacidade que um nó do sistema irá obter no futuro. As seguintes estratégias, adaptadas do trabalho de Sonmez at el. [49], foram usadas:

- **Constant.Obtained:** a predição é feita considerando que a capacidade de processamento obtida no turno atual t_c irá se manter constante durante toda a janela de predição Δ. Esta técnica é uma adaptação do método de predição *Last* apresentado por Sonmez et al. [49], aplicado em um outro contexto.

- **Constant.OR:** neste método, primeiro é verificado para o turno atual t_c qual a proporção de tarefas submetidas que foram alocadas para alguma máquina (chamamos esta proporção de *Obtained Ratio*, ou OR). Então, considera-se que durante toda a janela de predição Δ, a mesma proporção de tarefas será alocada, e que todas as máquinas da grade possuem capacidade de processamento igual à capacidade média $\overline{m}$.

- **RunningMean.Obtained:** a predição é feita utilizando a média das capacidades de processamento que foram obtidas em momentos passados para janelas de predição do mesmo tamanho de Δ. Esta técnica é uma adaptação do método de predição *Running Mean* apresentado por Sonmez et al. [49], aplicado em um outro contexto.

- **RunningMean.OR:** neste método, primeiro obtém-se a média da proporção de tarefas submetidas que foram alocadas para alguma máquina, para janelas de predição do passado que são do mesmo tamanho de Δ. Então, considera-se que a proporção de tarefas que serão alocadas será igual a esta média, e que todas as máquinas da grade possuem capacidade de processamento igual à capacidade média $\overline{m}$.

3.3 Estratégias de Predição Caixa-Cinza

A estratégia de predição caixa-cinza proposta, chamada *NoF-Based.CF*, utiliza o modelo de predição caixa-branca, baseado no conhecimento do sistema, apresentando na Seção 3.1, e combina com técnicas caixa-preta que utilizam dados históricos para fazer estimativas da capacidade de processamento.

O prefixo *NoF-Based* do nome é dado por ela ser uma extensão do modelo de predição com o mesmo nome, apresentado na Seção 3.1. O sufixo *CF* vem de *Correction Factor*, pois é aplicado um fator de correção no modelo baseado nos erros obtidos no passado. A estimativa deste modelo para a capacidade total de processamento que o nó local irá obter da grade P2P em um janela de tempo Δ será:

$$\Psi_{cf}(\Delta) = \Psi(\Delta) \cdot CF_{|\Delta|}$$

onde $\Psi(\Delta)$ é a estimativa gerada pelo modelo caixa-branca apresentado na Seção 3.1, $|\Delta|$ é o tamanho da janela Δ e $CF_{|\Delta|}$ é o fator de correção do modelo para janelas de tamanho iguais aos de Δ.

O fator de correção $CF_{|\Delta|}$ inicialmente é 1 para todos os tamanhos de janela $|\Delta|$, ou seja, a primeira estimativa do modelo será igual à estimativa do modelo caixa-branca para qualquer janela de predição. À medida que predições são realizadas para o nó local, $CF_{|\Delta|}$ é atualizado de acordo com o erro da predição observado. Para isto, a média dos erros

observados no passado $Err_{|\Delta|}$ para tamanhos de janelas $|\Delta|$ é armazenado à medida que as predições são realizadas. Deste modo, o fator de correção para uma janela de tamanho $|\Delta|$ é dado por:

$$CF_{|\Delta|} = 1 - min(Err_{|\Delta|} \cdot w_{cf}, 1)$$

onde $w_{cf} \in (0; 1]$ é o peso do fator de correção atribuído a cada erro ocorrido. O uso da função de mínimo (min) é usada para evitar que o fator de correção assuma valores negativos.

Quando a média dos erros ($Err_{|\Delta|}$) para um tamanho de janela $|\Delta|$ for positiva, o fator de correção ($CF_{|\Delta|}$) será menor que 1, resultando em uma estimativa menor do que a do modelo caixa-branca. Por outro lado, quando a média dos erros for negativa, o fator de correção ($CF_{|\Delta|}$) será maior que 1, resultando em uma estimativa maior do que a do modelo caixa-branca. Desta forma, pretende-se corrigir as estimativas futuras baseadas na média dos erros de predição obtidos no passado para os mesmos tamanhos de janela de predição.

Capítulo 4

Modelo de Geração de Carga de Trabalho Sintética

Neste capítulo é apresentado um modelo para a geração de carga de trabalho sintética realista para uma grade computacional P2P, que pode ser usado em simulações destes tipos de ambiente.

4.1 Visão Geral

Tendo em vista as limitações dos modelos de carga de trabalho usados para simular ambientes de grades P2P, listadas no Capítulo 1, é proposto a seguir um modelo hierárquico [23], baseado no comportamento dos usuários, para a geração de carga de trabalho sintética que seja representativa para o ambiente de grades computacionais P2P. Para isso, foram identificadas as semelhanças das grades P2P com diversos sistemas de computação paralela em produção, foram obtidos rastros de utilização desses diversos sistemas que estão disponíveis e, para cada atributo de carga de trabalho, foram extraídas informações dos rastros dos sistemas que possuem características mais próximas das grades P2P.

O processo de geração de carga de trabalho sintética para grades computacionais P2P proposto possui duas fases: a extração do modelo a partir de dados históricos e a geração da carga de trabalho sintética. O modelo obtido na primeira fase serve de entrada para a segunda fase, que terá como saída a carga de trabalho sintética gerada. A Figura 4.1 apresenta cada fase do processo e suas etapas, que são detalhadas a seguir.

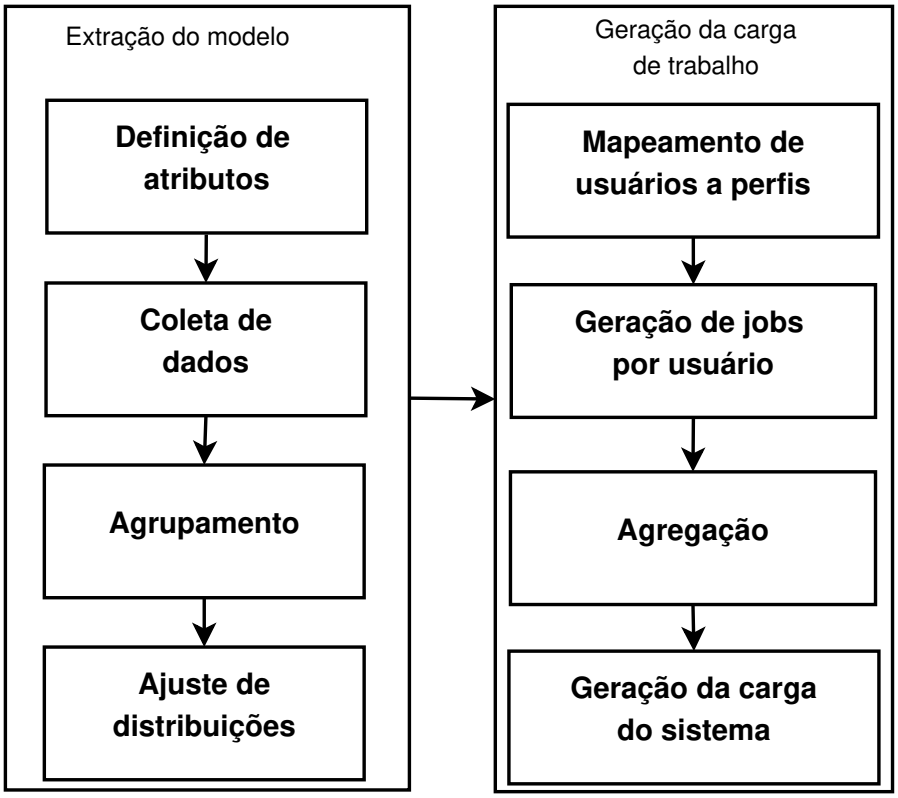


Figura 4.1: Fluxo do modelo de geração da carga de trabalho sintética

4.2 Extração do Modelo

A primeira fase consiste em extrair um modelo a partir de rastros históricos de utilização de sistemas em produção. Para isto, é necessário identificar quais atributos serão utilizados para gerar a carga de trabalho sintética, definir para cada atributo quais rastros serão usados para extrair o seu modelo, coletar os dados dos rastros para cada atributo, agrupar usuários em perfis de acordo com a similaridade para cada atributo e fazer ajustes de funções de distribuições de probabilidade aos dados para cada perfil de usuário. As etapas da extração do modelo são descritas a seguir.

4.2.1 Definição de Atributos de Carga de Trabalho

O modelo de carga de trabalho é, na verdade, a composição de modelos de diversos atributos que compõe a carga do sistema. É preciso, então, definir quais atributos serão usados na elaboração do modelo.

Dividiu-se os atributos de carga de trabalho em duas categorias: comportamento dos usuários e características das aplicações. A descrição dos atributos e a categoria à qual pertencem estão listados abaixo:

Comportamento dos usuários

Os usuários da grade P2P interagem com o sistema submetendo *jobs* para serem executados na infraestrutura e, após finalizarem as execuções, recebem como retorno os resultados gerados. O atributo usado para descrever esse comportamento das submissões de *jobs* pelos usuários é o:

- **Intervalo Entre Submissões (IES)**: Intervalo de tempo entre a submissão de dois *jobs* consecutivos por um mesmo usuário.

Características das aplicações por usuário

As aplicações dos usuários são submetidas para a grade P2P em *jobs* do tipo *Bag-of-Tasks* (BoT), que consiste em um conjunto de tarefas independentes que podem ser executadas em paralelo sem a necessidade de comunicação entre elas. Os atributos dessas aplicações que compõem o modelo de carga de trabalho são:

- **Tempo de Execução das Tarefas (TET)**: Tempo necessário para que uma tarefa seja executada em uma máquina de referência.

- **Tempo de Execução dos Jobs (TEJ)**: Somatório do tempo de execução de todas as tarefas que compõem um mesmo *job*.

Um outro atributo referente às aplicações é a quantidade de tarefas que compõem cada *job*. Porém, pode-se observar que uma estimativa para este atributo pode ser obtida a partir dos modelos do tempo de execução total dos jobs (TEJ) e do tempo de execução de tarefas (TET) que compõem esses jobs, que são os dois atributos explicitamente modelados.

Portanto, modelando os dois atributos listados acima, pode-se gerar aplicações com características determinadas pelos modelos.

4.2.2 Coleta de Dados

Para que o modelo para geração de carga de trabalho seja criado, é preciso que ele seja baseado em observações de dados históricos de sistemas existentes, ou que seja elaborado a partir da intuição de como um certo sistema deve se comportar. A primeira abordagem é a mais indicada, já que se sustenta em dados reais. Porém, nem sempre é possível encontrar rastros realistas de carga de trabalho do ambiente em questão. É o caso das grades computacionais P2P, para as quais não há registros de sistemas com uma utilização representativa.

Como pretende-se gerar carga de trabalho sintética de uma grade computacional P2P para simular este tipo de ambiente e não temos rastros disponíveis, a abordagem utilizada foi a de obter dados de outros sistemas em produção que possuem rastros de utilização disponíveis. Para isto, identificou-se quais tipos de infraestruturas de alto desempenho em produção possuíam mais similaridade, para cada atributo do modelo, com a visão da grade computacional P2P. Identificamos como possíveis rastros a serem utilizados na coleta de dados:

1. Rastros de grades computacionais de serviço disponíveis no Grid Workload Archive (GWA) [54].

2. Rastros de supercomputadores e *clusters* disponíveis no Parallel Workload Archive (PWA) [22].

3. Rastro da plataforma de computação voluntária Ibercivis [52], disponibilizada pelos responsáveis[1].

Das opções listadas, a infraestrutura que possui mais similaridade com a grade computacional P2P é a grade computacional de serviço, pois também consiste em vários usuários submetendo *jobs* compostos por múltiplas tarefas, para serem executados em recursos que são distribuídos em diferentes instituições [27]. Nos rastros de supercomputadores e *clusters*, os usuários e recursos geralmente estão dentro de uma mesma instituição, havendo pouca heterogeneidade na grade. Em plataformas de computação voluntária, como o Ibercivis, os usuários não submetem suas aplicações com frequência ao sistema. Ao invés disso, poucas aplicações são armazenadas em um servidor com uma enorme quantidade de cenários, o que normalmente leva muito tempo para ser finalizado. Logo, o seu uso não é adequado, pois o processo de submissão de *jobs* e, por consequência, o comportamento dos usuários do sistema, possui características diferentes das grades P2P. Desta forma, no modelo proposto, os atributos relacionados ao comportamento dos usuários são extraídos de rastros de grades de serviço, que é o que mais se aproxima das grades P2P.

Porém, nas grades de serviço os recursos são dedicados, resultando em longos períodos de disponibilidade dos recursos e, consequentemente, baixa taxa de falhas de execução das tarefas, o que traz uma certa garantia na qualidade de serviço oferecida. Por outro lado, as grades P2P são compostas por recursos oportunistas, que só são disponibilizados à grade quando ociosos, podendo se tornar indisponíveis para a grade a qualquer momento em que o dono dos recursos necessitar deles. Isto traz incertezas na qualidade de serviço provida e resulta em uma taxa de falhas na execução de tarefas maior do que nas grades de serviço. No trabalho de Kondo et al. [34] isto fica evidenciado, quando são apresentadas as médias e medianas da duração dos períodos de disponibilidade das máquinas que compõem clusters (do *Los Alamos National Laboratory* (LANL), Estados Unidos), uma grade computacional experimental usada como plataforma de teste (do *Grid'5000*, França) e uma grade computacional oportunista (da *University of Notre Dame*, França), como apresentados na Tabela 4.1. Os clusters possuem os maiores valores para períodos de disponibilidade, seguidos da grade experimental e, por último, as grades oportunistas, que possuem as menores média e mediana para períodos de disponibilidade de máquinas, entre os sistemas comparados.

[1] Dados cedidos pela equipe do Ibercivis da Universidade de Zaragoza.

Sistema	**Média**	**Mediana**
Clusters (Los Alamos)	1780 horas	280,3 horas
Grade experimental (Grid'5000)	32,4 horas	7,1 horas
Grade oportunista (Notre Dame)	13,7 horas	1,1 hora

Tabela 4.1: Média e mediana dos períodos de disponibilidade de máquinas em sistemas de computação de alto desempenho [34]

O fato da disponibilidade das máquinas em grades oportunistas possuírem curtos períodos de duração, com relação a outras infraestruturas de computação, implica em uma maior taxa de falhas na execução de tarefas. Além disso, quanto maior o tempo necessário para executar cada tarefa, maior a probabilidade de falha. Então, tarefas de longa duração não são adequadas para serem executadas em infraestruturas em que os recursos apresentam curtos períodos de disponibilidade. Este fato é evidenciado por Kondo et al., onde é constatado que o tempo de execução ideal para as tarefas executarem na grade oportunista analisada é entre 13 e 25 minutos [35].

Na Tabela 4.2 são apresentadas estatísticas do tempo de execução de tarefas extraído de rastros de diversos sistemas. Os rastros dos sistemas estão divididos em dois tipos: as grades de serviço (obtidas no GWA) e a grade de computação voluntária (Ibercivis). Observa-se que a maioria das grades de serviço possuem média de tempo de execução acima de 1 hora, com exceção de GS-1 e GS-2. O fato da média do tempo de execução desses dois sistemas serem menores que os das outras grades de serviço é que o primeiro (DAS-2) é uma infraestrutura usada para testes de novos sistemas, usada muitas vezes de forma interativa, não sendo um sistema que executa aplicações em produção. O GS-2 (Grid'5000) também é uma plataforma experimental, possuindo características semelhantes [27]. As demais grades de serviço possuem média de tempo de execução de tarefas longe do ideal para grades oportunistas. Os rastros da grade de computação voluntária, por outro lado, apresenta os valores mais próximos do ideal indicado por Kondo et al. [35]. Estas características fazem do rastro da grade de computação voluntária (Ibercivis) o mais adequado, entre os sistemas disponíveis, para a extração dos dados de tempo de execução de tarefas, para serem usados no modelo de carga de trabalho da grade P2P.

Tipo	**Sistema**	**Média**	**Mediana**	**Desvio padrão**
Grade de serviço	GS1 - DAS-2	6 min	1 min	66 min
	GS2 - Grid'5000	52 min	3 min	450 min
	GS3 - NorduGrid	1491 min	203 min	4743 min
	GS4 - AuverGrid	430 min	44 min	685 min
	GS5 - SHARCNET	533 min	49 min	1951 min
	GS6 - LCG	150 min	4 min	548 min
Grade voluntária	GV1 - Ibercivis	36 min	20 min	36 min

Tabela 4.2: Estatísticas do tempo de execução de tarefas em diferentes sistemas

O último atributo de carga de trabalho, *tempo de execução total dos jobs*, definido anteriormente, foi extraído dos rastros de grades de serviço. O rastro de grade voluntária foi descartado, pois não há uma divisão explícita de tarefas em *jobs*, e a quantidade de tarefas que compõem cada aplicação é muito grande, fazendo com que o somatório do tempo de execução das tarefas de compõem uma aplicação também seja muito grande. Logo, sistemas de grades voluntárias, no geral, não seguem o perfil das grades P2P onde usuários submetem *jobs* compostos de diversas tarefas, esperam o fim da execução, analisam os dados e voltam a submeter novos *jobs*. Como a quantidade de tarefas que fazem parte de uma aplicação normalmente é muito grande, cada aplicação pode levar meses ou até anos para ser finalizada. Deste modo, o *tempo de execução total dos jobs* foi obtido dos rastros das grades de serviço, onde o comportamento dos usuários é similar ao das grades P2P e, por consequência, foi assumido que o tempo que os usuários estão dispostos a esperar para finalizar a execução de seus *jobs* também serão similares. Nos rastros das grades de serviço não há informação explícita da associação de tarefas em *jobs*. Para isto, foi usado o modelo proposto por Iosup et al. [26], em que tarefas submetidas por um mesmo usuário, em intervalos inferiores a 2 minutos entre cada tarefa, são consideradas de um mesmo job.

A Tabela 4.3 resume os atributos que compõem o modelo de carga de trabalho da grade computacional P2P, indicando de quais sistemas foram obtidos os rastros usados para extrair os dados de cada atributo.

Atributo de carga de trabalho	**Tipo de sistema**	**Fonte dos rastros**
Intervalo Entre Submissões (IES)	Grade de serviço	GS1, GS2, GS3, GS4, GS5, GS6
Tempo de Execução de Tarefas (TET)	Grade voluntária	GV1
Tempo de Execução de Jobs (TEJ)	Grade de serviço	GS1, GS2, GS3, GS4, GS5, GS6

Tabela 4.3: Atributos de carga de trabalho usados na modelagem, os tipos de sistema utilizados para obtenção dos dados de cada atributo e a fonte dos rastros

4.2.3 Agrupamento de Usuários e Aplicações

Sistemas como grades computacionais são compostos de diversos usuários e aplicações, que nem sempre possuem as mesmas características. Por exemplo, alguns usuários podem submeter *jobs* para a grade com uma alta frequência, enquanto outros submetem com baixa frequência. Alguns *jobs* podem ser de longa duração, sendo compostos por uma grande quantidade de tarefas, e outros podem ser menores e com poucas tarefas. Deste modo, modelou-se os atributos de carga de trabalho agrupando usuários e aplicações em diferentes perfis, criando um modelo para cada grupo encontrado.

Baseado no trabalho de Javadi et al. [30], dois métodos de agrupamento foram considerados para a modelagem: o *k-means* e o agrupamento hierárquico [25]. No *k-means*, k centróides de grupos são escolhidos aleatoriamente e os pontos são incluídos no grupo em que seu centro esteja mais próximo a eles. Repete-se este procedimento até que se atinja convergência. Porém, o *k-means* necessita receber como entrada um número k fixo de grupos, além de apresentar um comportamento não determinístico, podendo apresentar diferentes resultados em cada execução.

O agrupamento hierárquico é realizado de forma iterativa, juntando em cada passo os grupos que possuem maior similaridade, a partir de uma matriz de distâncias entre cada ponto. Inicialmente cada ponto representa um grupo, então são realizados vários passos de agrupamento até que no fim é obtido apenas um grupo que descreve todos os pontos. Apesar de ser lento por necessitar diversas iterações, esta abordagem tem a vantagem de apresentar uma visão de agrupamentos de diversos tamanhos, ao contrário do *k-means* que necessita

de um *k* específico. Além disso, o algoritmo de agrupamento hierárquico é determinístico, apresenta sempre o mesmo resultado para uma dada matriz de distância entre os pontos.

Para a modelagem dos atributos de carga de trabalho, o agrupamento hierárquico se mostrou mais adequado, pois a quantidade de grupos não necessita ser especificada de antemão, apresentando uma visão geral de cada passo do agrupamento para só depois ser definido quantos grupos terão. O fato do algoritmo ser determinístico também é positivo, pois só uma execução é necessária para a obtenção dos grupos com validade estatística.

Para realizar o agrupamento hierárquico, é necessário fornecer como entrada uma matriz de distância entre os pontos a serem agrupados. No caso dos atributos de carga de trabalho, esses pontos serão as características de cada usuário (*intervalo entre submissões de jobs*) ou de cada aplicação (*tempo de execução de tarefas* e *tempo de execução total de jobs*). Uma maneira de descrever as características de cada usuário é através da distribuição de probabilidade dos dados obtidos para cada atributo. Isto pode ser feito gerando a função distribuição acumulada (FDA) empírica dos dados. Após obter a FDA para cada tupla <*atributo,usuário*>, deve-se usar algum método para indicar qual a distância entre cada distribuição. Um dos métodos mais utilizados para comparar FDAs empíricas é o teste Kolmogorov-Smirnov (KS), que calcula a distância máxima entre distribuições de probabilidade acumulada [23]. Este teste foi escolhido para ser usado na geração da matriz de distância do agrupamento hierárquico. Quanto maior a estatística "D" fornecida pelo teste KS, maior a distância máxima entre as distribuições de probabilidade acumulada empírica.

Uma dificuldade encontrada na comparação de FDAs empíricas é que o tamanho da amostra para cada distribuição pode ser diferente. Para contornar este problema, foi utilizada uma abordagem usada em outros trabalhos [30; 42; 34], em que amostras de tamanho fixo são obtidas dos dados, sendo então comparados cada par de amostra de mesmo tamanho várias vezes. Como nos trabalhos citados, o tamanho de cada amostra usado foi 30 e o teste foi repetido 1.000 vezes para cada par de distribuições, para assim obter o p-valor médio do teste KS, que foi usado como valor da distância entre as distribuições.

Cada grupo obtido para cada atributo é descrito por uma distribuição de probabilidade, que descreve o comportamento dos usuários e características das aplicações que compõem os grupos. A próxima etapa descreve o processo de obtenção de funções de distribuição de probabilidade para cada grupo.

4.2.4 Ajuste de Funções de Distribuição de Probabilidade

Após o agrupamento de usuários e aplicações de acordo com as características de cada atributo, são realizados ajustes de funções de distribuição de probabilidade aos dados de cada atributo para extrair uma função de distribuição de probabilidade que represente cada grupo de usuários e aplicações.

Inicialmente, é preciso definir um conjunto de funções de probabilidade candidatas ao ajuste de distribuições [38]. Assim como no trabalho de Iosup et al. [28], são escolhidas distribuições que são amplamente usadas na análise de desempenho de sistemas e apresentam baixa complexidade: exponencial, normal, log-normal, gamma e weibull.

Em seguida, é preciso encontrar para cada distribuição quais os valores dos parâmetros adequados para se obter o melhor ajuste possível. O método utilizado foi o de Estimação por Máxima Verossimilhança (MLE, do inglês *Maximum Likelihood Estimation*), que é amplamente usado e consiste em obter os valores dos parâmetros das distribuições que resultem na maior probabilidade de gerar uma amostra que coincida com os dados [23; 38].

Depois de definidos os melhores parâmetros para o ajuste de cada distribuição candidata, é preciso determinar a qualidade de cada ajuste realizado e escolher qual das distribuições obteve o ajuste de melhor qualidade. Para isto, são realizados os testes de Qualidade de Ajuste (GoF, do inglês *Goodness-of-Fit*), que oferece uma métrica que indica quão bom foi o ajuste de uma função de distribuição com os dados [21]. O método Kolmogorov-Smirnov (KS), que é amplamente usado para este propósito [38; 28; 34; 23], foi usado na modelagem da carga de trabalho.

O resultado do teste KS é um p-valor, que diz respeito ao resultado do teste de hipótese em que a hipótese nula é que a distribuição de probabilidade dos dados observados é a mesma da função de distribuição de probabilidade teórica utilizada no ajuste. Se o p-valor for maior que o nível de significância α escolhido, então a hipótese nula não é rejeitada, e o ajuste é considerado satisfatório. O nível de significância mais usado é de $\alpha = 0.05$ [23], que é o mesmo usado neste trabalho.

Após o ajuste dos dados às distribuições de probabilidade, têm-se diversas funções de distribuição de probabilidade que representam o comportamento de usuários e características de aplicações de uma grade computacional P2P.

4.3 Geração da Carga de Trabalho Sintética

A segunda fase do processo consiste em gerar a carga de trabalho sintética a partir do modelo extraído (descrito na Seção 4.2). Primeiro, é definida a quantidade de usuários na grade P2P e a qual perfil (ou grupo) cada um pertence. Em seguida, os valores para cada atributo são gerados através de suas distribuições, com a geração de números aleatórios. Depois, a geração de *jobs* de cada usuário é feita a partir dos valores gerados de cada atributo. Por fim, os *jobs* de cada usuário são agregados para formar a carga total do sistema. Este tipo de geração de carga de trabalho é chamado de modelo hierárquico, baseado no comportamento dos usuários [23].

4.3.1 Mapeamento de Usuários da Grade a Perfis

A grade P2P é composta por diversos nós (ou *peers*). Cada nó do sistema é formado por um conjunto de usuários. Cada usuário terá um perfil, descrito por um conjunto de funções de distribuições de probabilidade (FDP), sendo uma função para cada atributo do modelo. Os perfis são obtidos da fase de extração do modelo. A quantidade de usuários por nó da grade é definido como um parâmetro de entrada do gerador da carga de trabalho sintética.

O mapeamento de usuários a perfis é feito com base em dados obtidos da fase de extração do modelo. Na fase de agrupamento, descrita na Seção 4.2, que extrai os perfis para cada atributo de carga de trabalho, também é obtida a informação da proporção de usuários do rastro do sistema que cada perfil (ou grupo) cobre. Então, a probabilidade de um usuário ter um determinado perfil será igual à proporção de usuários que este perfil cobre no rastro do sistema do qual foi extraído o modelo. Por exemplo, se de um rastro de um sistema com 100 usuários foram extraídos 3 perfis, onde o perfil 1 cobre 60 usuários, o perfil 2 cobre 25 usuários e o perfil 3 cobre 15 usuários, a probabilidade de um usuário ter cada um desses perfis na geração de carga sintética será de 60%, 25% e 15%, respectivamente.

4.3.2 Geração de Jobs por Usuários

Após serem definidos os usuários do sistema, cada um com seu conjunto de distribuições de probabilidade que irá descrever seus perfis, os valores para cada atributo são gerados através de suas distribuições, com a geração de números aleatórios. Em seguida, os valores

gerados são agregados de acordo com um modelo, para formarem os *jobs* submetidos por cada usuário da grade.

O fluxo para a geração de um *job* da carga de trabalho está apresentado na Figura 4.2, que mostra a sequência dos atributos que é usada para gerar cada job de um usuário. Primeiro, obtém-se da distribuição do intervalo entre submissões de jobs, isto é, o valor de quanto tempo depois do *job* anterior o próximo *job* será submetido. Depois, obtém-se através de sua distribuição específica o valor do tempo de execução total do *job*, ou seja, do somatório do tempo de execução de todas as tarefas que compõem o *job*. Depois disso, geram-se diversos valores para tempos de execução de tarefas, até que o somatório desses valores gerados atinjam o valor gerado anteriormente do tempo de execução total do *job*. Por exemplo, se o valor para o tempo de execução total do *job* for de 10 horas e a distribuição do tempo de execução de tarefas for constante em 30 minutos, o *job* será composto por 20 tarefas de 30 minutos cada.

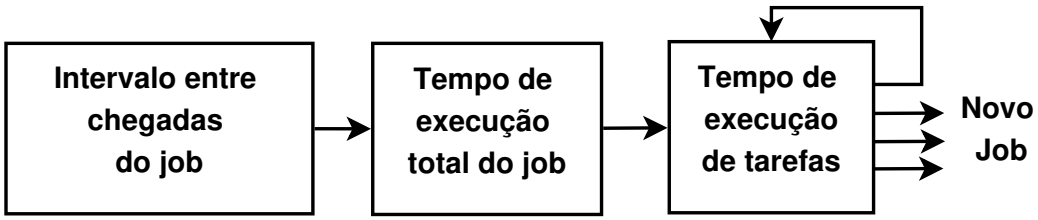


Figura 4.2: Fluxo da geração de um job da carga de trabalho sintética

4.3.3 Agregação e Geração da Carga Total do Sistema

Por fim, a carga de trabalho de cada usuário é agregada, gerando assim a carga total do sistema. Como resultado final do processo, será criado um registro com uma lista de *tarefas* submetidas à grade, que contém informações do *job* do qual elas fazem parte, do usuário que as submeteu, do nó do sistema que o usuário faz parte, do tempo em que a tarefa foi submetida e quanto tempo de processamento é necessário para que a tarefa seja executada em uma máquina de referência.

Capítulo 5

Avaliação

5.1 Metodologia

O processo de avaliação do modelo de predição foi dividido em três etapas, como mostra a Figura 5.1. Primeiro, realiza-se uma simulação do ambiente de grade computacional P2P, com uma determinada carga de trabalho e configuração da grade, gerando um rastro que descreve as interações entre os nós que compõem esta grade durante a simulação. Em seguida, a partir dos rastros gerados nas simulações, aplica-se a predição utilizando o modelo apresentado no Capítulo 3, além de outras estratégias de predição usadas para comparação. Finalmente, são obtidas as métricas de comparação do desempenho dos modelos de predição e os resultados são discutidos.

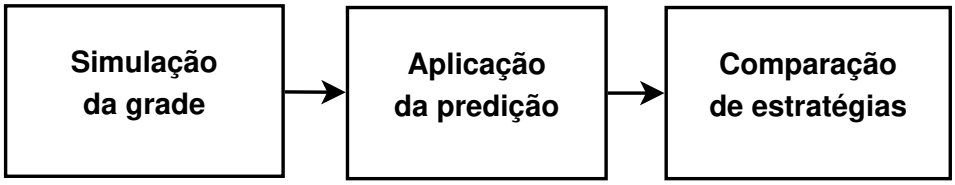


Figura 5.1: Etapas da avaliação dos modelos de predição

5.1.1 Simulação da Grade

Um simulador para a grade computacional P2P que usa a NoF como mecanismo de incentivo foi desenvolvido, utilizando a linguagem de programação Java. Como entrada, o simulador

recebe uma carga de trabalho que descreve as submissões de jobs realizadas pelos usuários da grade, e a configuração da grade, que descreve a quantidade de nós que compõem o sistema e quantos recursos cada nó possui.

Duas abordagens foram utilizadas para a carga de trabalho usada na simulação da grade P2P. Na primeira, a carga de trabalho usada na simulação foi gerada a partir do modelo de geração de carga de trabalho sintética proposto no Capítulo 4, com objetivo de avaliar os modelos de predição em cenários realistas de grades computacionais P2P. A aplicação do modelo de geração de carga sintética é apresentado na Seção 5.2.

Na segunda abordagem, foi usada uma carga de trabalho real, obtida do rastro de um sistema de grade computacional de serviço em produção que, apesar de não ser uma carga de trabalho representativa para grades computacionais P2P, foi analisada com o objetivo de investigar como seriam as predições caso o mecanismo de incentivo da NoF fosse utilizado no gerenciamento de recursos desse sistema.

Para a extração do modelo e geração de carga sintética da grade, que é usada como entrada da simulação, foi desenvolvido um pacote de *scripts* para o ambiente *R*, que oferece várias bibliotecas estatísticas úteis para ajuste de distribuições, agrupamento, etc.

A grade computacional P2P simulada é composta por máquinas heterogêneas, ou seja, com diferentes capacidades de processamento. Os dados do poder de processamento das máquinas usados na simulação foram obtidos do banco de dados da grade de computação voluntária Ibercivis [52], que utiliza o BOINC como o *middleware* do sistema. A capacidade de processamento das máquinas dos voluntários é medida através do *Whetstone benchmark* [55], que estima o poder de processamento de uma máquina em operações de ponto flutuante por segundo (FLOPS).

A capacidade média de processamento foi considerada como referência, e a capacidade das máquinas foi calculada com base na capacidade de referência, o que chamamos de capacidade relativa, que é calculada da seguinte forma:

$$capacidade\_relativa(x) = \frac{capacidade(x)}{capacidade\_media}$$

De acordo com esta fórmula, a média da capacidade relativa será 1. O desvio padrão obtido para a capacidade relativa das máquinas foi de 0,3, enquanto os valores de mínimo e máximo foram 0,5 e 4,4. Verificou-se que a distribuição das capacidades de processamento

em FLOPS, das máquinas que compõem a grade voluntária, obtém um bom ajuste com a distribuição normal. Foi aplicado o teste KS para verificar este ajuste, usando a mesma metodologia de ajuste de distribuições usada na Seção 4.2.4.

A Tabela 5.1 apresenta os valores dos parâmetros usados para a distribuição normal usada na geração dos dados de capacidade de processamento relativa das máquinas, usados nas simulações. A distribuição foi limitada em um mínimo e um máximo, para evitar valores negativos ou valores extremos. O limite inferior usado foi 0,2 (que corresponde ao 0,4-percentil da distribuição) e o limite superior foi 2 (que corresponde ao 99,9-percentil da distribuição).

Distribuição	**Média**	**Desvio Padrão**	**Limite inferior**	**Limite superior**	**p-valor**
Normal	1	0,3	0,2	2	18,5%

Tabela 5.1: Função de distribuição usada na geração da capacidade de processamento das máquinas da grade

A saída gerada pelo simulador é um rastro de todas as interações entre os nós da grade durante o período de simulação. O rastro inclui informações da doação de máquinas entre os usuários, com as respectivas capacidades de processamento de cada máquina. Também inclui os períodos de submissão de jobs e o tempo em que cada tarefa finalizou sua execução.

5.1.2 Aplicação das Predições

As predições são aplicadas utilizando como entrada os rastros gerados pelas simulações da grade. Os modelos de predição caixa-branca e caixa-cinza apresentados, respectivamente, nas Seções 3.1 e 3.3 são executado, além das outras quatro técnicas de predição caixa-preta apresentadas na Seção 3.2, obtidas da literatura, que são aplicadas em outros contextos e foram adaptadas para o nosso problema.

Aplicamos as predições para cada um dos nós do sistema fazendo o papel do nó local ℓ, e para várias combinações de janela de predição Δ. Para cada nó do sistema, executamos avaliações para os períodos em que este nó está no estado de consumidor de recursos, que serão chamados de janela de consumo. Uma janela de consumo é o intervalo que se inicia

quando o nó local vira consumidor (passa a requisitar recursos) na grade e termina quando estas requisições são satisfeitas (ou a execução de todas suas tarefas são finalizadas).

Diversas predições são realizadas durante um período de consumo. Definimos t_c como o turno em que se inicia a janela de consumo, e que também será o inicio da janela de predição Δ. O nó local ℓ obtém informações da grade em t_c e aplica a predição para janelas de tempo $\Delta = [t_c, t_p]$, que varia de tamanho de acordo com o fim da janela de predição t_p escolhido. O erro da predição ξ é calculado para diferentes tamanhos da janela de predição (Δ), incrementando t_p em uma unidade de turno até que o final da janela de consumo seja atingido. Por exemplo, se um nó passa a ser consumidor a partir do turno $t_c = 0$ e se mantém consumindo recursos até o turno $t = 60$, então 60 predições serão realizadas com diferentes tamanhos de janela de predição, com $t_p \in \{1, 2, ..., 60\}$.

Durante a avaliação, consideramos que cada turno tem a duração de 2 minutos, ou seja, um Δ de 60 turnos é equivalente a uma janela de predição de duas horas.

5.1.3 Comparação de Estratégias

Para avaliar o modelo de predição, foi realizada uma comparação entre os valores das estimativas do modelo para a capacidade de processamento obtida em janelas de tempo Δ com os valores que foram obtidos de fato nas simulações. O erro da predição ξ foi calculado de acordo com a fórmula do erro relativo:

$$\xi = \frac{estimado(\Delta) - obtido(\Delta)}{obtido(\Delta)} \tag{5.1}$$

onde $obtido(\Delta)$ é a capacidade de processamento agregada obtida pelo nó local ℓ durante toda a janela de predição Δ na simulação, e $estimado(\Delta)$ é a estimativa para esta capacidade de processamento obtida, que é gerada pelos modelos de predição.

O erro da predição ξ foi calculado para cada estratégia em diversos cenários de simulação, utilizando carga de trabalho sintética e rastros de sistemas em produção. Os parâmetros de entrada da simulação, que definem os cenários são:

- n : quantidade de nós na grade
- u : quantidade de usuários por nó

- r : quantidade de recursos por nó

Consideramos que os valores da quantidade de usuários por nó u e da quantidade de recursos por nó r serão os mesmos para todos os n nós da grade.

5.1.4 Infraestrutura para Execução de Experimentos

Para realizar os experimentos, foram utilizados 3 ambientes distintos de execução:

- **Desktop**: máquina com processador Intel Core 2 Duo CPU E6550 @ 2.33GHz e 2GB de memória RAM
- **Cluster**: composto por 8 nós, cada um com 2 processadores Intel Xeon CPU X5550 @ 2.67GHz, cada processador com 8 cores. Cada nó possui memória RAM de 8GB
- **OurGrid**: grade computacional oportunista, que durante o período dos experimentos tinha disponível aproximadamente 200 máquinas virtuais

A extração do modelo foi realizada em sua maior parte no OurGrid, principalmente a fase de agrupamento e ajuste de distribuições, que é bastante paralelizável. Cada tarefa desta fase durou em média meia hora para ser executada na grade.

A geração da carga de trabalho sintética, as simulações da grade P2P e a aplicação das estratégias de predição foram executadas no Cluster, pois exigiram uma maior capacidade de processamento e possuíam um longo tempo de execução para cada cenário. A execução dos cenários de simulação da grade variaram entre 1 hora e 2 dias.

A análise dos resultados e outras tarefas menores que exigiam menor poder computacional foram realizadas na máquina desktop.

5.2 Aplicação do Modelo de Geração de Carga Sintética

Nesta seção, é apresentada uma aplicação do processo de extração do modelo de carga de trabalho descrito na Seção 4.2, para gerar carga de trabalho sintética que será usada em simulações da grade computacional P2P. São apresentados os resultados do agrupamento dos usuários em perfis e do ajuste de distribuições aos dados.

5.2.1 Extraindo o Modelo

De acordo com o processo apresentado, os atributos usados para extração do modelo são os listados na Tabela 4.3 (página 28). Primeiro, são apresentados os resultados dos modelos extraídos do Intervalo Entre Submissões (IES) e do Tempo de Execução de Jobs (TEJ), ambos extraídos de 6 rastros de grades de serviço, descritos na Seção 4.2 (ver Tabela 4.3, página 28). Para estes dois atributos, uma transformação logarítmica com base 2 foi aplicada aos dados, para reduzir o impacto de valores extremos no ajuste de distribuições. Esta é uma prática bastante usada em modelagem de carga de trabalho, que não afeta a qualidade do ajuste de distribuições [38; 28].

Inicialmente, deve-se definir a quantidade de grupos que serão extraídos para cada atributo do modelo. Como dito anteriormente, cada grupo de cada atributo terá uma função de distribuição de probabilidade associada, que irá descrever o perfil dos usuários que compõem esses grupos. Então, quanto maior a quantidade de grupos, menor a quantidade de usuários por grupo e maior a quantidade de distribuições que serão usadas para descrever o comportamento desses usuários.

Com base nisto, foi criada uma métrica que define a qualidade do ajuste das distribuições de probabilidade para cada grupo, definida abaixo:

- **Coeficiente de Qualidade de Ajuste** (Q): proporção de usuários de um grupo que possuem ajuste de distribuição de probabilidade satisfatório, considerando que cada grupo é composto por um ou mais usuários e que possui apenas uma função de distribuição de probabilidade associada, onde um ajuste de distribuição de probabilidade é considerado satisfatório quando o p-valor obtido no teste KS é maior que o nível de significância ($\alpha = 0.05$).

Se a quantidade de grupos (k) para a modelagem de um atributo for igual a 1, existirá apenas uma distribuição de probabilidade que irá representar o comportamento de todos os usuários do sistema. Isto faz com que o modelo fique bastante genérico, o que é um ponto positivo. Porém, a probabilidade de apenas uma distribuição obter um ajuste satisfatório para todos os usuários do sistema é baixa, pois a tendência é que usuários possuam diferentes características.

Por outro lado, se k for igual à quantidade de usuários do sistema, então cada grupo só será composto por um usuário e para cada usuário do sistema existirá uma distribuição de probabilidade que irá representar seu comportamento. Desta forma, será mais fácil obter um ajuste satisfatório para todos os usuários de cada grupo ($Q \approx 1$), pois cada distribuição só terá que representar o comportamento de um único usuário. Porém, como a quantidade de grupos será muito grande e, consequentemente, existirão muitas distribuições de probabilidade que só representarão um único usuário cada uma, o modelo se tornará muito específico e apresentará grande complexidade para uso. Desta forma, procura-se escolher uma quantidade k de grupos que não seja tão grande, mas que também consiga obter uma alta proporção de usuários com ajustes de distribuição satisfatórios (ou seja, altos valores de Q).

A Figura 5.2 mostra a métrica de Coeficiente de Qualidade de Ajuste (Q) com quantidades de grupos (k) variando de 1 a 20, para os atributos Intervalo Entre Chegadas (IEC) e Tempo de Execução de Jobs (TEJ). Os agrupamentos e ajustes de distribuições foram feitos para cada um dos 6 rastros de grades de serviço (GS) separadamente. Cada gráfico representa um cenário de atributo de carga de trabalho e rastro de sistema, apresentando um total de 12 cenários. As 5 distribuições usadas no ajuste são aquelas mencionadas na Seção 4.2.4. Além do resultado do ajuste de cada distribuição utilizada separadamente, também é apresentado o resultado chamado de *best*, que permite que diferentes funções de distribuição sejam usadas em diferentes grupos, sendo escolhida para cada grupo a melhor distribuição obtida entre as 5, de acordo com o teste KS de qualidade de ajuste.

A distribuição exponencial (ou *exponential* no gráfico) apresenta baixos valores de Q para todos os cenários. Este fato é relevante, pois esta distribuição é largamente usada em análise de desempenho de sistemas, principalmente na modelagem do intervalo entre submissões de *jobs*. Neste caso, ela não se mostrou adequada para a modelagem dos dois atributos em questão. Todas as outras distribuições obtiveram tendências parecidas, apresentando pequenas variações dependendo do rastro utilizado ou do atributo em questão.

Ao invés de utilizar para cada atributo apenas uma função de distribuição de probabilidade e realizar o ajuste apenas variando os seus parâmetros, optou-se por permitir que, para um único atributo, cada grupo possua diferentes funções de distribuição de probabilidade, além de diferentes parâmetros dessas distribuições. Este é o caso da abordagem *best* apresentada nos gráficos, que escolhe a melhor função de distribuição para cada grupo.

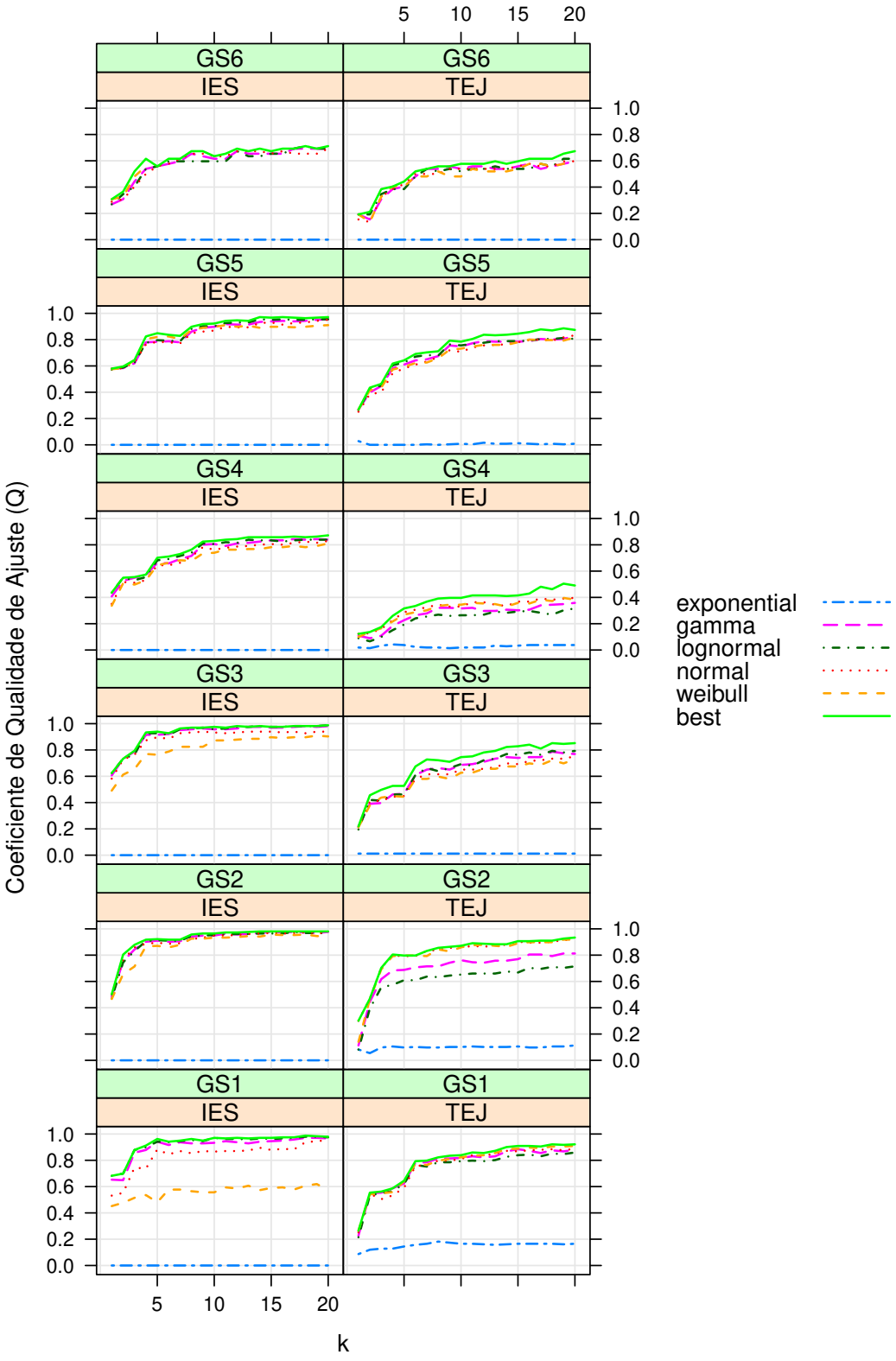


Figura 5.2: Coeficiente de Qualidade de Ajuste (Q) em diversos cenários

Observa-se que, à medida que o k aumenta, o coeficiente de qualidade do ajuste Q também tende a aumentar. Isto é esperado, pois quanto maior o k, maior a quantidade de distribuições usadas no ajuste e menor a quantidade de usuários que cada distribuição terá que representar, aumentando assim a probabilidade de se obter ajustes satisfatórios para os usuários. No geral, o coeficiente Q tem um alto crescimento ao incrementar os valores de k iniciais. Após um certo valor de k, a curva de Q tende a estabilizar, não havendo mais um crescimento significativo à medida que o k aumenta. O ideal é obter um k que não seja muito grande, mas que apresente um Q próximo ao máximo obtido.

Está fora do escopo deste trabalho elaborar uma técnica que procure obter o melhor valor de k para cada atributo. Por simplicidade, foi utilizado um único valor de k para todos os cenários de atributo e rastro. Observa-se através dos gráficos que, para a maioria dos cenários, com um $k = 5$ a curva do Coeficiente de Qualidade de Ajuste (Q) para a abordagem *best* já começa a estabilizar o seu crescimento. Deste modo, foi escolhida a quantidade de grupos $k = 5$ para cada par de atributo e rastro.

Como foram utilizados 6 rastros de sistemas para o Intervalo Entre Submissões (IES) e o Tempo de Execução de Jobs (TEJ) e para cada rastro foram divididos 5 grupos para cada atributo, o resultado foi um total de 30 funções de distribuição de probabilidade que representarão diferentes comportamentos para cada atributo. As Tabelas A.1 e A.2, do Apêndice A, mostram as distribuições de probabilidade obtidas para cada grupo para os atributos Intervalo Entre Submissões (IES) e Tempo de Execução de Jobs (TEJ), respectivamente. A Função Distribuição Acumulada (FDA) para cada grupo, dividido por atributo e rastro, é apresentada na Figura 5.3. Observa-se que há uma grande variedade nas curvas das distribuições, destacando as diferenças de comportamento dos usuários do sistema que cada distribuição representa.

Por fim, o modelo para o atributo Tempo de Execução das Tarefas (TET) é extraído do rastro da grade voluntária Ibercivis, como descrito na Seção 4.2. Nos rastros, apenas 4 aplicações possuem uma quantidade de dados relevante para a análise (mais de 100 ocorrências). As descrições dessas aplicações, obtidas do sítio do Ibercivis[1], são apresentadas abaixo:

- **adsorcion** - usada por pesquisadores do Instituto de Química-Física Rocasolano, do Consejo Superior de Investigaciones Científicas (CSIC). Consiste na análise das pro-

[1] Ibercivis - http://www.ibercivis.es

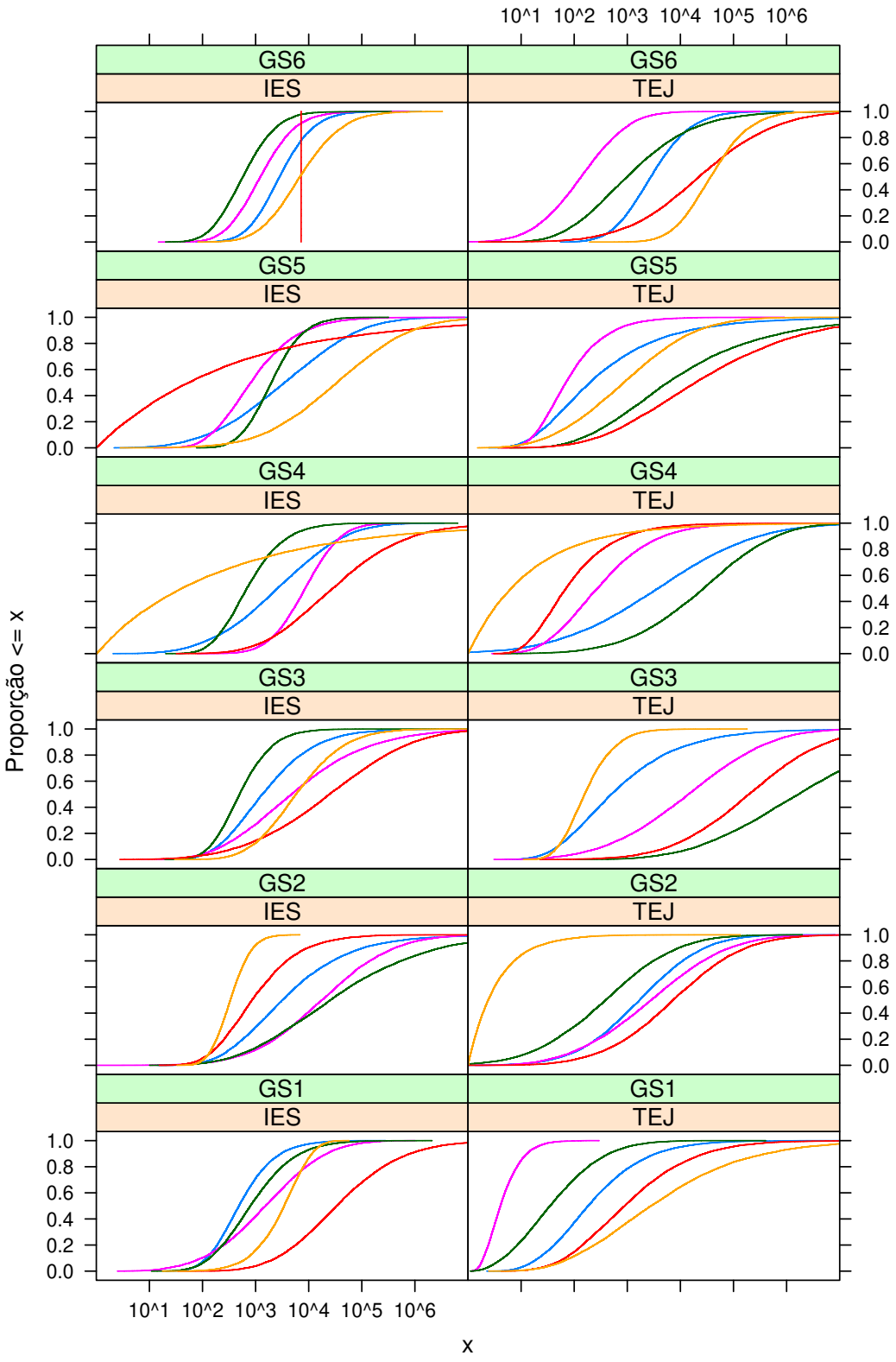


Figura 5.3: Função Distribuição Acumulada para cada grupo

priedades de materiais, como a adsorção de argilas.

- **amld-docking** - ou amiloide, é usada pelo Grupo de Biologia Estrutural e Computacional do Centro de Neurociências e Biologia Celular (CNC), da Universidade de Coimbra, em Portugal. Consiste na procura de fármacos para combater doenças amiloides neurodegenerativas.

- **criticalidad** - usada por pesquisadores do Instituto de Física de la Benemérita, da Universidad Autónoma de Puebla, no México. Consiste no estudo do transporte elétrico em sistemas desordenados com propriedades fractais.

- **nanoluz** - usada por pesquisadores do Instituto de Óptica Daza Valdés, do CSIC. Consiste no estudo do comportamento da luz em nanopartículas de metal.

A Tabela 5.2 mostra estatísticas referentes ao tempo de execução de tarefas (em minutos) das 4 aplicações obtidas do Ibercivis. Os dados coletados são referentes ao período de 30 de Junho de 2009 a 5 de outubro de 2010. As colunas são referentes ao mínimo, primeiro quartil, mediana, média, terceiro quartil, máximo, desvio padrão (DP) e tamanho da amostra (N).

Aplicação	**Min**	**1ºQuartil**	**Mediana**	**Média**	**3ºQuartil**	**Max**	**DP**	**N**
A1 - adsorcion	0,61	34,6	45,05	59,87	69,79	464,6	48,68	321
A2 - amld-docking	0	17,19	39,17	47,15	64,64	936,2	40,87	637419
A3 - criticalidad	0	12,07	15,12	17,61	19,41	653,5	11,28	406567
A4 - nanoluz	2,6	6,96	7,8	11,6	15,16	66,57	8,25	194

Tabela 5.2: Estatísticas do Tempo de Execução de Tarefas (TET) de 4 aplicações do Ibercivis

A Figura 5.4 apresenta o gráfico-caixa (ou *boxplot*) do Tempo de Execução de Tarefas, em minutos, para cada aplicação do Ibercivis. O retângulo representa o primeiro e terceiro quartis. O ponto no centro do retângulo representa a mediana. Os dados que estão fora dos limites das linhas tracejadas são considerados *outliers*.

Para extrair o modelo para este parâmetro, foi aplicado o ajuste de distribuições de probabilidade descrito na Seção 4.2.4. Para realizar o ajuste, os *outliers* indicados na Figura 5.4 foram excluídos para evitar que valores extremos afetem a qualidade do ajuste [23].

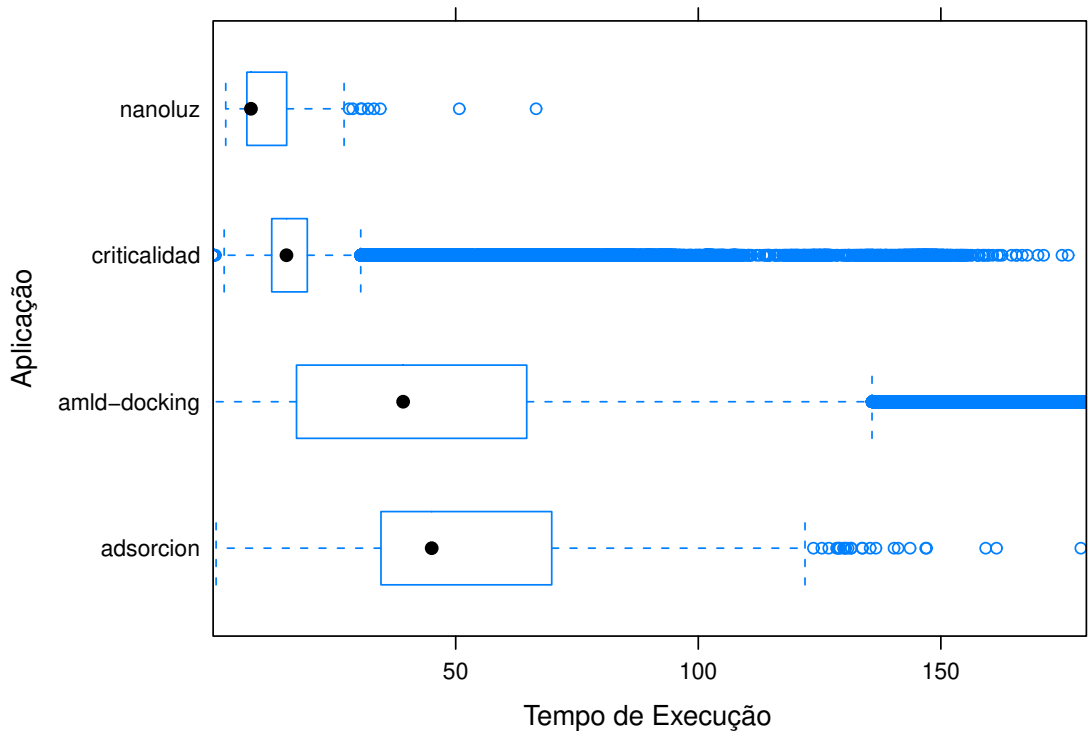


Figura 5.4: Gráfico-caixa do Tempo de Execução de Tarefas para cada aplicação

Verificou-se que o ajuste utilizando a distribuição normal obteve bons resultados de qualidade de ajuste através do teste KS, para as 4 aplicações. A Tabela 5.3 apresenta o resultado do ajuste, onde as colunas se referem ao nome da aplicação, a distribuição usada no ajuste (a normal, no caso), o primeiro parâmetro da distribuição (média da normal), o segundo parâmetro da distribuição (desvio padrão da normal) e, por último, o p-valor resultante do teste KS. Os valores da média e desvio padrão são apresentados em horas.

Aplicação	**Distribuição**	**Média**	**Desvio Padrão**	**p-valor**
A1 - adsorcion	normal	0,790	0,437	0,259
A2 - amld-docking	normal	0,7	0,472	0,237
A3 - criticalidad	normal	0,262	0,08	0,374
A4 - nanoluz	normal	0,173	0,092	0,13

Tabela 5.3: Resultados do ajuste da distribuição normal com os dados do Tempo de Execução de Tarefas

Pode-se observar que todos os p-valores foram superiores ao nível de significância $\alpha = 0,05$ escolhido, o que indica que os ajustes foram satisfatórios para todas as aplicações. Além disso, nota-se que os valores dos parâmetros da distribuição normal para as aplicações A1 e A2 não possuem uma diferença significativa, assim como a diferença entre os valores dos parâmetros das distribuições A3 e A4. Este fato também pode ser observado no gráfico-caixa da Figura 5.4. Com base nisto, e por motivos de simplificação da modelagem, apenas duas aplicações foram utilizadas no modelo. As aplicações A2 e A3 foram priorizadas, visto que o tamanho da amostra coletada é superior às outras.

Então, para o modelo do Tempo de Execução de Tarefas (TET), foram usadas duas distribuições normais: $N(\mu = 0,7; \sigma = 0,472)$, referente à aplicação A2 - amld-docking, e $N(\mu = 0,262; \sigma = 0,08)$, referente à aplicação A3 - criticalidad.

5.2.2 Gerando Carga de Trabalho a Partir do Modelo

Para gerar a carga de trabalho sintética a partir do modelo, é preciso especificar a quantidade de nós que irão compor a grade P2P, além da quantidade de usuários que fazem parte de cada nó do sistema.

Cada nó da grade será associado a um rastro do qual o modelo foi extraído. Ou seja, cada nó será composto por usuários com características obtidas de algum dos sistemas utilizados para a extração do modelo. Na extração do modelo descrito na Seção 4.2, o comportamento dos usuários foi obtido de 6 rastros de grades de serviço. Deste modo, os nós do sistema, gerados sinteticamente, terão 6 possíveis perfis.

Além dos perfis gerais dos nós, os usuários que fazem parte de um mesmo nó também podem possuir diferentes características. Na aplicação do modelo utilizada na Seção 5.2.1, existem 5 perfis (ou grupos) de usuários por cada rastro usado na extração. Ao gerar a carga sintética, cada usuário será associado a um dos 5 grupos. A probabilidade de um usuário ser associado a um certo grupo, é igual à proporção de usuários que faziam parte deste grupo no rastro original.

Cada submissão de job será associado a um dos dois modelos de aplicação descritos, de forma aleatória, onde cada modelo tem a mesma probabilidade de ser utilizado. O tempo de execução de cada tarefa do job será gerado a partir da distribuição que corresponde ao modelo sorteado.

5.3 Simulação com Carga de Trabalho Sintética

Nesta seção, são apresentados os resultados da aplicação do modelo de predição através de simulação, utilizando carga de trabalho sintética gerada a partir do modelo descrito no Capítulo 4.

5.3.1 Cenários

Os cenários de simulação utilizados nesta seção utilizaram o modelo de geração de carga de trabalho sintética apresentado no Capítulo 4. Para gerar carga de trabalho a partir do modelo, é preciso especificar a quantidade de nós que vão compor a grade P2P (n) e a quantidade de usuários que cada nó terá (u). Além disso, é preciso especificar para o simulador a quantidade de recursos que cada nó da grade terá (r).

Os cenários de simulação foram divididos em duas fases. Primeiro, a quantidade de nós da grade foi fixada em $n = 120$ e a quantidade de usuários em $u = 10$, enquanto a quantidade de recursos por nó foi variada em $r = \{10, 20, 40, 100\}$, para analisar o impacto

causado no erro da predição através da variação da oferta de recursos e, consequentemente, da variação da contenção no sistema. O período de simulação foi fixado em 6 meses para todos os cenários.

A Figura 5.5 apresenta a média diária da proporção de tarefas submetidas que foram propriamente alocadas para alguma máquina da grade (OR) para estes 4 cenários. Como esperado, quanto maior a quantidade de recursos disponíveis, maior a proporção de requisições de recursos atendidas. Observa-se que, quando a quantidade de recursos é baixa, são obtidos valores baixos para OR, além de uma grande variação para estes valores. Isto ocorre porque a demanda no sistema se torna maior que a oferta de recursos em vários momentos, causando uma alta contenção no sistema. À medida que a oferta de recursos aumenta, a proporção de recursos obtidos também tende a aumentar, e a variância tende diminuir. Com isto, consegue-se simular com estes 4 cenários o ambiente de grade P2P em diversos momentos de contenção.

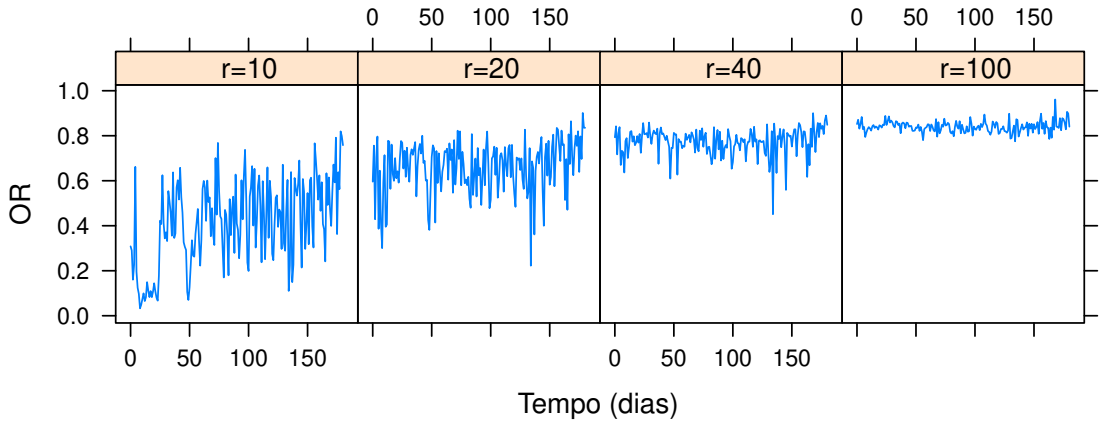


Figura 5.5: Média de OR por dia, para n e u fixos e variando r

Em uma segunda fase, a quantidade de recursos por nó é fixada em $r = 50$, a quantidade de nós na grade é variada em $n = \{30, 60, 120, 240, 480\}$ e a quantidade de usuários por nó é variada em $u = \{5, 10, 20\}$, para avaliar o erro da predição com o aumento da escala da grade P2P. A Figura 5.6 apresenta OR para estes 15 cenários. Acima de cada gráfico aparecem dois números que descrevem o cenário, onde o de cima indica a quantidade de nós no sistema (n) e o de baixo indica a quantidade de usuários por nó (u). Os valores de

u aumentam da esquerda para a direita, enquanto os valores de n aumentam de baixo para cima.

À medida que a quantidade de nós participantes da grade P2P diminui (gráficos no sentido de cima para baixo), maior é a variância do OR. Isto ocorre porque quando um nó do sistema muda de estado de consumidor para doador ou vice-versa, isto é, quando um usuário submete um novo job ou quando os jobs em fila terminam de ser executados, o impacto causado na contenção da grade será baixo quando existirem muitos participantes na grade, e será alto quando existirem poucos participantes, pois a proporção de máquinas afetadas com uma mudança de estado do nó será maior nestes casos. Por outro lado, mesmo aumentando a quantidade de nós no sistema, a contenção da grade não aumenta pois a quantidade de recursos também aumenta proporcionalmente.

O impacto na contenção da grade causado pelo aumento da quantidade de usuários por nó do sistema (gráficos no sentido da esquerda para a direita) é menor, resultando em uma pequena diminuição na média de OR à medida que u aumenta, pois a demanda da grade será maior, o que aumenta a contenção do sistema.

5.3.2 Resultados

Os resultados da predição da grade para os cenários descritos acima são apresentados a seguir. Inicialmente, é apresentado um resumo geral dos resultados. Em seguida, são apresentados os erros da predição para os cenários em que é variada a quantidade de recursos por nó do sistema. Finalmente, são apresentados os resultados para os cenários em que são variadas a quantidade de nós no sistema e a quantidade de usuários por nó, de acordo com os cenários descritos na Seção 5.3.1.

Cenários variando a quantidade de recursos por nó

Em um primeiro conjunto de cenários avaliados, a quantidade de nós no sistema é fixada em $n = 120$ e a quantidade de usuários por nó é fixada em $u = 10$. A Figura 5.7 mostra gráficos-caixa do erro absoluto na predição para os cenários variando r em $\{10, 20, 40, 100\}$, para cada estratégia de predição. A linha no centro de cada caixa representa a mediana dos dados, enquanto as bordas esquerda e direita representam o primeiro e terceiro quartis,

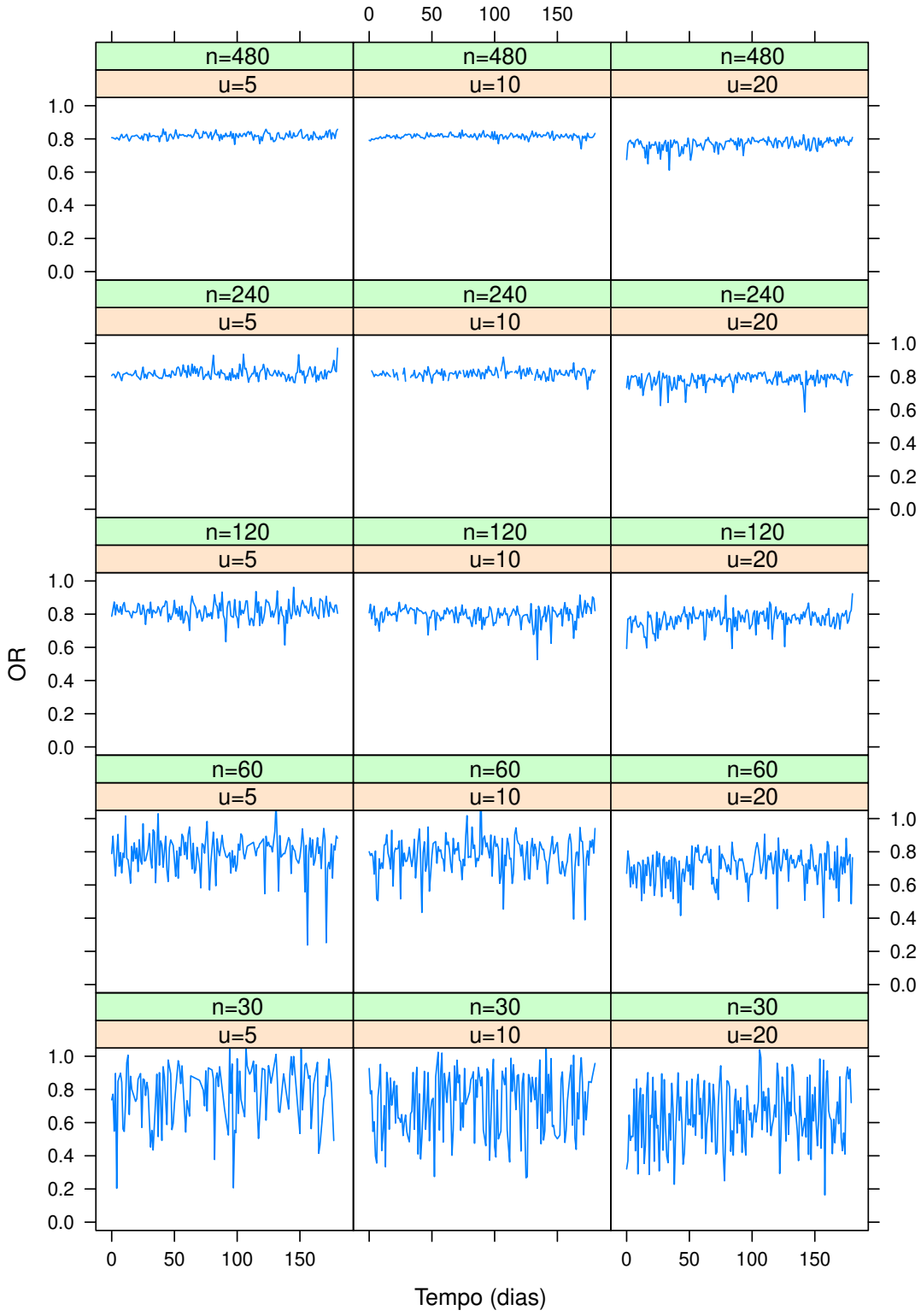


Figura 5.6: Média de OR por dia, para r fixo e variando n e u

respectivamente. As linhas horizontais nos lados das caixas representam o intervalo entre o primeiro quartil até 1,5*IQR antes dele e do terceiro quartil até 1,5*IQR depois dele, onde IQR (*Interquartile range*) é a diferença entre o terceiro e o primeiro quartis. Foi adicionada aos gráficos-caixa a média nos erros da predição, representados por um ponto. Para verificar mais detalhes destes resultados, uma tabela com os valores apresentados na figura pode ser vista no Apêndice C.

Pode-se observar através da Figura 5.7 que as estratégias *NoF-Based* e *NoF-Based.CF* tendem a obter médias e medianas de erro mais baixas à medida que a quantidade de recursos por nó aumenta (ou seja, a contenção no sistema diminui), apesar deste decaimento ser sutil. O mesmo comportamento é observado para as outras estratégias, porém com decaimento mais significativo. Observa-se também que o tamanho das caixas (IQR) é menor à medida que a quantidade de recursos por nó aumenta. Isto significa que, quanto maior a contenção no sistema, maior a dispersão dos erros. Desta forma, os erros de predição tendem a ser menores e com menor variância quanto menor for a contenção no sistema.

Além disso, nota-se que as linhas que representam as medianas nem sempre estão no centro das caixas, e que os pontos que representam as médias nem sempre estão próximos às linhas das medianas. Isto significa que a distribuição dos erros de predição tendem a ser enviesados e a não seguir uma distribuição normal. Para confirmar esta tendência, foi executado o teste de normalidade *Anderson-Darling* [21], com nível de significância de 5%, para todos os cenários. O resultado confirmou que, para este nível de significância, a hipótese de que os valores dos erros de predição para cada cenário seguem uma distribuição normal foi rejeitada, confirmando a tendência observada.

Como os erros da predição não são normalmente distribuídos, foi utilizado um teste de hipótese não-paramétrico para verificar qual estratégia apresenta os melhores resultados para os cenários apresentados. Como os valores dos erros nas predições para cada estratégia são pareados, pois as predições para cada estratégia foram realizadas nas mesmas condições, foi utilizado o teste *Wilcoxon signed-rank*, que é o teste não-paramétrico equivalente ao teste t de Student para dados pareados.

Primeiro, utilizou-se para este teste a hipótese nula de que a mediana da diferença dos pares de erros de 2 estratégias A e B para cada predição é zero, com nível de significância de 5%. Foi utilizada como referência a estratégia *NoF-Based.CF*, aplicando-se o teste para

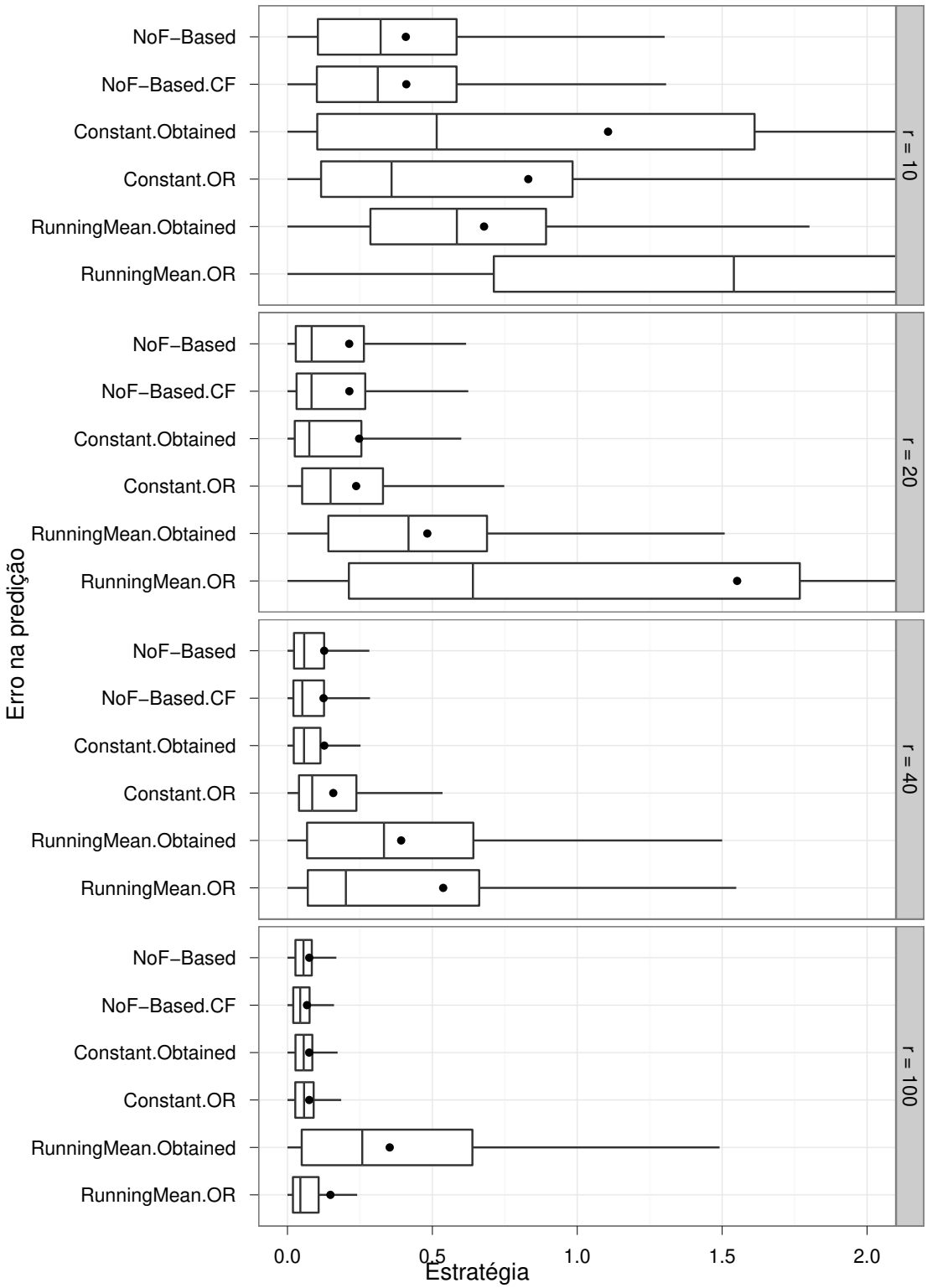


Figura 5.7: Gráfico-caixa para os erros na predição para $n = 120$, $u = 10$ e variando r

todas as outras estratégias com relação a ela. O resultado da aplicação do teste foi que a hipótese nula foi rejeitada para todas as estratégias em todos os cenários. Isto significa que a diferença entre os erros de predição da estratégia *NoF-Based.CF* com relação às outras estratégias foi significativo em todos os cenários.

Como a hipótese nula anterior foi rejeitada para todos os cenário e estratégias em relação à estratégia *NoF-Based.CF*, ou seja, a mediana da diferença dos erros é significantemente diferente de zero, é preciso saber se essa diferença é positiva ou negativa para determinar se a estratégia *NoF-Based.CF* é superior ou inferior a cada uma das outras estratégias para cada cenário. Para isto, o teste *Wilcoxon signed-rank* foi aplicado novamente, agora com a hipótese nula de que a mediana da diferença dos pares de erros da estratégia *NoF-Based.CF* com relação a cada uma das outras estratégias para cada predição é significantemente *maior* que zero, com nível de significância de 5%. A hipótese nula não foi rejeitada apenas quando a estratégia *NoF-Based.CF* foi comparada com a estratégia *NoF-Based* no cenário para $r = 20$, ou seja, a mediana da diferença dos erros da predição da estratégia *NoF-Based.CF* com relação a todas as outras estratégias é significativamente menor que zero, com exceção da estratégia *NoF-Based* no cenário em que $r = 20$ no qual a mediana é significativamente maior que zero. Isto significa que os erros de predição da estratégia *NoF-Based.CF*, quando comparados pareados com os erros das outras estratégias para cada predição nos cenários avaliados, tendem a apresentar menores valores que as outras estratégias, com exceção da estratégia *NoF-Based* que tende a possuir menores erros no cenário onde $r = 20$.

Algumas estatísticas para a melhor estratégia de predição em cada cenário variando r, de acordo com os testes de hipótese realizados, são apresentadas na Tabela 5.4.

r	**Estratégia**	**1ºQuart**	**Mediana**	**Média**	**3ºQuart**	**Max**	**DP**	**IQR**
10	NoF-Based.CF	10.14	31.17	41.01	58.36	580.00	46.91	48.21
20	NoF-Based	2.87	8.36	21.31	26.38	400.30	36.29	23.51
40	NoF-Based.CF	2.05	5.13	12.45	12.62	280.90	21.87	10.57
100	NoF-Based.CF	2.00	4.39	6.74	7.63	124.00	8.85	5.63

Tabela 5.4: Estatísticas do erro (em %) para as melhores estratégias variando r

A Figura 5.8 apresenta a média do erro da predição para diferentes tamanhos de janela de predição, comparando o erro das estratégias descritas na Seção 5.1. As barras em cada

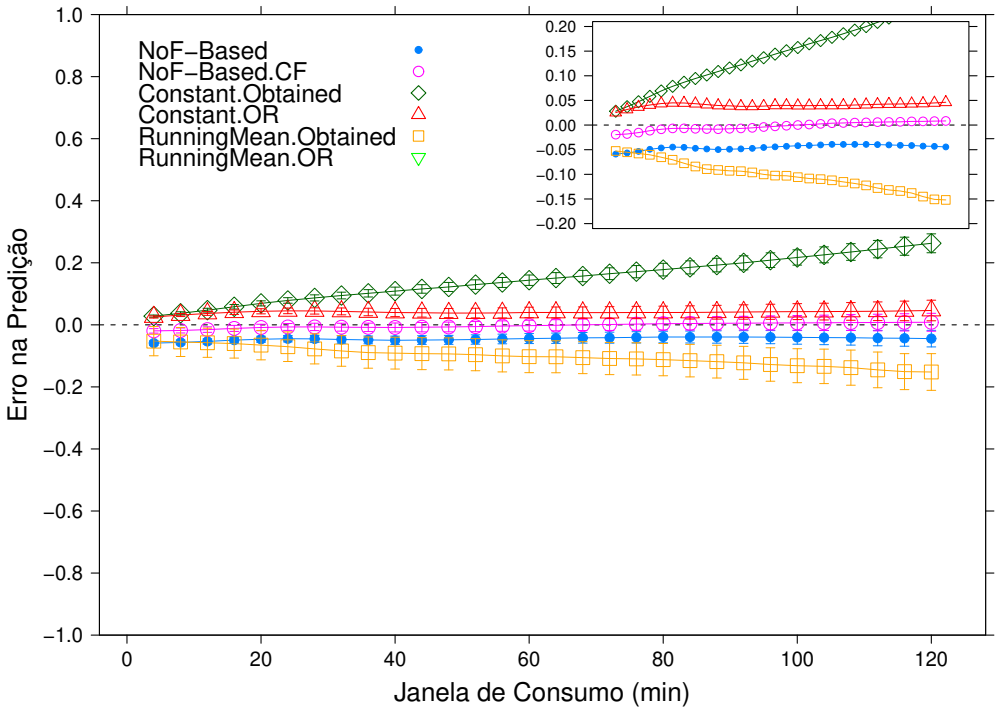
ponto indicam os intervalos de confiança das médias, para um nível de confiança de 95%. Para melhor visualização, o gráfico foi limitado para valores de janela de consumo entre 2 e 120 minutos. Além disso, foi inserido um gráfico interno em cada figura, que apresenta o mesmo resultado, porém sem as barras de intervalo de confiança e com uma aproximação em uma faixa menor de valores para o erro da predição ([-0,2 ; 0,2]), permitindo uma melhor comparação das estratégias que obtiveram médias no erro da predição próximas de zero. Quanto mais próximos os pontos estão dos centros dos gráficos, representados pela linha tracejada no erro igual a zero, menor é a média do erro da predição.

Verifica-se através dos gráficos que, na maioria dos cenários, a média dos erros das estratégias *NoF-Based* e *NoF-Based.CF* não sofrem uma alteração muito ampla à medida que a janela de consumo aumenta. Este fato não é observado para as outras estratégias, onde a média do erro da predição tende a ter uma variação mais significativa à medida que a janela de consumo aumenta, na maioria dos casos.

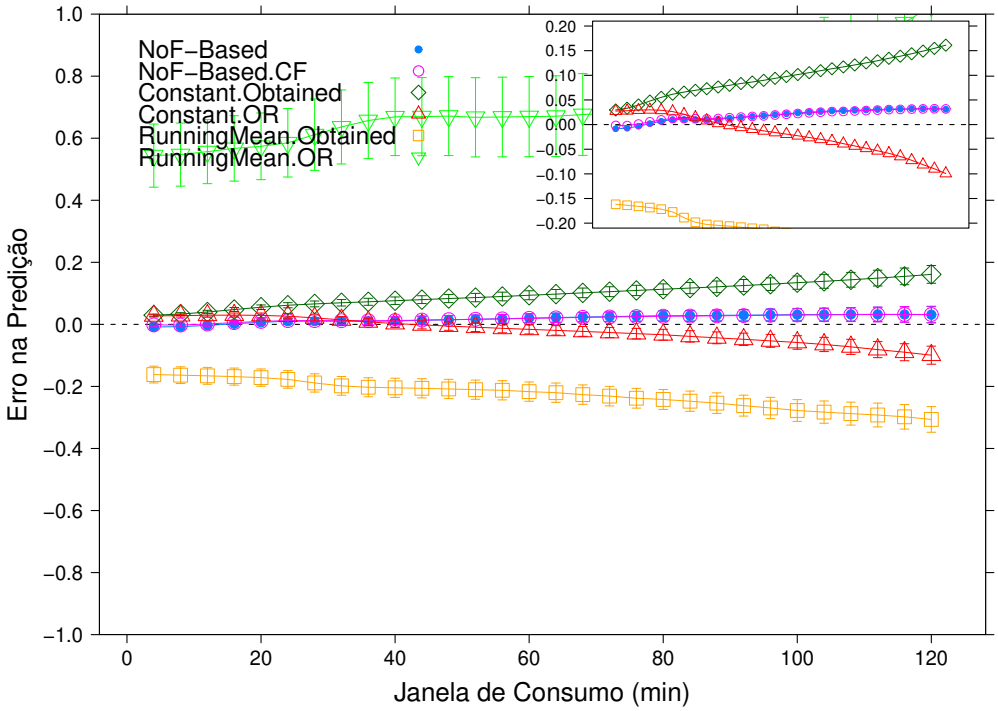
Cenários variando a quantidade de nós no sistema e quantidade de usuários por nó

Em um segundo conjunto de cenários avaliados, a quantidade de recursos por nó do sistema é fixada em $r = 50$, enquanto a quantidade de nós na grade é variada em $n = \{30, 60, 120, 240, 480\}$ e a quantidade de usuários por nó é variada em $u = \{5, 10, 20\}$. As Figuras 5.9, 5.10 e 5.10 mostram gráficos-caixa do erro absoluto na predição para u igual a 5, 10 e 20, respectivamente.

Observa-se nos gráficos que, para as estratégias *NoF-Based* e *NoF-Based.CF*, ao aumentar a quantidade de nós na grade, a média e mediana do erro tendem a ser ligeiramente maiores (consultar Tabela C.2, na página 94, para observar os valores com mais detalhes). Isto ocorre porque, a taxa de submissão de tarefas na grade é maior à medida que se aumenta o valor de n, existindo então muitas transições de estados dos nós para valores altos de n. Observa-se também que a média tende a ser maior que a mediana para cenários com valores baixo de n, diminuindo esta diferença à medida que n aumenta. Isto ocorre porque a quantidade total de recursos no sistema também tem um aumento proporcional à quantidade de nós na grade. Desta forma, apesar de existirem muitas transições de estados dos nós para valores altos de n, o impacto da redistribuição de recursos não é tão grande, pois a quantidade de recursos também será alta, então a proporção de recursos afetados não será tão grande

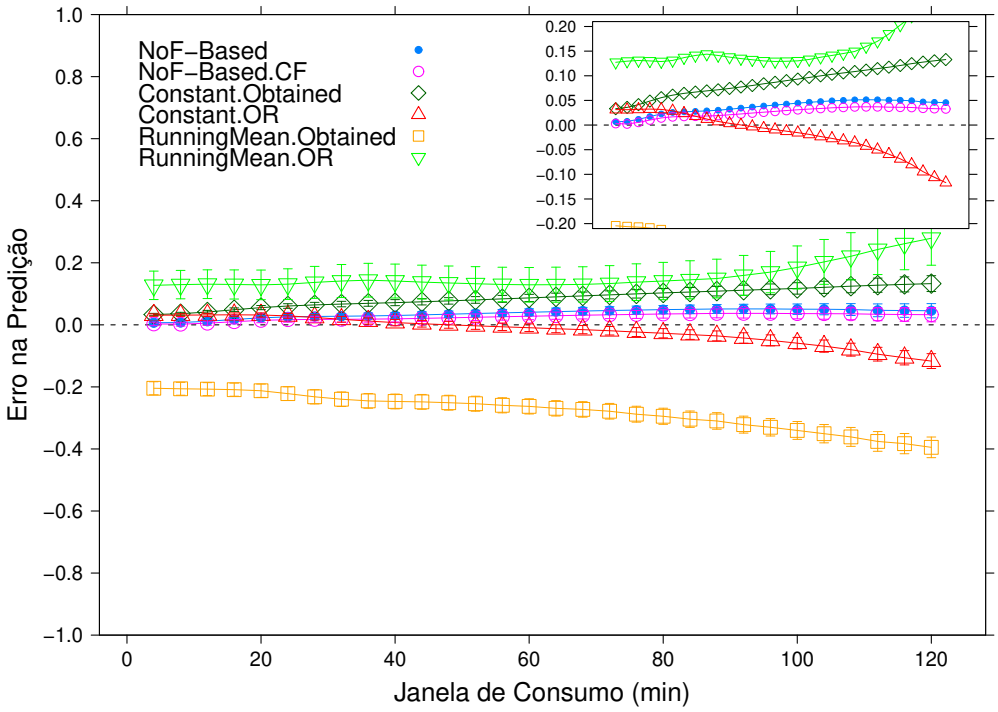


(a) $n = 120$, $u = 10$, $r = 10$

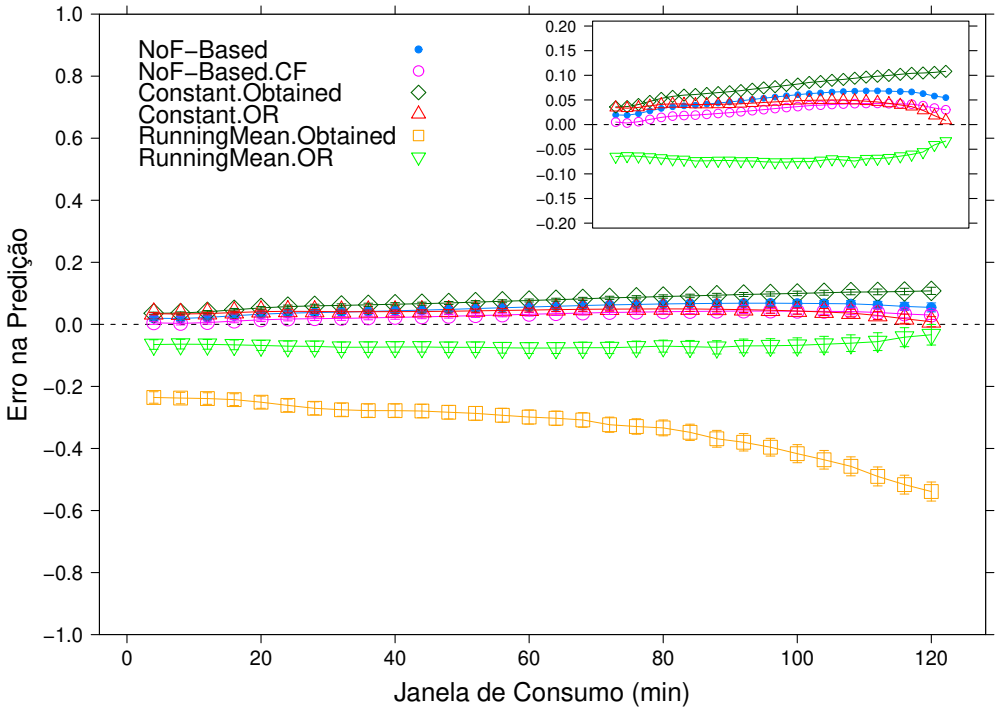


(b) $n = 120$, $u = 10$, $r = 20$

Figura 5.8: Erros na predição variando r em $\{10, 20\}$ e Δ de 2 a 120 minutos



(c) $n = 120$, $u = 10$, $r = 40$



(d) $n = 120$, $u = 10$, $r = 100$

Figura 5.8: Erros na predição variando r em $\{40, 100\}$ e Δ de 2 a 120 minutos

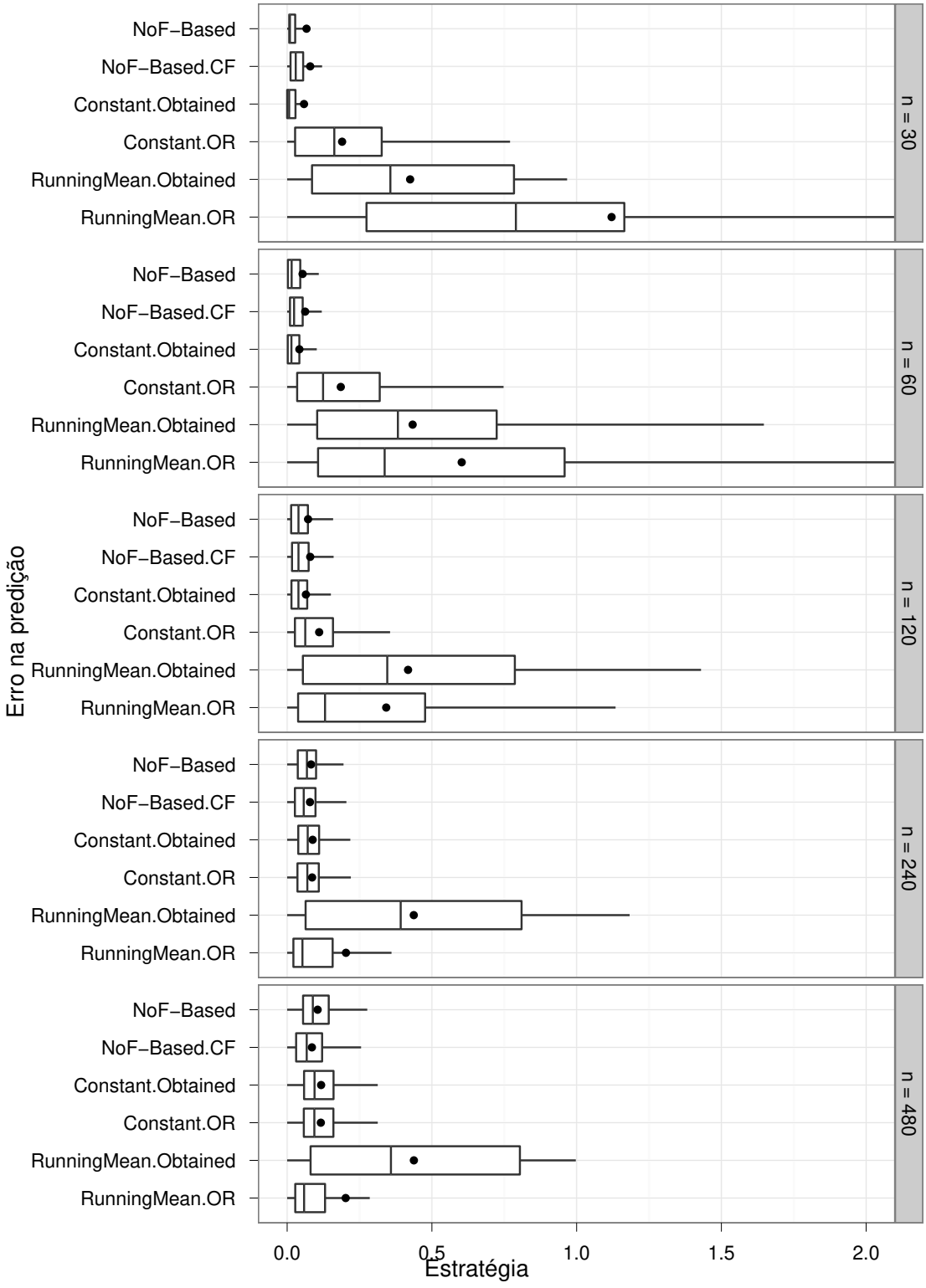


Figura 5.9: Gráfico-caixa para os erros na predição para $r = 50$, $u = 5$ e variando n

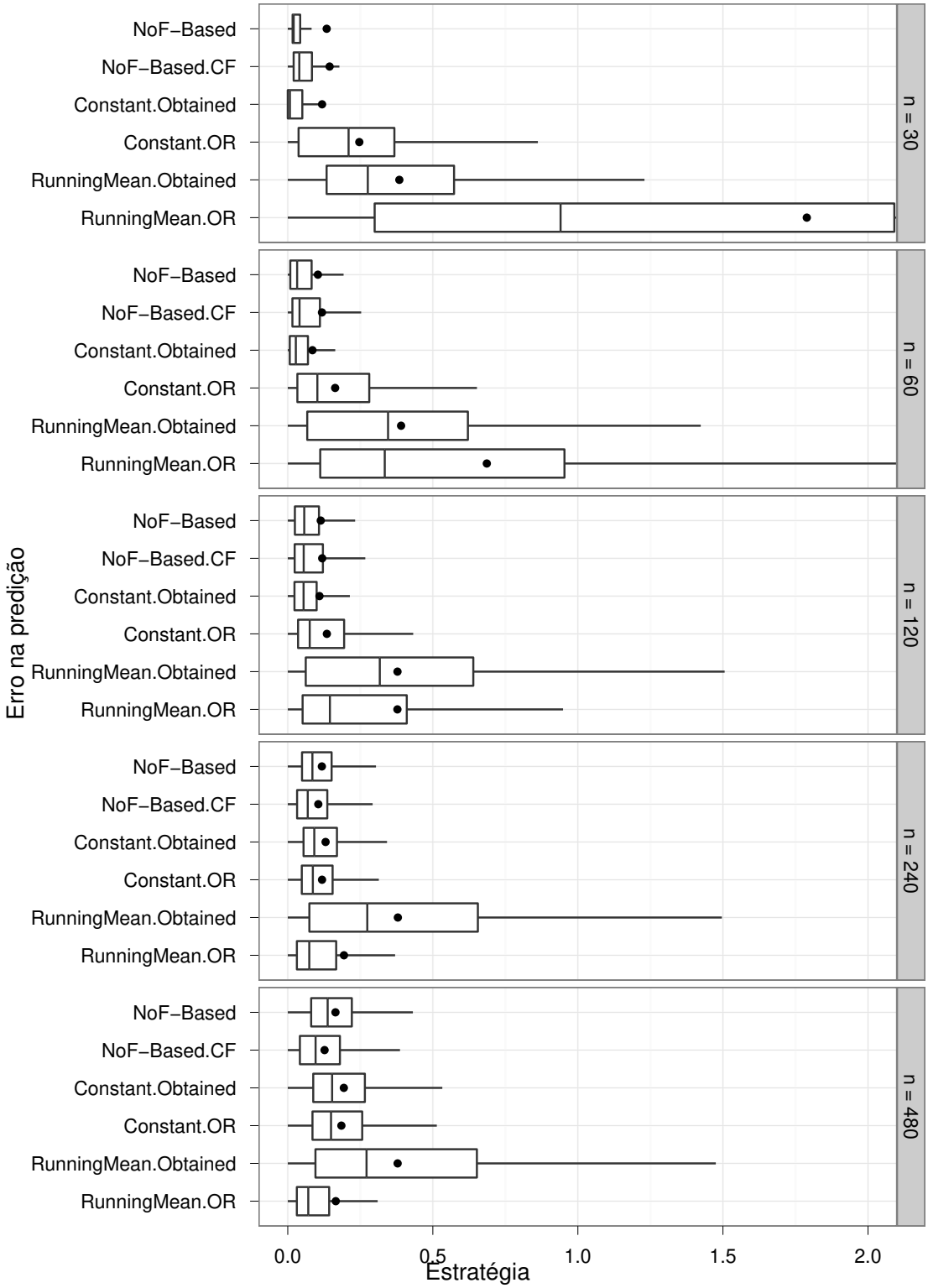


Figura 5.10: Gráfico-caixa para os erros na predição para $r = 50$, $u = 10$ e variando n

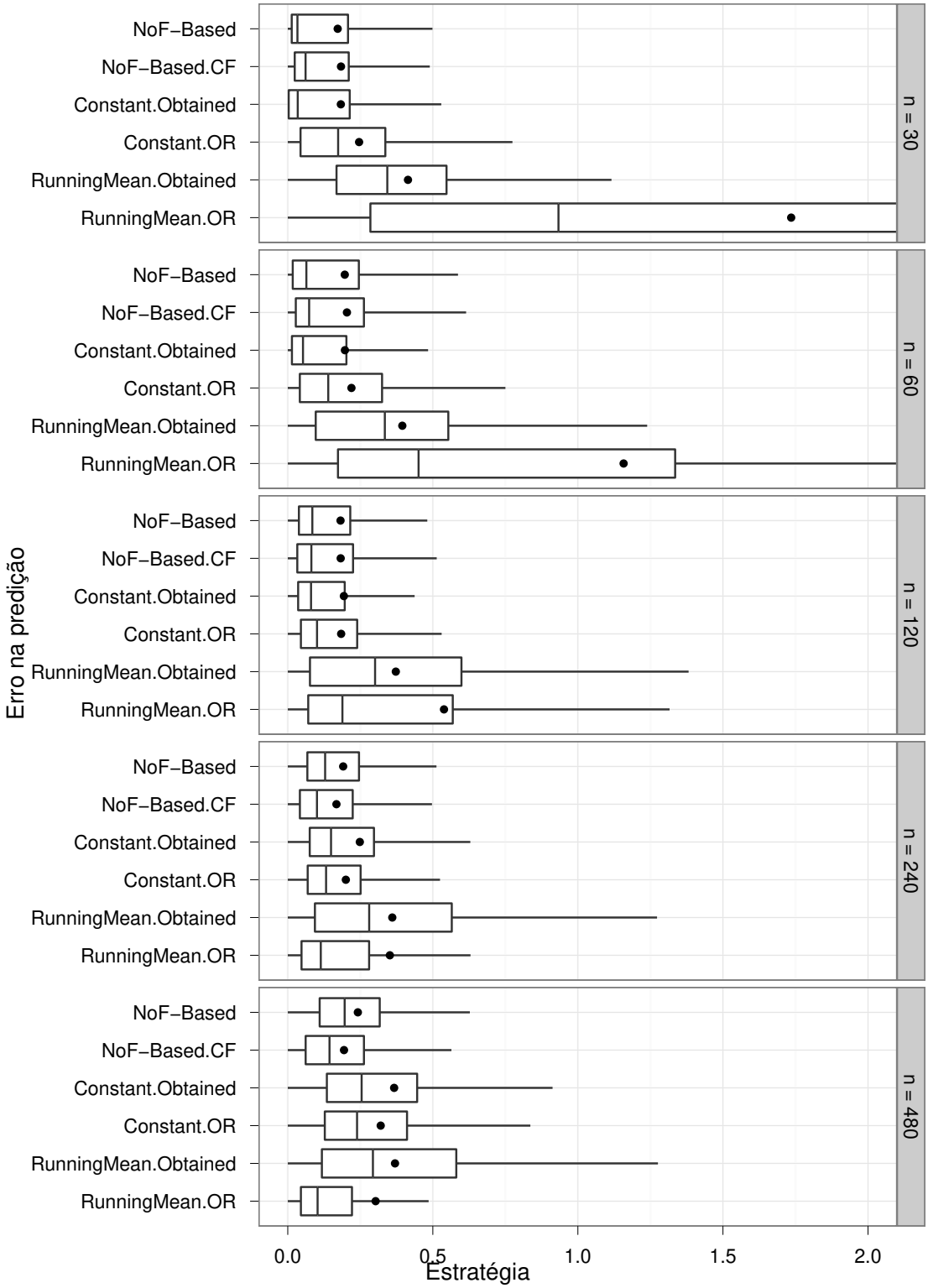


Figura 5.11: Gráfico-caixa para os erros na predição para $r = 50$, $u = 20$ e variando n

na maioria dos casos, o que diminui a ocorrência de valores extremos e consequentemente diminui a média, aproximando-a da mediana. Além disso, a dispersão dos erros tende a ser maior ao aumentar o n, como pode ser visto observando o tamanho das caixas (IQR). Isto ocorre porque a quantidade de submissões de tarefas, e consequentemente a quantidade de transições de estado dos nós, aumenta à medida que a quantidade de nós no sistema aumenta, aumentando a dispersão nos erros das predições.

Também pode-se observar através do gráfico que as estratégias *NoF-Based* e *NoF-Based.CF* tendem a obter médias e medianas dos erros mais altas à medida que a quantidade de usuários por nó (u) aumenta. Além disso, a dispersão nos erros da predição também tende a crescer ao aumentar u, como pode ser visto ao observar o tamanho das caixas (IQR). Isto ocorre porque, quanto maior é a quantidade de usuários por nó, maior é a taxa de submissão de jobs para a grade, como pode ser observado na Figura 5.12, olhando os gráficos da esquerda para a direita e observando o aumento na densidade das barras das tarefas submetidas. Uma maior taxa de submissão resulta em uma maior quantidade de mudanças de estados dos nós, de consumidor para doador e vice-versa. Estas mudanças de estado dos nós são a principal fonte de erros na predição.

Quando um nó que inicialmente é doador submete um novo job, passando a ser consumidor, ele requisita seus recursos que estão doados de volta, causando uma perda de recursos para os nós consumidores. Além disso, este nó também pode requisitar mais recursos da grade, o que vai gerar uma redistribuição dos recursos dos outros doadores de acordo com a NoF, causando uma perda ainda maior de recursos por parte dos outros consumidores. Isto pode causar erros na predição, pois este decréscimo na quantidade de recursos para os nós consumidores nem sempre é previsto.

Por outro lado, quando os jobs de um nó consumidor terminam de executar, este nó passa a ser um nó doador. Os seus próprios recursos passarão a ficar disponíveis para a grade, além dos recursos de outros doadores que este nó estava utilizando anteriormente. Desta forma, novos recursos ficarão disponíveis para a grade e serão redistribuídos para os consumidores. Isto também pode causar erros na predição, pois o aumento na quantidade de recursos para os nós também pode não ser previsto.

Assim como nos cenários anteriores, os erros da predição também não seguem uma distribuição normal. Da mesma forma, o teste *Wilcoxon signed-rank* foi utilizado para estes

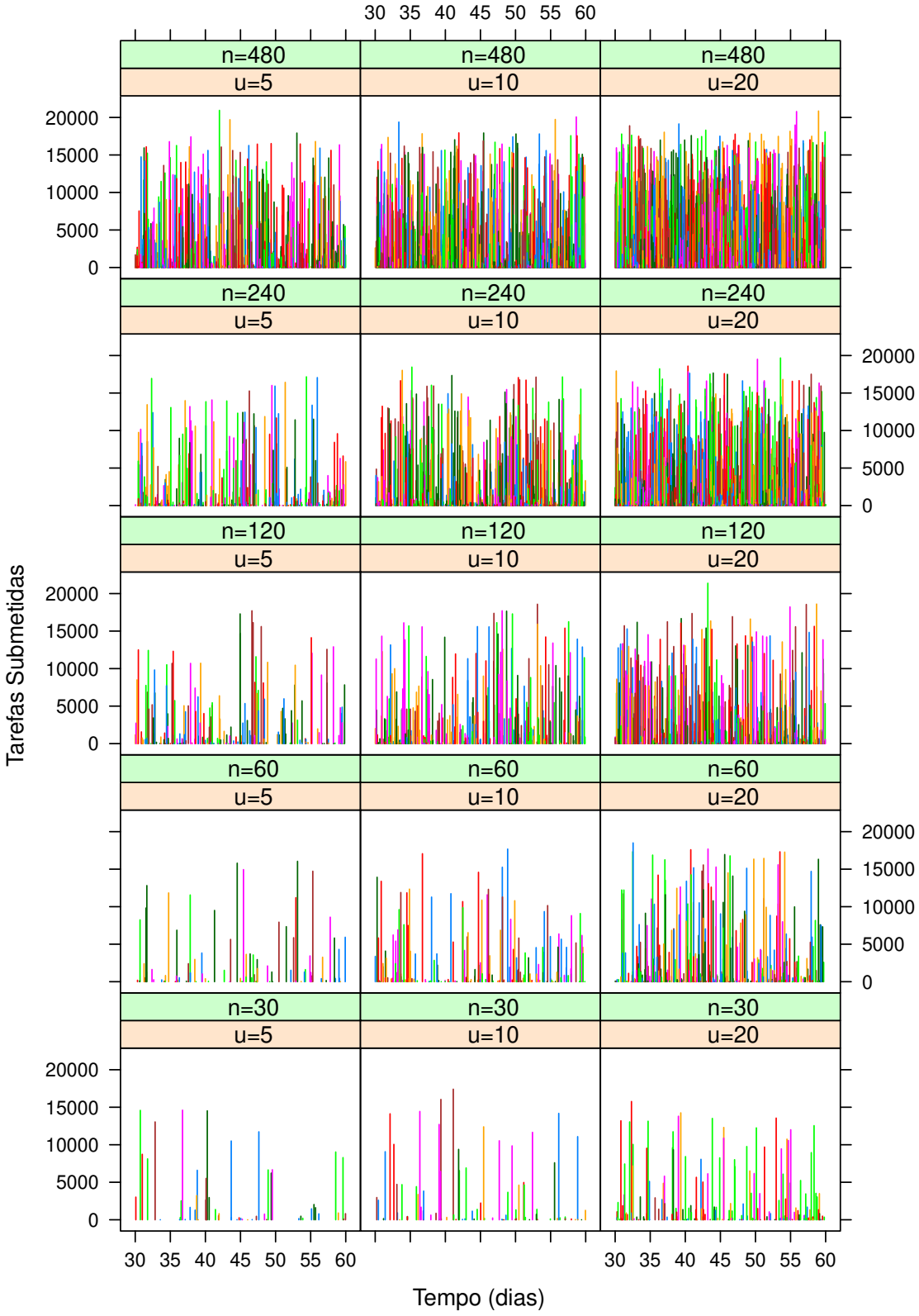


Figura 5.12: Tarefas submetidas por cada nó da grade durante um mês

cenários, utilizando a mesma metodologia dos cenários anteriores. A Tabela 5.5 apresenta estatísticas do erro de predição (em porcentagem) para as melhores estratégias em cada cenário, obtidas através dos testes de hipótese.

n	u	**Estratégia**	**1ºQuart**	**Mediana**	**Média**	**3ºQuart**	**Max**	**DP**	**IQR**
30	5	Constant.Obt	0.00	0.70	5.82	2.88	183.20	19.22	2.88
30	10	Constant.Obt	0.00	0.74	11.82	5.01	256.10	34.76	5.01
30	20	Constant.Obt	0.32	3.43	18.29	21.36	340.70	36.33	21.03
60	5	Constant.Obt	0.21	1.48	4.23	4.22	167.90	10.59	4.01
60	10	Constant.Obt	0.68	2.78	8.50	6.96	161.50	18.28	6.27
60	20	Constant.Obt	1.44	5.25	19.70	20.21	360.80	36.87	18.77
120	5	Constant.Obt	1.48	3.90	6.48	6.93	221.20	13.41	5.45
120	10	NoF-Based.CF	2.38	5.49	11.87	12.12	264.40	19.74	9.74
120	20	NoF-Based.CF	3.28	8.13	18.24	22.50	701.50	32.99	19.22
240	5	NoF-Based.CF	2.68	5.72	7.88	9.80	74.01	8.21	7.11
240	10	NoF-Based.CF	3.22	6.88	10.52	13.63	239.30	11.71	10.41
240	20	NoF-Based.CF	4.18	10.06	16.78	22.39	347.10	20.96	18.21
480	5	NoF-Based.CF	3.10	6.76	8.55	12.08	78.47	7.27	8.98
480	10	RunningMean.OR	3.13	7.07	16.55	14.27	1507.00	32.04	11.14
480	20	RunningMean.OR	4.52	10.28	30.28	22.13	1792.00	73.41	17.61

Tabela 5.5: Estatísticas do erro (em %) para as melhores estratégias variando u e n

Assim como os resultados observados para os cenários variando o r, a média do erro da estratégia *NoF-Based* também não sofre uma alteração muito significativa à medida que a janela de consumo aumenta para os cenários variando o n e o u. Também se repete o fato da média do erro da predição para as outras estratégias tenderem a ter uma variação mais significativa à medida que a janela de consumo aumenta, na maioria dos casos. Como este comportamento já foi verificado para as figuras com o r variando, as figuras para várias janelas de consumo com o n e u variando foram colocadas no Apêndice B.

5.4 Simulação com Rastros

Nas simulações realizadas nesta seção, foram utilizados registros obtidos da execução de grades computacionais reais em produção, disponibilizados no *Grid Workloads Archive* (GWA)

[27].

5.4.1 Cenários

Dentre os registros das grades disponíveis, escolhemos o NorduGrid [53] por apresentar as características mais adequadas para o propósito deste trabalho: um alto número de *sites* participantes (75), uma alta quantidade de tarefas submetidas (781.370), e a maioria de suas aplicações comportando-se como BoTs.

A carga de trabalho utilizada foi construída a partir de informações dos registros da grade, como: o tempo de submissão, o tempo de execução, o número de processadores requisitados e a identificação do nó de origem, para cada tarefa. Alguns filtros foram aplicados, de forma que na carga de trabalho utilizada fiquem apenas: (i) tarefas submetidas durante uma janela de 6 meses a partir da data de inicio de produção da grade; (ii) nós que possuem ao menos um job submetido em mais do que 25% dos meses durante o período de produção da grade (nós que quase nunca requisitam recursos da grade não são representativos para ambientes de grades P2P); e (iii) tarefas que requisitam apenas um único processador. Após a aplicação do filtro descrito acima, a carga de trabalho final continha 207.926 tarefas submetidas, a partir de 40 nós diferentes durante seis meses.

A carga de trabalho, que define a demanda do sistema, foi definida pelos rastros. Foi realizada um varredura em vários valores para a quantidade de recursos por nó, com a intenção de analisar diferentes cenários de contenção de recursos. Foram utilizados 20, 40, 60 e 80 recursos por nó. Em cada simulação executada, considerou-se que os nós possuem a mesma quantidade de recursos para cada cenário. É importante notarmos que r não afeta a quantidade requisitada de recursos pelos nós consumidores, apenas a oferta de recursos.

A Figura 5.13 apresenta uma média da fração de recursos requisitados que foram realmente obtidos da grade (OR), durante todo o período de simulação. Como podemos verificar pelos gráficos, a variação do OR é alta durante o tempo de simulação, principalmente para cenários de alta contenção por recursos ($r = 20$).

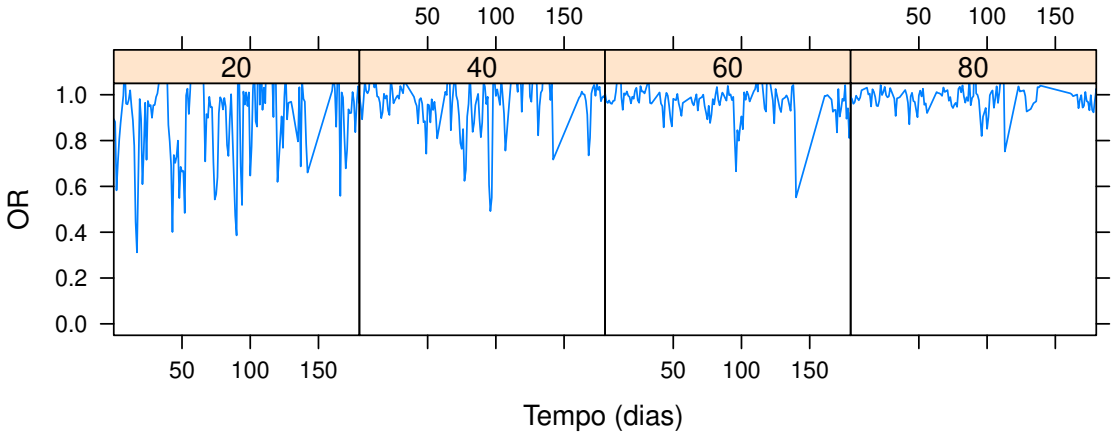


Figura 5.13: Média da proporção de recursos obtidos em relação ao total requisitado

5.4.2 Resultados

A Figura 5.14 mostram gráficos-caixa do erro absoluto na predição para estes cenários. Pode-se observar através do gráfico que as estratégias *NoF-Based* e *NoF-Based.CF* tendem a obter médias e medianas dos erros de predição menores à medida que aumenta a quantidade de recursos por nó. Além disso, a dispersão dos erros também diminui à medida que r aumenta, como pode ser observado através do tamanho das caixas (IQR). Isto ocorre porque, quanto maior a quantidade de recursos, menor a competição por recursos e, consequentemente menor a contenção no sistema. O aumento da quantidade da oferta de recursos por nó r traz um impacto ainda mais visível nas outras estratégias, que tendem a ter erros menores à medida que o r aumenta.

Assim como nos cenários de simulação com carga sintética, os erros da predição também não seguem uma distribuição normal. Da mesma forma, o teste *Wilcoxon signed-rank* foi utilizado para estes cenários, utilizando a mesma metodologia dos cenários anteriores. A Tabela 5.6 apresenta estatísticas do erro de predição (em porcentagem) para as melhores estratégias em cada cenário, obtidas através dos testes de hipótese.

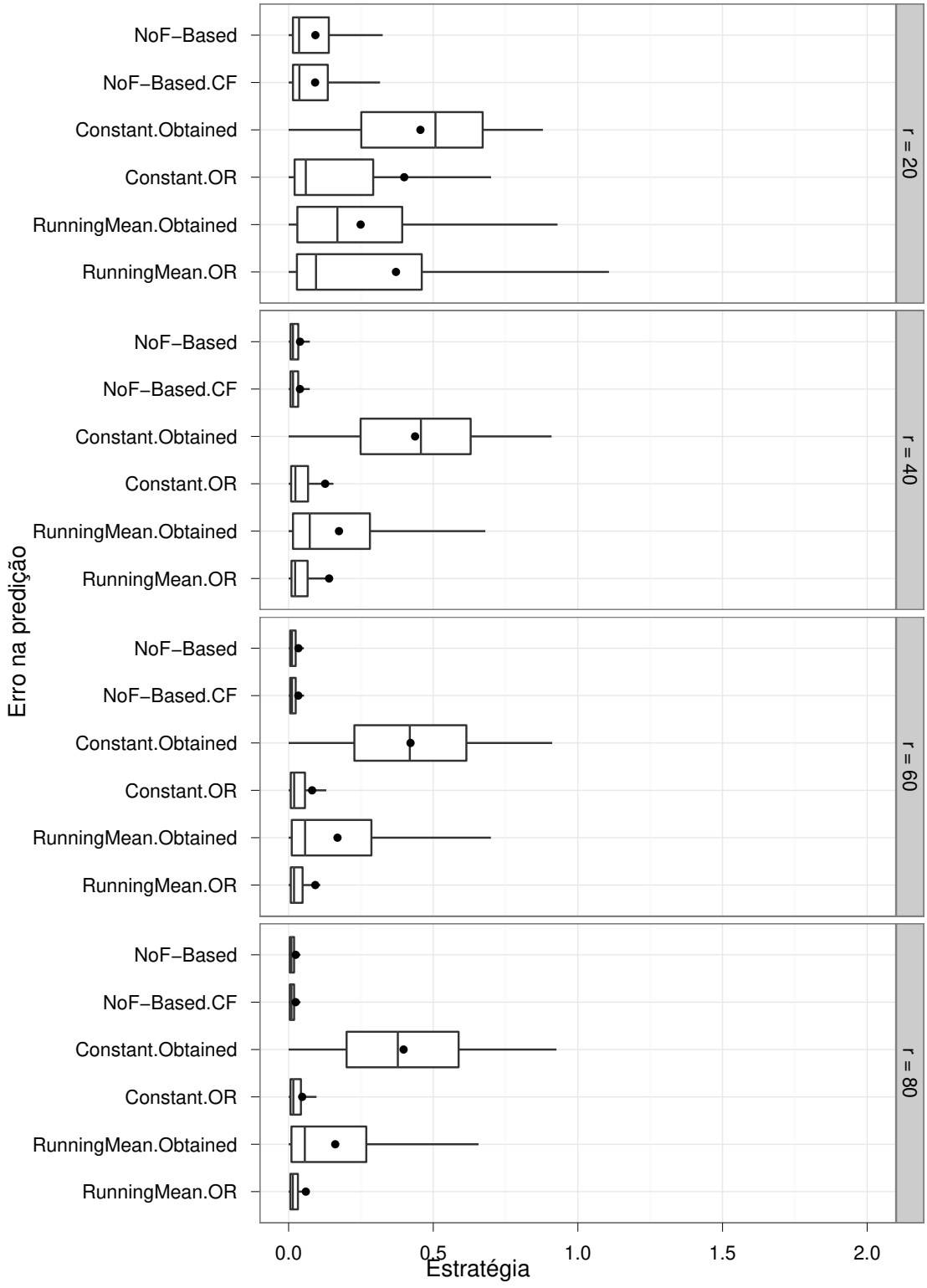


Figura 5.14: Gráfico-caixa para os erros na predição variando r

r	**Estratégia**	**1ºQuart**	**Mediana**	**Média**	**3ºQuart**	**Max**	**DP**	**IQR**
20	NoF-Based.CF	1.50	3.73	9.17	13.55	59.84	11.33	12.04
40	NoF-Based.CF	0.66	1.53	3.91	3.33	71.89	7.29	2.66
60	NoF-Based.CF	0.53	1.18	3.34	2.46	53.62	6.97	1.92
80	NoF-Based	0.41	0.96	2.44	1.88	54.68	5.72	1.47

Tabela 5.6: Estatísticas do erro (em %) para as melhores estratégias variando r

5.5 Discussão

Os resultados dos erros das predições para as simulações usando carga de trabalho sintética foram divididos em duas fases, uma variando a quantidade de recursos por nó (r) e a outra variando tanto a quantidade de nós no sistema (n) quanto a quantidade de usuário (u).

Nos cenários em que o r foi variado, observa-se que, através da metologia adotada, as predições que utilizam de algum conhecimento do sistema (caixa-cinza e caixa-branca) obtiveram os melhores resultados nos 4 cenários avaliados, sendo a *NoF-Based.CF* a melhor estratégia em 3 cenários e a *NoF-Based* em 1 (ver Tabela 5.4).

Já nos 15 cenários em que n e u foram variados, houve um maior balanço entre as estratégias, sendo as predições caixa-preta as melhores estratégias para 9 cenários (7 *Constant.Obtained* e 2 *RunningMean.OR*), enquanto a predição caixa-cinza (*NoF-Based.CF*) foi a melhor em 6 cenários (ver Tabela 5.5).

Observa-se que a estratégia *Constant.Obtained* é a melhor quando a quantidade de nós no sistema é baixa, ou seja, quando a quantidade de submissões de tarefas é reduzida e, consequentemente, a frequencia de transições de estados dos nós é menor. Porém, ela apresenta piores desempenhos quando a quantidade de nós no sistema é muito grande, ou quando a quantidade de recursos por nó é muito baixa. Ou seja, ela não apresenta bons resultados quando a contenção no sistema é alta. Por outro lado, a estratégia *RunningMean.OR* só apresenta bons resultados quando a quantidade de nós no sistema é alto. Em todos os cenários em que foi variado o r e pra cenários com valores médios para n e u, as estratégias *NoF-Based* e *NoF-Based.CF* foram as melhores estratégias.

A Tabela 5.7 apresenta estatísticas para as médias dos erros de predição obtidas nos 19 cenários avaliados com carga de trabalho sintética, para cada estratégia de predição.

Estratégia	Min	Mediana	Média	Max	DP	IQR
NoF-Based	5,27	12,51	14,66	40,67	8,27	9,19
NoF-Based.CF	6,18	12,29	14,32	40,84	8,03	9,91
Constant.Obtained	4,15	12,46	19,45	110,00	23,34	10,69
Constant.OR	7,34	18,09	20,76	82,66	16,18	10,10
RunningMean.Obtained	35,06	39,03	41,27	67,74	7,19	5,09
RunningMean.OR	14,78	53,55	82,73	377,10	89,29	88,25

Tabela 5.7: Estatísticas da média e mediana dos erros (em %)

Observa-se que a estratégia *Constant.Obtained* apesar de ser a melhor estratégia em 7 cenários (ver Tabela 5.5), pode obter valores muito altos de média, como foi o caso da média do erro de 110% no cenário em que $r = 10$, $n = 120$ e $u = 10$ (consultar Tabela C.1, na página 92). Além disso, o desvio padrão das médias também foi alto, o que indica uma alta variação na média dos erros para os diferentes cenários.

A estratégia *RunningMean.OR*, apesar de ser a melhor estratégia em 2 cenários (ver Tabela 5.5), apresenta valores muito altos para a média e mediana das médias de todos os cenários, além de um desvio padrão também bastante acentuado. Deste modo, esta estratégia só apresenta bons resultados para aqueles 2 cenários particulares, apresentando maus resultados no geral.

A estratégia *NoF-Based* é a melhor estratégia em 1 cenário e a *NoF-Based.CF* é a melhor em 10 cenários (ver Tabelas 5.4 e 5.5). Elas possuem um baixo desvio padrão para as médias dos cenários, com relação às outras estratégias, indicando uma baixa variação na média dos erros para diferentes cenários. Além disso, apresentam as menores médias das médias dos erros.

Para as simulações usando rastros, as estratégias *NoF-Based* e *NoF-Based.CF* foram as que obtiveram melhores resultados (ver Tabela 5.6). A diferença dessas estratégias para as outras fica ainda mais evidenciada quando a quantidade de recursos por nó é baixa, pois as outras estratégias apresentam resultados bem inferiores. Quando a oferta de recursos é alta, a maioria das estratégias apresentam bons resultados.

Deste modo, se o ambiente de grade P2P no qual o modelo de predição será aplicado

for desconhecido ou se variar bastante com o tempo, as estratégias *NoF-Based* ou *NoF-Based.CF* são as mais indicadas, por sofrerem menos impacto no erro das predições em diferentes condições da grade, além de apresentarem baixos erros de predição no geral. Se as condições da grade forem estáveis e bem conhecidas, também podem ser utilizados modelos caixa-preta como a estratégia *Constant.Obtained*, que apresenta bons resultados quando a contenção no sistema é baixa, ou a *RunningMean.OR*, que apresenta bons resultados quando a quantidade de nós no sistema e a quantidade de usuários por nó é alta. Quando a quantidade de recursos por nó é baixa, as estratégias *NoF-Based* e *NoF-Based.CF* se mostraram superiores em relação às outras.

Capítulo 6

Conclusões e Trabalhos Futuros

Nesta capítulo são apresentadas as conclusões do trabalho e trabalhos futuros que podem ser desenvolvidos.

6.1 Conclusões

Neste trabalho, são apresentados modelos para a predição da capacidade de processamento que um nó participante de uma grade computacional P2P é capaz de obter em períodos futuros. Um novo modelo de predição caixa-branca é proposto, com base no funcionamento interno do sistema. Outros modelos de predição caixa-preta, que utilizam apenas dados históricos da interação do participante do sistema com a grade, são adaptados da literatura para o problema em estudo. Além disso, é apresentado um novo modelo caixa-cinza, que combina o modelo caixa-branca proposto com técnicas estatísticas obtidas de modelos caixa-preta.

Para avaliar os modelos de predição apresentados, foi proposto um modelo hierárquico de geração de carga de trabalho sintética para grades computacionais P2P, que é baseado em diferentes comportamentos de usuários e diferentes características de aplicações, que sejam representativas para o tipo de ambiente em estudo. Foi discutido porque um modelo em que são usadas distribuições diferentes para modelar o comportamento de diferentes grupos de usuários, como o modelo proposto neste trabalho, é mais adequado do que usar apenas uma distribuição de probabilidade para descrever o comportamento de todos os usuários de um sistema.

Os resultados da avaliação mostraram que, no geral, as estratégias que utilizam conhecimento do funcionamento do sistema (caixa-cinza e caixa-branca) tendem a apresentar menores erros na predição, de acordo com os cenários avaliados. Algumas estratégias caixa-preta apresentaram bons resultados para cenários específicos, porém elas tendem a ser sensíveis às mudanças de característica do ambiente, como por exemplo a mudança na contenção do sistema. Por outro lado, as predições caixa-cinza e caixa-branca possuem uma menor variação no erro da predição quando se varia a quantidade de nós no sistema, a quantidade de recursos por nó ou a quantidade de usuários por nó, apresentando baixos erros em todos os cenários.

Também foi constatado que quanto maior a quantidade de recursos por nó do sistema, menores os erros da predição tendem a ser. Por outro lado, quanto maior a quantidade de nós no sistema e quanto maior a quantidade de usuários por nó, maiores os erros da predição tendem a ser.

Variando os cenários de simulação em condições diversas, a média dos erros absolutos de predição obtida para todos os cenários avaliados foi entre 2,43% e 41% para a estratégia *NoF-Based.CF*. O que mostra que é possível realizar predição da capacidade de processamento em uma grade P2P, sujeita a diferentes cargas de trabalho e ofertas de recursos, e apresentar erros de predição não muito significativos, sem apresentar uma grande variação no erro da predição ao expor a grade a diferentes cenários de contenção e escala.

6.2 Trabalhos Futuros

Os trabalhos futuros podem ser divididos em três categorias: (i) predição da qualidade de serviço, (ii) geração de carga de trabalho sintética para grades P2P e (iii) experimentos em um ambiente real de grade P2P.

6.2.1 Predição da Qualidade de Serviço

Os modelos de predição apresentados podem ser utilizados para estimar outros atributos de qualidade de serviço de grades computacionais, como o tempo de resposta, taxa de falha de tarefas, disponibilidade, etc.

Além disso, o modelo de predição caixa-branca, baseado no funcionamento do sistema,

pode ser adaptado para outros sistemas em que em que são utilizados mecanismos de incentivo baseado na reciprocidade dos participantes, como por exemplo o compartilhamento de arquivos em sistemas P2P.

O modelo caixa-branca apresentado pode ser extendido, fazendo com que a estimativa dos parâmetros do modelo seja feita de uma forma mais elaborada e que algumas simplificações do modelo sejam removidas. Este modelo também pode ser avaliado em um ambiente em que as informações fornecidas pelo sistema não são confiáveis ou são imprecisas.

6.2.2 Geração de Carga de Trabalho Sintética para Grades P2P

O modelo de geração de carga de trabalho sintética pode ser refinado, permitindo uma maneira dinâmica de obter a quantidade ideal de grupos para cada atributo do modelo. O ajuste das distribuições do modelo também pode ser efetuado com uma quantidade maior de distribuições candidatas. Também podem ser usados diferentes rastros de sistemas para efetuar a extração do modelo. Além disso, uma validação do modelo com rastros de uma grade computacional P2P, mesmo com limitações de utilização, pode ser realizada.

6.2.3 Experimentos em um Ambiente Real de Grade P2P

Os modelos de predição podem ser aplicados em tempo de execução e avaliados em um sistema de grade P2P real, através de experimentos. A carga de trabalho sintética pode ser usada para gerar uma carga em uma ambiente de grade controlado, criado para a realização dos experimentos.

Bibliografia

[1] Karl Aberer and Zoran Despotovic. Managing trust in a peer-2-peer information system. In *Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM)*, pages 310–317, 2001.

[2] Nazareno Andrade, Francisco Brasileiro, Walfredo Cirne, and Miranda Mowbray. Discouraging free riding in a peer-to-peer CPU-sharing Grid. *High-Performance Distributed Computing, International Symposium on*, 0:129–137, 2004.

[3] Nazareno Andrade, Francisco Brasileiro, Walfredo Cirne, and Miranda Mowbray. Automatic grid assembly by promoting collaboration in peer-to-peer grids. *Journal of Parallel and Distributed Computing*, 67(8):957–966, 2007.

[4] Nazareno Andrade, Walfredo Cirne, Francisco Brasileiro, and Paulo Roisenberg. Ourgrid: An approach to easily assemble grids with equitable resource sharing. In *Proceedings of the 9th Workshop on Job Scheduling Strategies for Parallel Processing*, Seattle, WA, USA, June 2003.

[5] Nazareno Andrade, Miranda Mowbray, Walfredo Cirne, and Francisco Brasileiro. When can an autonomous reputation scheme discourage free-riding in a peer-to-peer system? In *CCGRID '04: Proceedings of the 2004 IEEE International Symposium on Cluster Computing and the Grid*, pages 440–448, Washington, DC, USA, 2004. IEEE Computer Society.

[6] Artur Andrzejak, Derrick Kondo, and David P. Anderson. Ensuring collective availability in volatile resource pools via forecasting. In *DSOM '08*, pages 149–161, 2008.

[7] Cosimo Anglano, John Brevik, Massimo Canonico, Dan Nurmi, and Rich Wolski. Fault-aware scheduling for bag-of-tasks applications on desktop grids. In *Proceedings*

of the 7th IEEE/ACM International Conference on Grid Computing, GRID '06, pages 56–63, Washington, DC, USA, 2006. IEEE Computer Society.

[8] E. C. Araujo, W. Cirne, G. Mendes, N. Oliveira, E. P. Souza, C. Galvao, and E. S. Martins. The SegHidro Experience: Using the Grid to Empower a Hydro-Meteorological Scientific Network. In *Proc. of the 1st IEEE International Conference on e-Science and Grid Computing (e-Science 2005)*, Melbourne, Australia, 2005.

[9] Francisco Brasileiro, Lauro B. Costa, Alisson Andrade, Walfredo Cirne, Sujoy Basu, and Sujata Banerjee. A large scale fault-tolerant grid information service. In *MGC '06: 4th International Workshop on Middleware for Grid Computing*, 2006.

[10] Francisco Brasileiro, Alexandre Duarte, Rafael Silva, and Matheus Gaudencio. On the co-existence of service and opportunistic grids. In *First EELA-2 Conference*, 2009.

[11] Francisco Brasileiro, Matheus Gaudencio, Rafael Silva, Alexandre Duarte, Diego Carvalho, Diego Scardaci, Leandro Ciuffo, Rafael Mayo, Herbert Hoeger, Michael Stanton, Raul Ramos, Roberto Barbera, Bernard Marechal, and Philippe Gavillet. Using a simple prioritisation mechanism to effectively interoperate service and opportunistic grids in the eela-2 e-infrastructure. *Journal of Grid Computing*, pages 1–17, 2011. 10.1007/s10723-010-9177-5.

[12] J. Brevik, D. Nurmi, and R. Wolski. Automatic methods for predicting machine availability in desktop grid and peer-to-peer systems. In *CCGRID '04: Proceedings of the 2004 IEEE International Symposium on Cluster Computing and the Grid*, pages 190–199, 2004.

[13] Sergio Camorlinga and B. Schofield. Modeling of workflow-engaged networks on radiology transfers across a metro network. *IEEE Transactions on Information Technology in Biomedicine*, 10(2):275–281, 2006.

[14] Marcus Carvalho, Eliane Araújo, Renato O. Fernandes, and Rodolfo L. B. Nóbrega. Producing and sharing regional weather forecast data for e-science applications. In *eScience '08: Proceedings of the 2008 Fourth IEEE International Conference on eScience*, pages 348–349, Washington, DC, USA, 2008. IEEE Computer Society.

[15] Marcus Carvalho, Renato Miceli, Paulo Ditarso Maciel Jr., Francisco Brasileiro, and Raquel Lopes. Predicting the quality of service of a peer-to-peer desktop grid. In *4th Workshop on Desktop Grids and Volunteer Computing Systems (PCGrid). In proceedings of the 10th IEEE/ACM International Symposium on Cluster Computing and the Grid*, volume 0, pages 649–654, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[16] W. Cirne and F. Berman. A comprehensive model of the supercomputer workload. *Workload Characterization, Annual IEEE International Workshop*, 0:140–148, 2001.

[17] W. Cirne, F. Brasileiro, N. Andrade, L. Costa, A. Andrade, R. Novaes, and M. Mowbray. Labs of the world, unite!!! *Journal of Grid Computing*, 2006.

[18] Walfredo Cirne, Francisco Brasileiro, Daniel Paranhos, Luís Fabrício W. Góes, and William Voorsluys. On the efficacy, efficiency and emergent behavior of task replication in large distributed systems. *Journal of Parallel Computing*, 33:213–234, April 2007.

[19] Andy Cooke, Alasdair Gray, Werner Nutt, James Magowan, Manfred Oevers, Paul Taylor, Roney Cordenonsi, Steve Fisher, Steve Hicks, Jason Leake, Robin Middleton, Xiaomei Zhu, Norbert Podhorszki, and Brian Coghlan Stuart Kenny. The relational grid monitoring architecture: Mediating information about the grid. *Journal of Grid Computing*, 2, 2004.

[20] Karl Czajkowski, Carl Kesselman, Steven Fitzgerald, and Ian Foster. Grid information services for distributed resource sharing. *HPDC '01*, 2001.

[21] R. B. D'agostino and M. S. Stephen, editors. *Goodness-of-Fit Techniques*. Marcel Dekker, New York and Basel, 1986.

[22] Dror Feitelson. Parallel Workloads Archive, Acessado em agosto de 2010. http://www.cs.huji.ac.il/labs/parallel/workload.

[23] Dror Feitelson. Workload modeling for computer systems performance evaluation, Book draft acessado em agosto de 2011. http://www.cs.huji.ac.il/ feit/wlmod/.

[24] Marcelo Finger, Germano C. Bezerra, and Danilo R. Conde. Resource use pattern analysis for opportunistic grids. In *MGC '08: 6th international workshop on Middleware for grid computing*, 2008.

[25] J. A. Hartigan and M. A. Wong. A K-means clustering algorithm. *Applied Statistics*, 28:100–108, 1979.

[26] Alexandru Iosup, Mathieu Jan, Ozan Sonmez, and Dick Epema. The characteristics and performance of groups of jobs in grids. In *In Euro-Par, volume 4641 of LNCS*, pages 382–393. Springer-Verlag, 2007.

[27] Alexandru Iosup, Hui Li, Mathieu Jan, Shanny Anoep, Catalin Dumitrescu, Lex Wolters, and Dick H. J. Epema. The Grid Workloads Archive. *Future Generation Computing Systems*, 24(7):672–686, 2008.

[28] Alexandru Iosup, Ozan Sonmez, Shanny Anoep, and Dick Epema. The performance of bags-of-tasks in large-scale distributed systems. In *Proceedings of the 17th International Symposium on High Performance Distributed Computing*, HPDC '08, pages 97–108, New York, NY, USA, 2008. ACM.

[29] M. Irving, G. Taylor, and P. Hobson. Plug into grid computing. *Power and Energy Magazine, IEEE*, 2(2):40–44, Mar-Apr 2004.

[30] Bahman Javadi, Derrick Kondo, Jean-Marc Vincent, and David P. Anderson. Discovering statistical models of availability in large distributed systems: An empirical study of seti@home. *IEEE Transactions on Parallel and Distributed Systems*, 99(PrePrints), 2011.

[31] Carl Kesselman and Ian Foster. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, November 1998.

[32] D. Kondo, A. Andrzejak, and D. P. Anderson. On correlated availability in internet-distributed systems. In *GRID '08: Proceedings of the 2008 9th IEEE/ACM International Conference on Grid Computing*, 2008.

[33] Derrick Kondo, Gilles Fedak, Franck Cappello, Andrew A. Chien, and Henri Casanova. Characterizing resource availability in enterprise desktop grids. *Future Generation Computer Systems*, 23(7):888–903, 2007.

[34] Derrick Kondo, Bahman Javadi, Alexandru Iosup, and Dick Epema. The failure trace archive: Enabling comparative analysis of failures in diverse distributed systems. *Cluster Computing and the Grid, IEEE International Symposium on*, 0:398–407, 2010.

[35] Derrick Kondo, Michela Taufer, Charles L. Brooks III, Henri Casanova, and Andrew A. Chien. Characterizing and evaluating desktop grids: An empirical study. *Parallel and Distributed Processing Symposium, International*, 1:26b, 2004.

[36] E. Korpela, D. Werthimer, D. Anderson, J. Cobb, and M. Leboisky. Seti@home-massively distributed computing for seti. *Computing in Science & Engineering*, 3(1):78–83, 2001.

[37] Richard Kuntschke, Tobias Scholl, Sebastian Huber, Alfons Kemper, Angelika Reiser, Hans-Martin Adorf, Gerard Lemson, and Wolfgang Voges. Grid-Based Data Stream Processing in e-Science. In *E-SCIENCE '06: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing*, page 30, Washington, DC, USA, 2006. IEEE Computer Society.

[38] Uri Lublin and Dror G. Feitelson. The workload on parallel supercomputers: modeling the characteristics of rigid jobs. *J. Parallel Distrib. Comput.*, 63:1105–1122, November 2003.

[39] Paulo Ditarso Maciel Jr., Flavio Figueiredo, David Candeia, Francisco Brasileiro, and Álvaro Coêlho. On the Planning of a Hybrid IT Infrastructure. *IEEE/IFIP Network & Operations Management Symposium (NOMS'08)*, April 2008.

[40] Andrew Stephen McGough, Asif Akram, Li Guo, Marko Krznaric, Luke Dickens, David Colling, Janusz Martyniak, Roger Powell, Paul Kyberd, and Constantinos Kotsokalis. GRIDCC: real-time workflow system. In *WORKS '07: Proceedings of the 2nd workshop on Workflows in support of large-scale science*, pages 3–12, New York, NY, USA, 2007. ACM.

[41] Harvey B. Newman, Mark H. Ellisman, and John A. Orcutt. Data-intensive e-science frontier research. *ACM Communications*, 46(11):68–77, 2003.

[42] Daniel Nurmi, John Brevik, and Richard Wolski. Modeling machine availability in enterprise and wide-area distributed computing environments. In *Euro-Par*, pages 432–441, 2005.

[43] Nelson Nóbrega-Júnior, Leonardo Assis, and Francisco Brasileiro. Scheduling cpu-intensive grid applications using partial information. *International Conference on Parallel Processing*, 0:262–269, 2008.

[44] OurGrid Team. OurGrid status. http://status.ourgrid.org.

[45] Beth Plale, Dennis Gannon, Daniel A. Reed, Sara J. Graves, Kelvin Droegemeier, Robert Wilhelmson, and Mohan Ramamurthy. Towards Dynamically Adaptive Weather Analysis and Forecasting in LEAD. In *Computational Science - ICCS 2005, 5th International Conference, Atlanta, GA, USA, May 22-25, 2005, Proceedings, Part II*, pages 624–631, 2005.

[46] Antonio Plaza, David Valencia, Javier Plaza, and Pablo Martinez. Commodity cluster-based parallel processing of hyperspectral imagery. *Journal of Parallel and Distributed Computing*, 66(3):345–358, 2006.

[47] A. A. Sawchuk, E. Chew, R. Zimmermann, C. Papadopoulos, and C. Kyriakakis. From Remote Media Immersion to Distributed Immersive Performance. In *ETP '03: Proceedings of the 2003 ACM SIGMM workshop on Experiential telepresence*, pages 110–120, New York, NY, USA, 2003. ACM.

[48] Edi Shmueli and Dror G. Feitelson. On simulation and design of parallel-systems schedulers: Are we doing the right thing? *IEEE Transactions on Parallel and Distributed Systems*, 20:983–996, 2009.

[49] Ozan Sonmez, Nezih Yigitbasi, Alexandru Iosup, and Dick Epema. Trace-based evaluation of job runtime and queue wait time predictions in grids. In *HPDC '09*, pages 111–120, 2009.

[50] CERN Tridas Team. Control and monitor the high energy physics experiments.

[51] Condor Team. Hawkeye - a monitoring and management tool for distributed systems. http://www.cs.wisc.edu/condor/hawkeye.

[52] Ibercivis team. Ibercivis - plataforma de computación ciudadana, Acessado em agosto de 2010. http://www.ibercivis.es/.

[53] NorduGrid Team. Nordugrid. http://www.nordugrid.org.

[54] The Grid Workloads Archive Team. Grid Workloads Archive, Acessado em agosto de 2010. http://gwa.ewi.tudelft.nl/.

[55] Reinhold P. Weicker. An overview of common benchmarks. *Computer*, 23:65–75, December 1990.

[56] Li Weigang, Marcos V. Pinheiro Dib, and Daniel Amaral Cardoso. Grid Service Agents for Real Time Traffic Synchronization. In *WI '04: Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 619–623, Washington, DC, USA, 2004. IEEE Computer Society.

[57] Julia Zilber, Ofer Amit, and David Talby. What is worth learning from parallel workloads? a user and session based analysis. In *Proceedings of the 19th annual international conference on Supercomputing*, ICS '05, pages 377–386, New York, NY, USA, 2005. ACM.

Apêndice A

Distribuições de Probabilidade do Modelo de Carga de Trabalho

Grupo	Rastro	Distribuição	Param1	Param2	Tamanho
1	GS1	lognormal	2,19	0,20	154
2		weibull	3,96	11,64	81
3		weibull	8,53	12,26	6
4		gamma	19,19	1,95	2
5		gamma	21,60	1,42	2
6	GS2	lognormal	2,46	0,29	72
7		gamma	9,59	0,64	57
8		normal	14,13	3,54	24
9		lognormal	2,28	0,24	10
10		gamma	63,77	7,57	2
11	GS3	gamma	10,24	0,81	83
12		lognormal	2,34	0,25	47
13		weibull	3,90	16,09	39
14		lognormal	2,19	0,18	32
15		gamma	26,62	2,10	3
16	GS4	weibull	3,88	12,51	62
17		gamma	16,86	1,12	38
18		normal	13,06	1,79	17
19		lognormal	2,24	0,19	9
20		exponential	0,13	NA	5
21	GS5	weibull	3,56	13,06	80
22		gamma	46,32	4,19	59
23		exponential	0,12	NA	44
24		normal	15,02	3,64	41
25		lognormal	2,27	0,24	22
26	GS6	gamma	32,58	3,14	72
27		gamma	46,22	4,00	58
28		gamma	36,59	2,85	22
29		gamma	1864075758,21	145474275,75	15
30		gamma	29,92	3,22	2

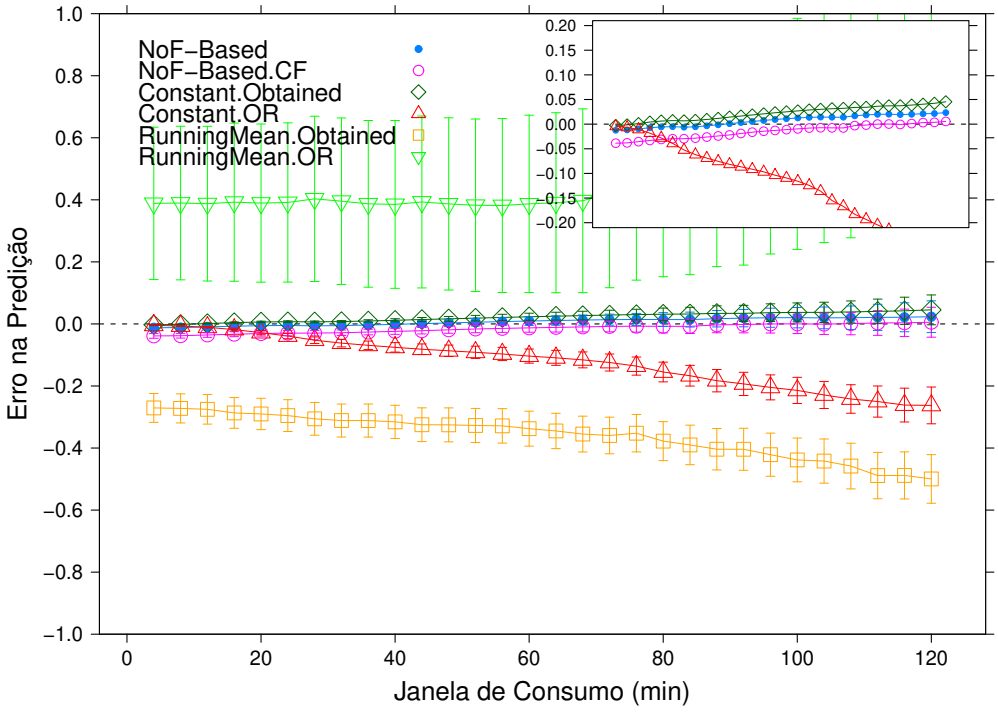
Tabela A.1: Distribuições de probabilidade para o Intervalo Entre Submissões de jobs

Grupo	Rastro	Distribuição	Param1	Param2	Tamanho
1	GS1	gamma	7,26	0,92	129
2		weibull	2,24	6,13	73
3		gamma	4,01	1,82	33
4		gamma	8,65	0,86	2
5		gamma	6,30	0,52	2
6	GS2	normal	10,69	3,20	152
7		normal	12,88	3,53	55
8		weibull	3,37	12,82	28
9		normal	8,56	3,66	11
10		exponential	0,57	NA	9
11	GS3	gamma	14,37	0,68	184
12		weibull	3,84	15,21	25
13		normal	17,55	3,90	20
14		lognormal	2,19	0,36	7
15		lognormal	1,99	0,20	1
16	GS4	normal	11,80	5,00	144
17		gamma	8,49	1,01	71
18		weibull	4,47	16,07	23
19		lognormal	1,82	0,37	2
20		exponential	0,26	NA	1
21	GS5	lognormal	2,53	0,39	154
22		lognormal	2,04	0,46	51
23		gamma	7,49	0,50	24
24		weibull	3,10	10,96	9
25		lognormal	1,81	0,31	4
26	GS6	gamma	9,36	0,92	93
27		gamma	33,77	2,92	72
28		normal	7,07	2,38	52
29		normal	14,48	3,81	34
30		gamma	59,13	3,87	5

Tabela A.2: Distribuições de probabilidade para o Tempo de Execução de Jobs

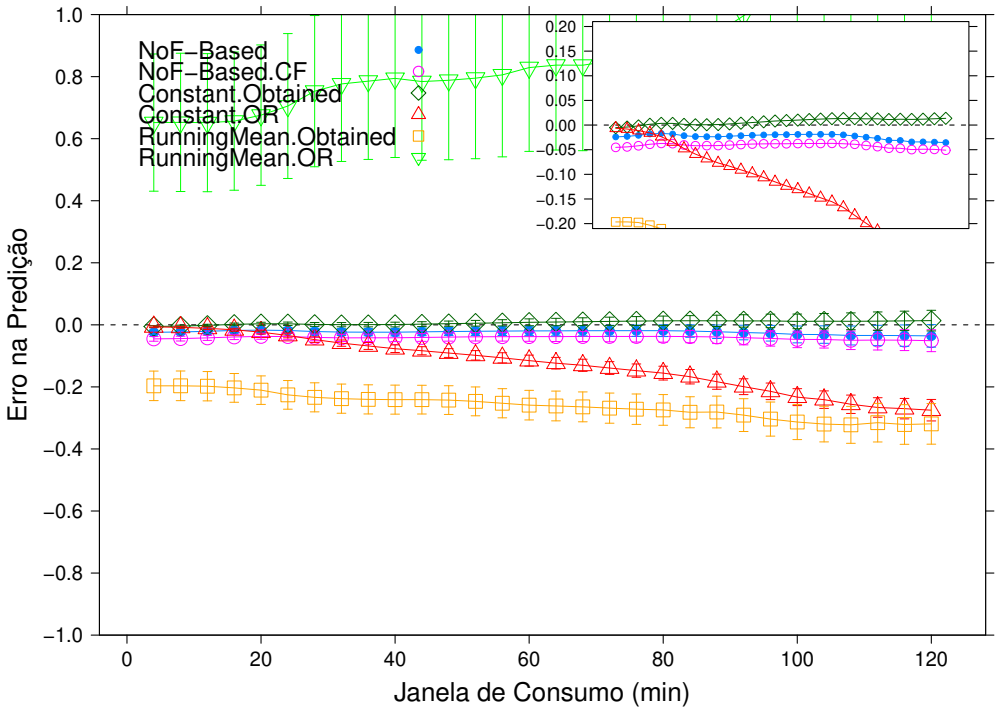
Apêndice B

Gráficos dos Erros na Predição

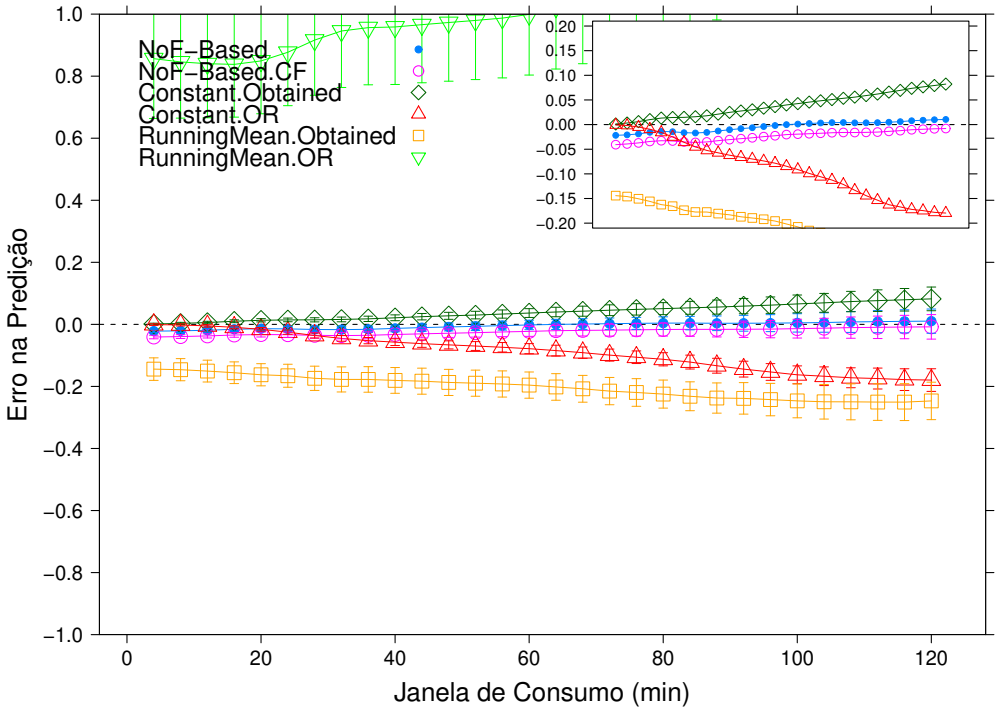


(a) $n = 30$, $u = 5$, $r = 50$

Figura B.1: Erros usando carga sintética para $r = 50$, $n = 30$ e $u = 5$

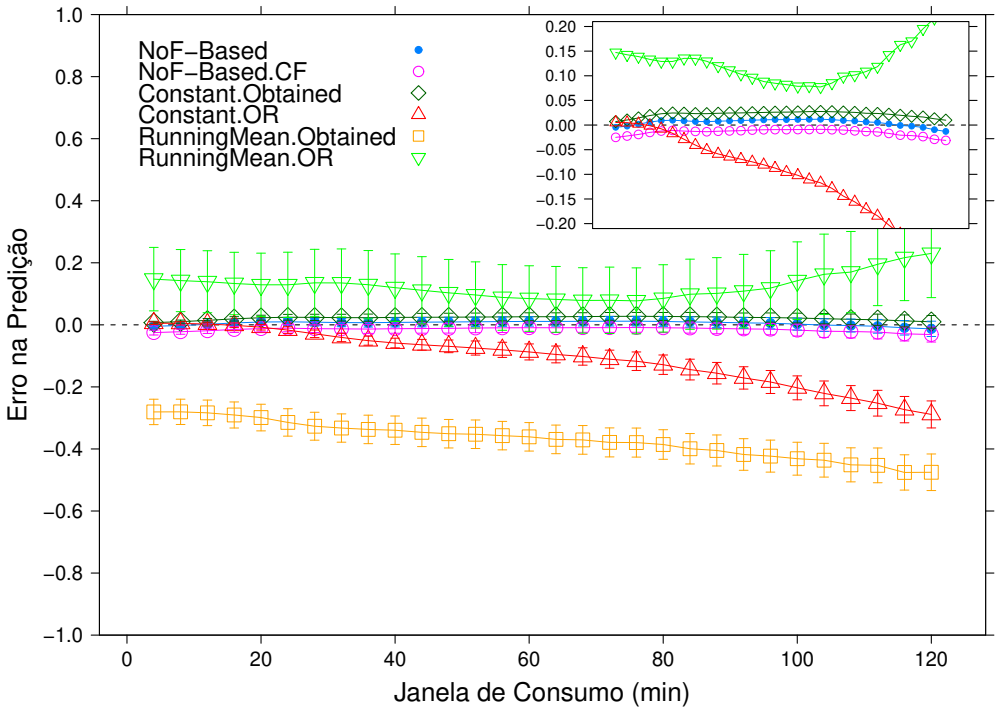


(b) $n = 30$, $u = 10$, $r = 50$

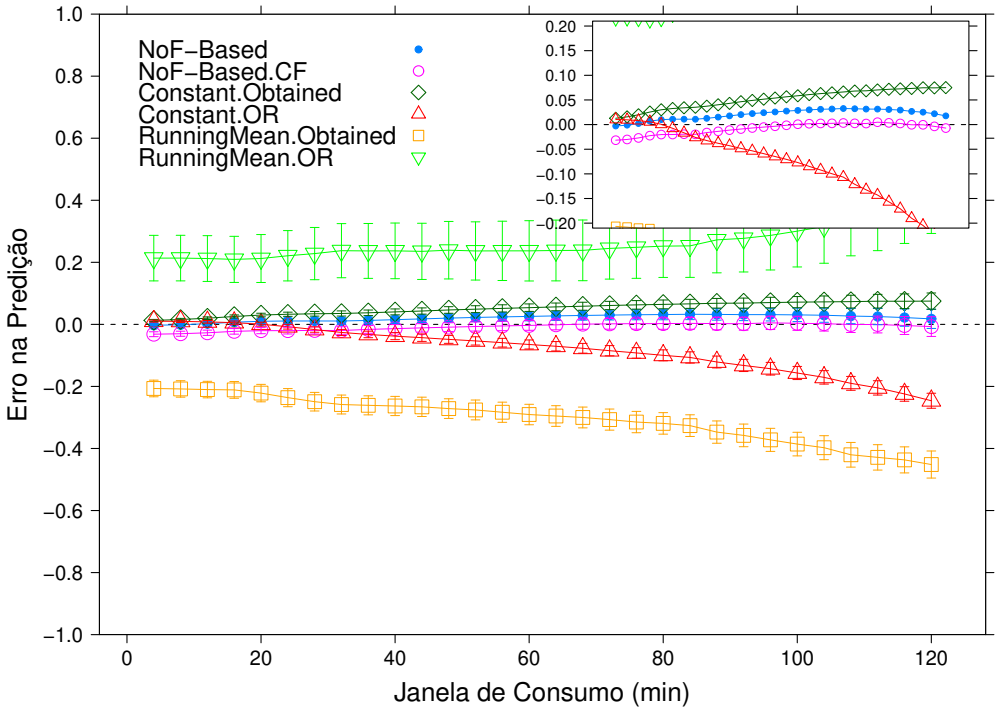


(c) $n = 30$, $u = 20$, $r = 50$

Figura B.1: Erros usando carga sintética para $r = 50$, $n = 30$ e $u = \{10, 20\}$

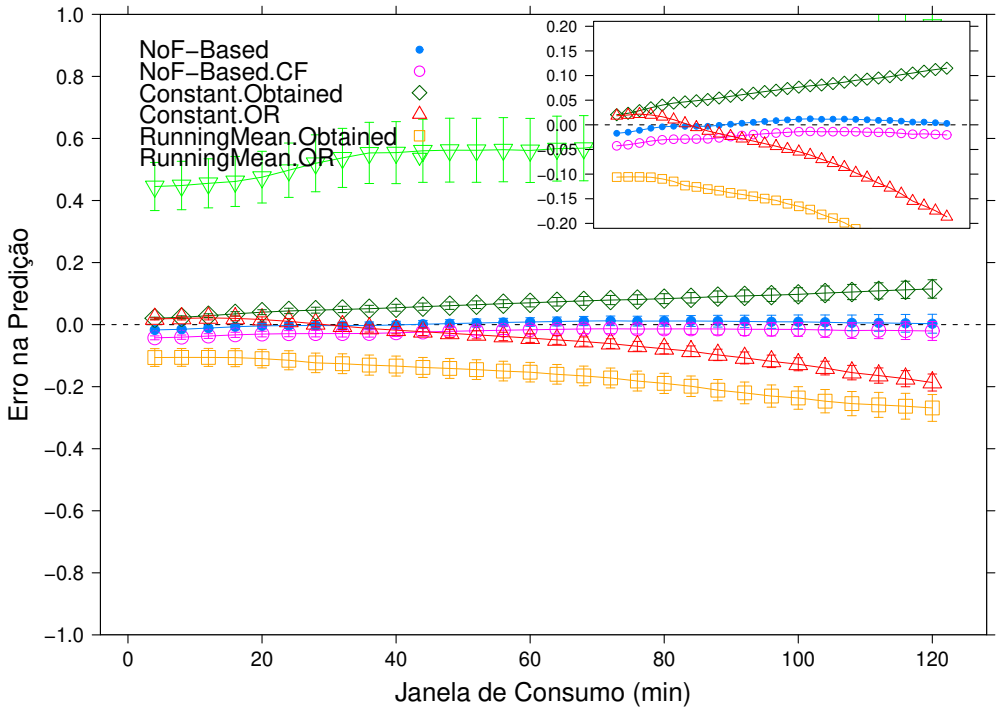


(a) $n = 60$, $u = 5$, $r = 50$



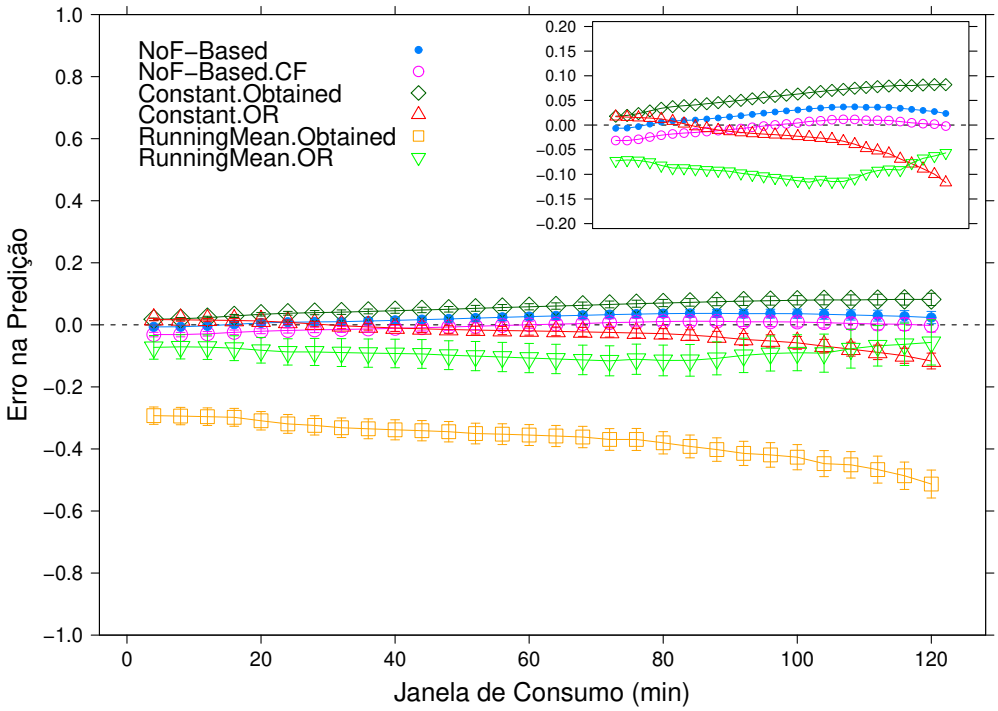
(b) $n = 60$, $u = 10$, $r = 50$

Figura B.2: Erros usando carga sintética para $r = 50$, $n = 60$ e $u = \{5, 10\}$

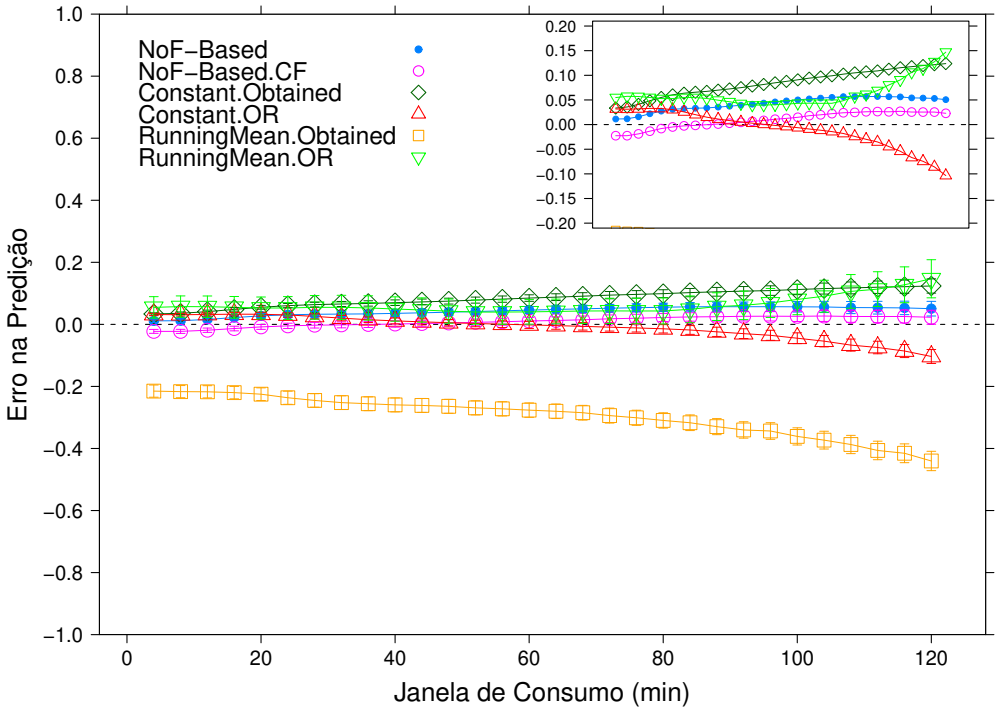


(c) $n = 60$, $u = 20$, $r = 50$

Figura B.2: Erros usando carga sintética para $r = 50$, $n = 60$ e $u = 20$

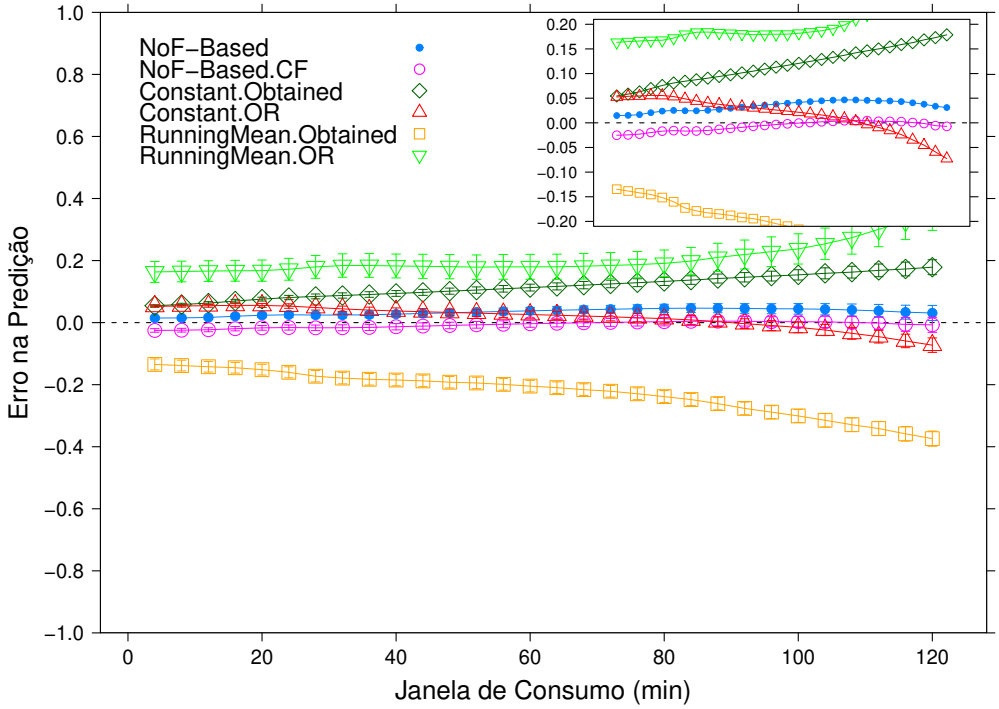


(a) $n = 120$, $u = 5$, $r = 50$



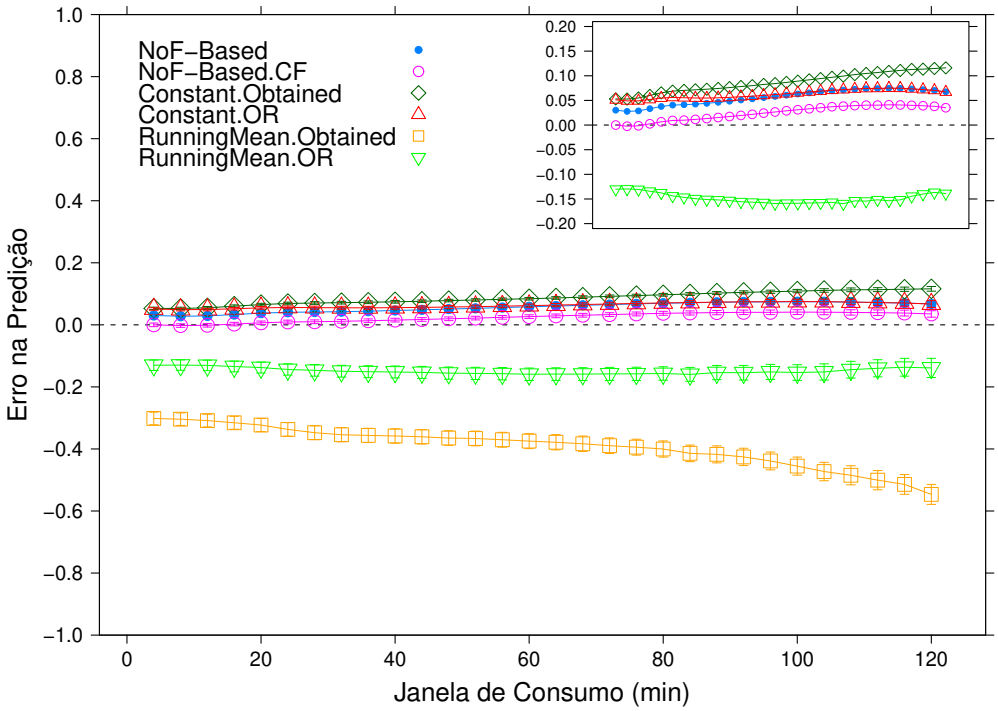
(b) $n = 120$, $u = 10$, $r = 50$

Figura B.3: Erros usando carga sintética para $r = 50$, $n = 120$ e $u = \{5, 10\}$

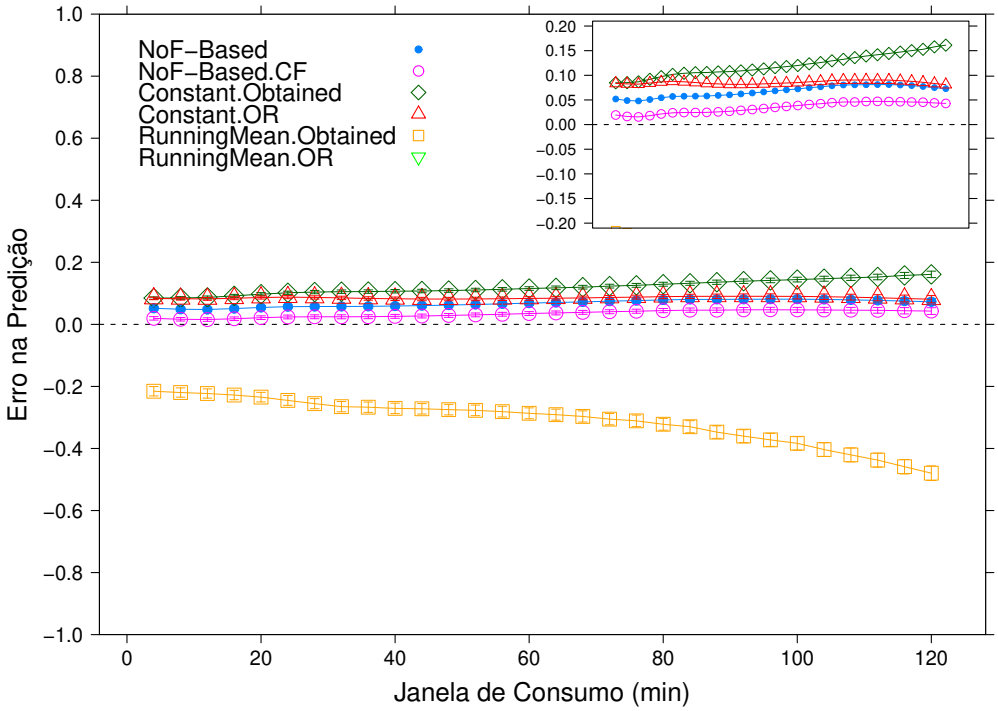


(c) $n = 120$, $u = 20$, $r = 50$

Figura B.3: Erros usando carga sintética para $r = 50$, $n = 120$ e $u = 20$

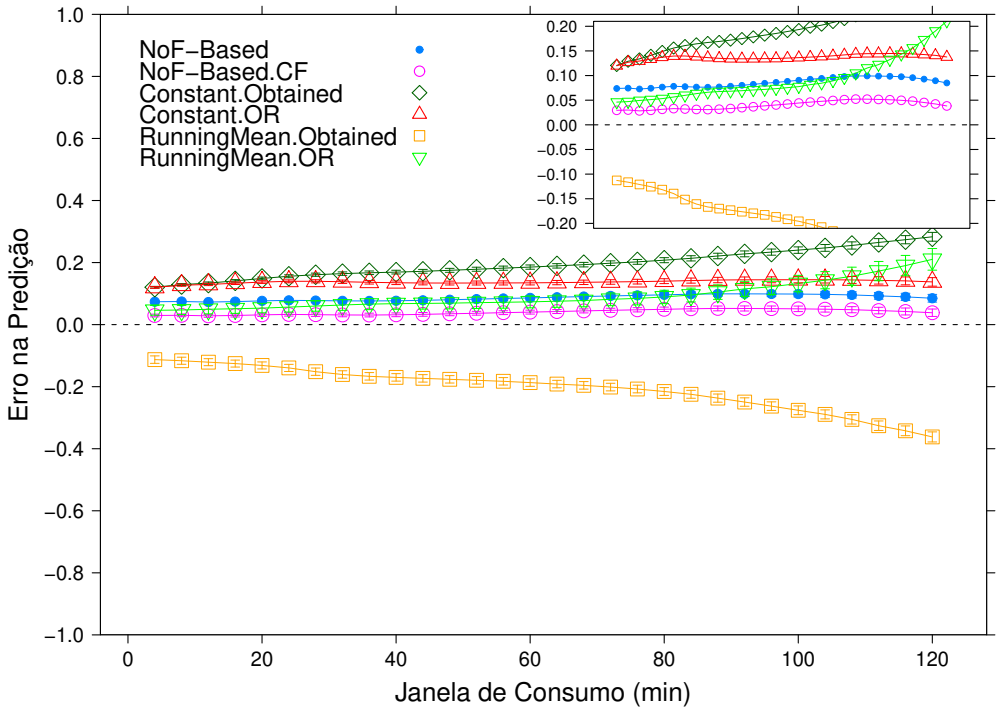


(a) $n = 240$, $u = 5$, $r = 50$



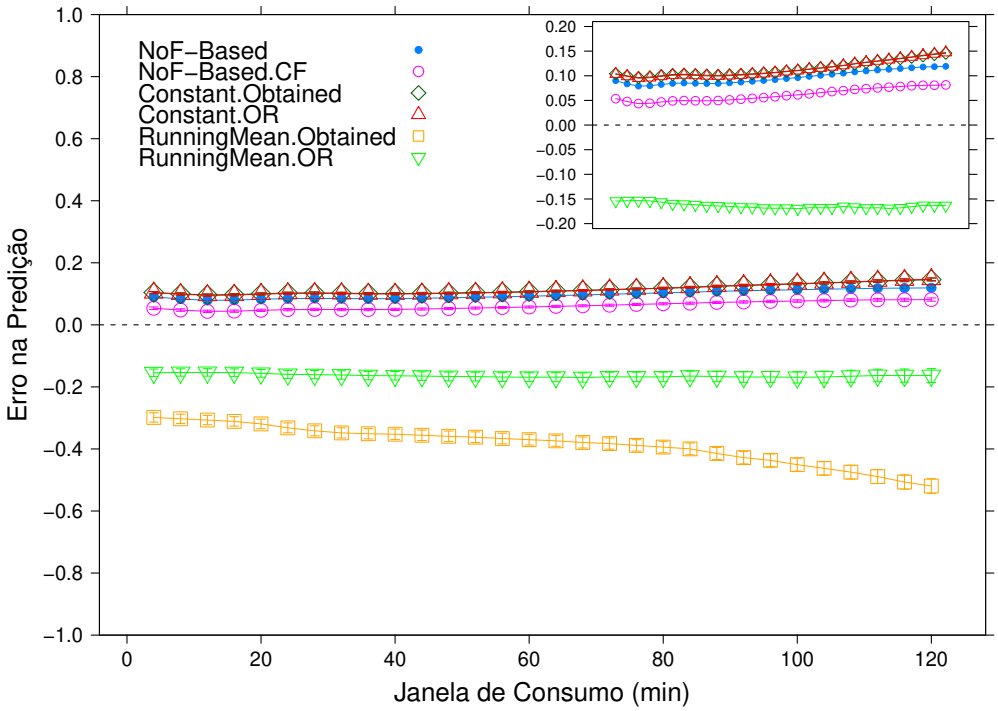
(b) $n = 240$, $u = 10$, $r = 50$

Figura B.4: Erros usando carga sintética para $r = 50$, $n = 240$ e $u = \{5, 10\}$

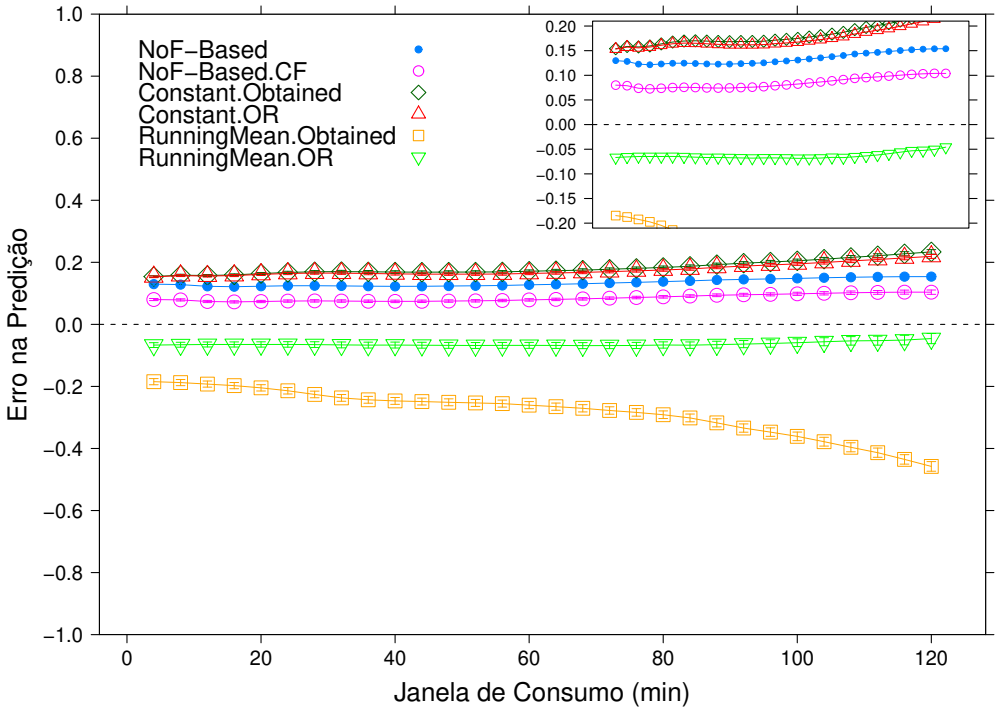


(c) $n = 240$, $u = 20$, $r = 50$

Figura B.4: Erros usando carga sintética para $r = 50$, $n = 240$ e $u = 20$

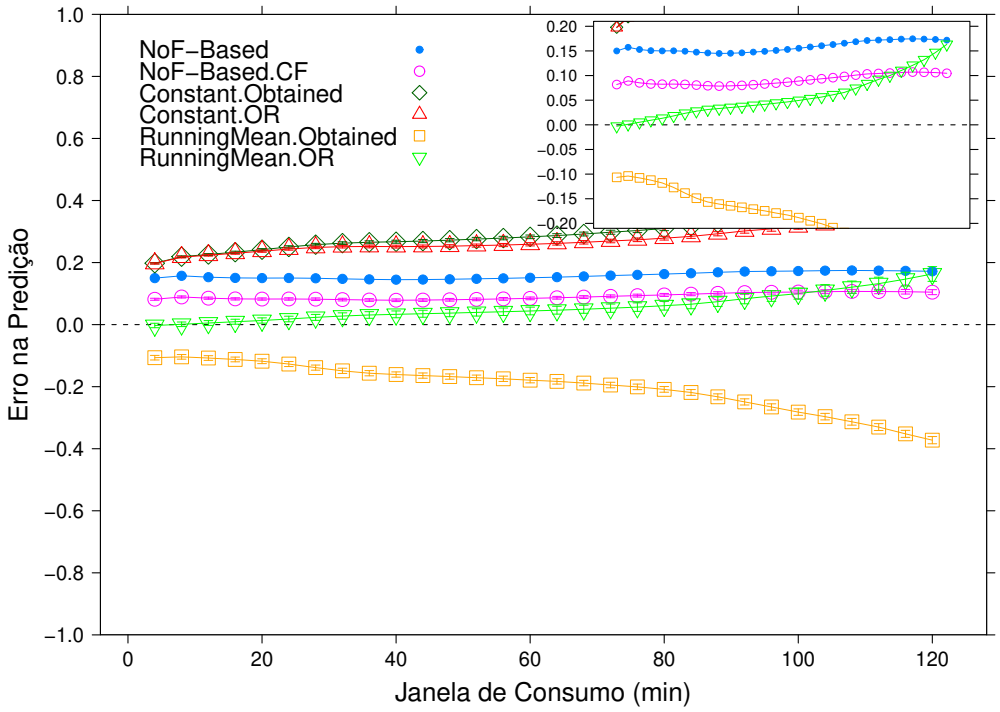


(a) $n = 480$, $u = 5$, $r = 50$



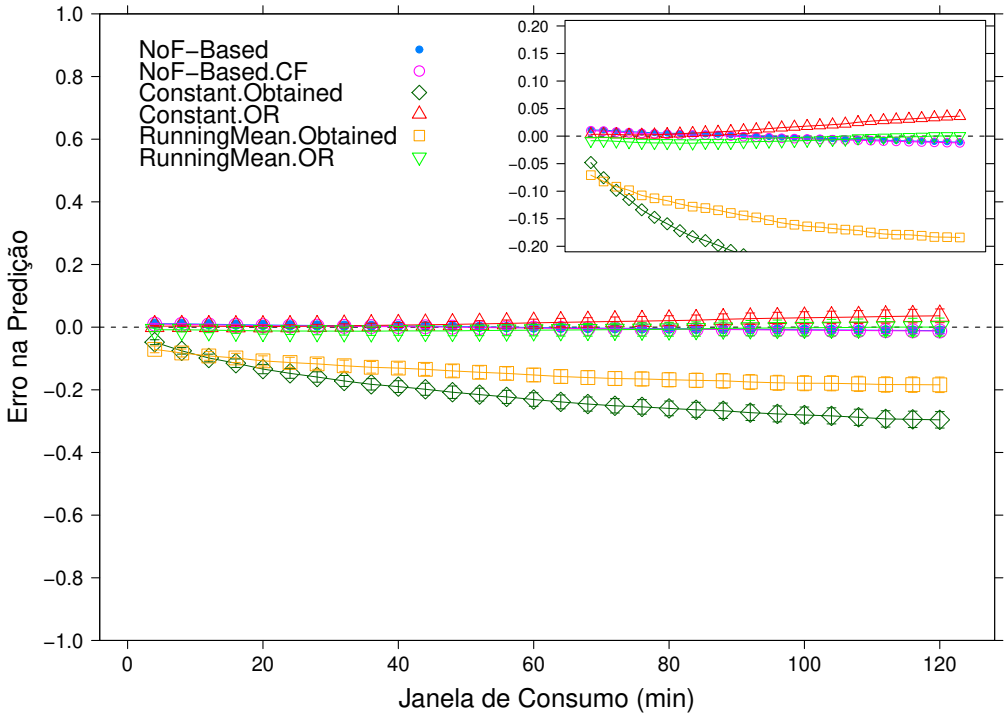
(b) $n = 480$, $u = 10$, $r = 50$

Figura B.5: Erros usando carga sintética para $r = 50$, $n = 480$ e $u = \{5, 10\}$

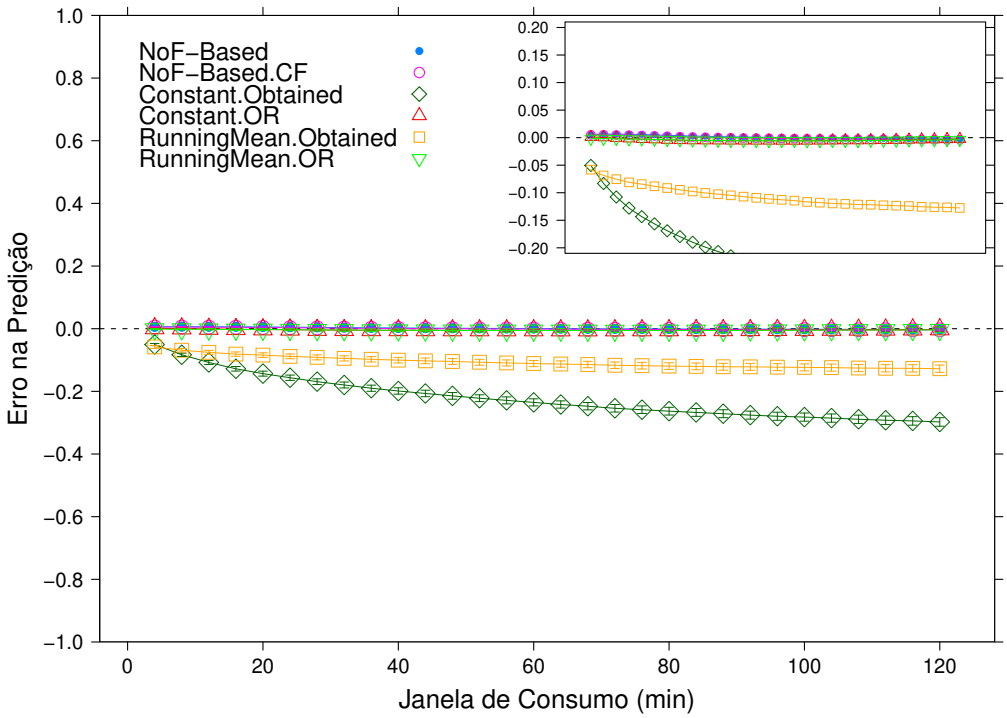


(c) $n = 480$, $u = 20$, $r = 50$

Figura B.5: Erros usando carga sintética para $r = 50$, $n = 480$ e $u = 20$

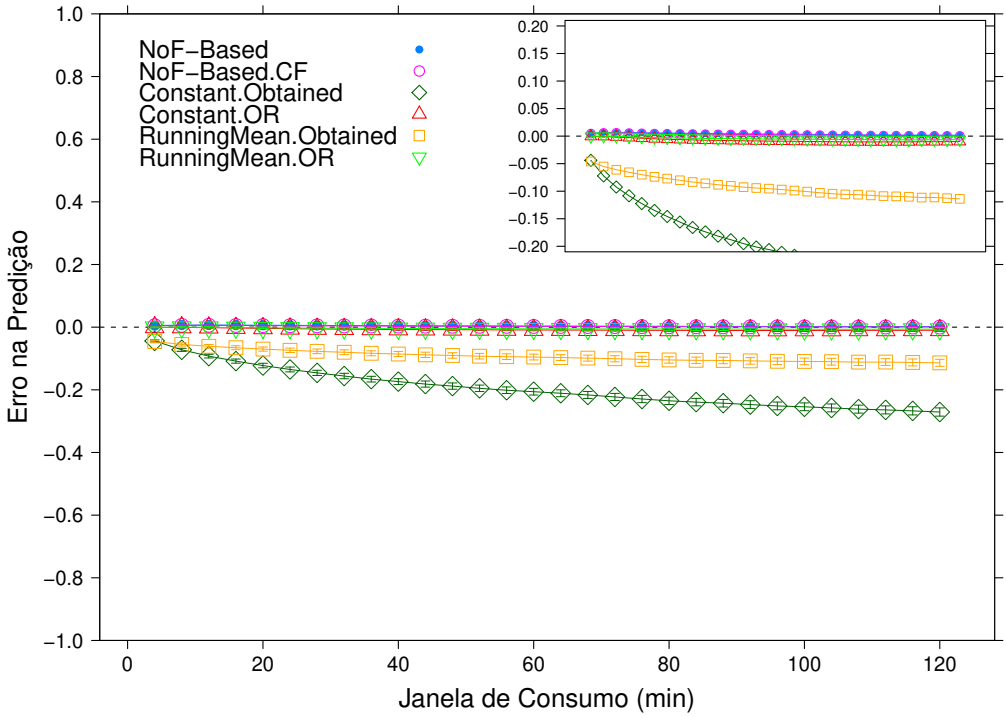


(a) $r = 20$

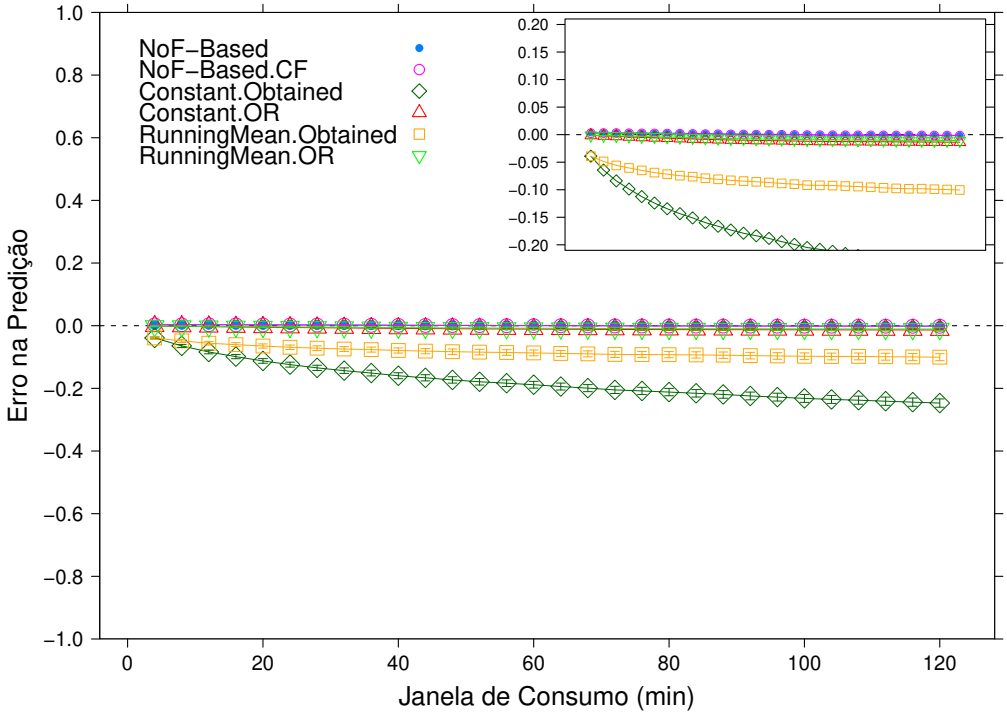


(b) $r = 40$

Figura B.6: Erros usando rastros variando $r = \{20, 40\}$



(c) $r = 60$



(d) $r = 80$

Figura B.6: Erros usando rastros variando $r = \{60, 80\}$

Apêndice C

Estatísticas dos Erros de Predição

r	Estratégia	1ºQrt	Mediana	Média	3ºQrt	Max	DP	IQR
10	NoF-Based	10.49	32.15	40.84	58.36	551.10	45.04	47.87
10	NoF-Based.CF	10.14	31.17	41.01	58.36	580.00	46.91	48.21
10	Constant.Obt	10.28	51.50	110.60	161.20	595.40	135.97	150.90
10	Constant.OR	11.60	35.92	83.09	98.38	1536.00	147.06	86.78
10	RunningMean.Obt	28.62	58.46	67.84	89.23	1425.00	65.61	60.61
10	RunningMean.OR	71.21	154.00	378.00	471.10	5575.00	531.88	399.88
20	NoF-Based	2.87	8.36	21.31	26.38	400.30	36.29	23.51
20	NoF-Based.CF	3.14	8.31	21.36	26.84	389.00	36.21	23.70
20	Constant.Obt	2.52	7.54	24.73	25.50	513.30	46.73	22.99
20	Constant.OR	5.07	14.88	23.71	32.97	448.50	33.29	27.90
20	RunningMean.Obt	14.12	41.77	48.26	68.86	652.20	47.11	54.74
20	RunningMean.OR	21.19	63.99	155.20	176.80	2810.00	261.28	155.60
40	NoF-Based	2.26	5.77	12.68	12.66	272.40	21.65	10.41
40	NoF-Based.CF	2.05	5.13	12.45	12.62	280.90	21.87	10.57
40	Constant.Obt	2.17	5.72	12.69	11.39	408.90	26.11	9.22
40	Constant.OR	3.97	8.54	15.80	23.79	285.00	19.36	19.82
40	RunningMean.Obt	6.77	33.30	39.23	64.15	344.90	34.99	57.38
40	RunningMean.OR	7.01	20.15	53.73	66.16	1428.00	94.10	59.15
100	NoF-Based	2.75	5.53	7.51	8.40	125.90	8.91	5.65
100	NoF-Based.CF	2.00	4.39	6.74	7.63	124.00	8.85	5.63
100	Constant.Obt	2.77	5.60	7.49	8.58	125.90	8.69	5.82
100	Constant.OR	2.70	5.73	7.51	9.04	125.90	8.11	6.34
100	RunningMean.Obt	4.92	25.80	35.28	63.82	192.80	32.64	58.90
100	RunningMean.OR	1.90	4.45	14.82	10.76	201.00	26.17	8.86

Tabela C.1: Estatísticas dos erros (em %) usando carga sintética variando r

u	n	Estratégia	1ºQrt	Mediana	Média	3ºQrt	Max	DP	IQR
5	30	NoF-Based	0.72	0.96	6.69	2.78	181.10	19.71	2.06
5	30	NoF-Based.CF	1.17	2.96	7.96	5.53	159.60	18.29	4.36
5	30	Constant.Obt	0.00	0.70	5.82	2.88	183.20	19.22	2.88
5	30	Constant.OR	2.74	16.29	18.98	32.66	139.90	17.87	29.92
5	30	RunningMean.Obt	8.59	35.66	42.48	78.36	96.70	34.72	69.77
5	30	RunningMean.OR	27.40	79.01	112.10	116.50	1693.00	160.43	89.07
5	60	NoF-Based	0.27	1.55	5.33	4.53	167.40	12.22	4.26
5	60	NoF-Based.CF	0.97	2.39	6.22	5.37	152.80	11.96	4.39
5	60	Constant.Obt	0.21	1.48	4.23	4.22	167.90	10.59	4.01
5	60	Constant.OR	3.46	12.41	18.53	31.98	262.40	20.87	28.52
5	60	RunningMean.Obt	10.37	38.26	43.35	72.38	278.30	34.64	62.01
5	60	RunningMean.OR	10.67	33.68	60.30	95.84	944.80	74.92	85.17
5	120	NoF-Based	1.38	3.92	7.22	7.18	128.10	11.72	5.80
5	120	NoF-Based.CF	1.68	3.91	7.96	7.42	133.40	12.50	5.75
5	120	Constant.Obt	1.48	3.90	6.48	6.93	221.20	13.41	5.45
5	120	Constant.OR	2.69	6.26	11.06	15.84	192.00	13.24	13.15
5	120	RunningMean.Obt	5.44	34.58	41.76	78.65	143.00	36.28	73.22
5	120	RunningMean.OR	3.78	13.08	34.22	47.65	603.70	48.98	43.87
5	240	NoF-Based	3.67	6.83	8.26	9.99	74.01	7.37	6.32
5	240	NoF-Based.CF	2.68	5.72	7.88	9.80	74.01	8.21	7.11
5	240	Constant.Obt	3.84	7.10	8.80	11.04	86.27	8.03	7.20
5	240	Constant.OR	3.56	6.97	8.62	10.93	82.40	7.77	7.38
5	240	RunningMean.Obt	6.38	39.22	43.74	80.98	118.30	36.74	74.60
5	240	RunningMean.OR	2.15	5.28	20.32	15.72	240.50	32.22	13.56
5	480	NoF-Based	5.50	8.87	10.53	14.37	77.65	7.44	8.87
5	480	NoF-Based.CF	3.10	6.76	8.55	12.08	78.47	7.27	8.98
5	480	Constant.Obt	5.84	9.46	11.73	16.02	73.61	8.70	10.18
5	480	Constant.OR	5.75	9.39	11.64	15.97	73.61	8.64	10.22
5	480	RunningMean.Obt	8.09	35.85	43.78	80.40	99.71	35.99	72.31
5	480	RunningMean.OR	2.80	5.86	20.21	13.09	99.71	32.33	10.30
10	30	NoF-Based	1.67	2.04	13.41	4.29	248.60	33.56	2.62
10	30	NoF-Based.CF	2.03	3.97	14.42	8.33	239.80	31.95	6.31
10	30	Constant.Obt	0.00	0.74	11.82	5.01	256.10	34.76	5.01
10	30	Constant.OR	3.73	20.96	24.66	36.75	207.70	27.11	33.02
10	30	RunningMean.Obt	13.39	27.58	38.45	57.32	596.90	36.26	43.93
10	30	RunningMean.OR	29.95	94.04	178.90	209.10	2318.00	268.80	179.12
10	60	NoF-Based	0.86	3.23	10.39	8.22	162.30	19.57	7.36
10	60	NoF-Based.CF	1.60	4.06	11.79	11.08	159.40	19.99	9.49
10	60	Constant.Obt	0.68	2.78	8.50	6.96	161.50	18.28	6.27
10	60	Constant.OR	3.31	10.20	16.32	28.08	131.50	15.71	24.78
10	60	RunningMean.Obt	6.71	34.57	39.06	62.09	367.30	33.17	55.38
10	60	RunningMean.OR	11.18	33.43	68.61	95.37	1239.00	98.15	84.19
10	120	NoF-Based	2.45	5.68	11.37	10.76	288.30	20.27	8.31
10	120	NoF-Based.CF	2.38	5.49	11.87	12.12	264.40	19.74	9.74
10	120	Constant.Obt	2.31	5.45	10.93	9.95	310.10	21.38	7.64
10	120	Constant.OR	3.55	7.57	13.47	19.43	235.20	16.42	15.88
10	120	RunningMean.Obt	6.20	31.72	37.82	63.96	276.60	33.31	57.76

10	120	RunningMean.OR	5.12	14.52	37.79	41.02	941.50	62.61	35.91
10	240	NoF-Based	4.91	8.49	11.75	15.11	246.90	11.49	10.20
10	240	NoF-Based.CF	3.22	6.88	10.52	13.63	239.30	11.71	10.41
10	240	Constant.Obt	5.43	9.14	13.03	16.95	251.50	13.67	11.51
10	240	Constant.OR	4.83	8.65	11.81	15.45	242.00	11.89	10.62
10	240	RunningMean.Obt	7.44	27.41	37.94	65.52	279.60	33.12	58.08
10	240	RunningMean.OR	3.14	7.41	19.37	16.68	537.00	30.63	13.55
10	480	NoF-Based	8.06	13.75	16.44	22.08	342.70	13.05	14.03
10	480	NoF-Based.CF	4.17	9.59	12.69	17.97	251.90	12.36	13.80
10	480	Constant.Obt	8.77	15.28	19.32	26.57	342.70	15.66	17.80
10	480	Constant.OR	8.51	14.91	18.48	25.66	342.70	14.32	17.15
10	480	RunningMean.Obt	9.54	27.15	37.87	65.19	471.90	31.95	55.65
10	480	RunningMean.OR	3.13	7.07	16.55	14.27	1507.00	32.04	11.14
20	30	NoF-Based	1.36	3.34	17.20	20.77	333.80	34.09	19.40
20	30	NoF-Based.CF	2.39	6.13	18.35	21.02	334.90	33.55	18.62
20	30	Constant.Obt	0.32	3.43	18.29	21.36	340.70	36.33	21.03
20	30	Constant.OR	4.40	17.36	24.59	33.63	305.20	33.09	29.24
20	30	RunningMean.Obt	16.80	34.31	41.43	54.71	572.70	39.72	37.91
20	30	RunningMean.OR	28.48	93.31	173.50	230.20	4585.00	219.64	201.71
20	60	NoF-Based	1.71	6.41	19.65	24.50	337.00	32.56	22.79
20	60	NoF-Based.CF	2.77	7.37	20.39	26.25	347.30	31.85	23.48
20	60	Constant.Obt	1.44	5.25	19.70	20.21	360.80	36.87	18.77
20	60	Constant.OR	4.15	13.95	21.95	32.50	354.10	27.04	28.35
20	60	RunningMean.Obt	9.63	33.45	39.48	55.33	689.70	43.68	45.70
20	60	RunningMean.OR	17.30	45.09	115.80	133.50	2144.00	190.48	116.22
20	120	NoF-Based	3.84	8.47	18.19	21.54	708.00	33.34	17.70
20	120	NoF-Based.CF	3.28	8.13	18.24	22.50	701.50	32.99	19.22
20	120	Constant.Obt	3.58	8.02	19.34	19.64	708.10	38.67	16.06
20	120	Constant.OR	4.52	10.09	18.40	23.93	561.10	28.60	19.41
20	120	RunningMean.Obt	7.65	30.09	37.24	59.87	769.80	35.42	52.22
20	120	RunningMean.OR	7.06	18.83	53.85	56.88	1430.00	97.72	49.82
20	240	NoF-Based	6.75	12.88	19.10	24.55	357.80	21.66	17.80
20	240	NoF-Based.CF	4.18	10.06	16.78	22.39	347.10	20.96	18.21
20	240	Constant.Obt	7.57	14.91	24.84	29.71	423.30	33.39	22.14
20	240	Constant.OR	6.86	13.20	20.01	25.11	491.90	24.38	18.25
20	240	RunningMean.Obt	9.35	28.06	36.03	56.53	1421.00	32.55	47.18
20	240	RunningMean.OR	4.74	11.38	35.19	28.04	2461.00	75.01	23.30
20	480	NoF-Based	10.99	19.60	24.16	31.70	1060.00	22.10	20.71
20	480	NoF-Based.CF	6.16	14.39	19.38	26.26	958.20	21.15	20.10
20	480	Constant.Obt	13.48	25.46	36.67	44.59	636.60	44.90	31.12
20	480	Constant.OR	12.76	23.88	32.03	41.10	647.30	35.05	28.34
20	480	RunningMean.Obt	11.76	29.35	36.97	58.09	678.90	30.25	46.33
20	480	RunningMean.OR	4.52	10.28	30.28	22.13	1792.00	73.41	17.61

Tabela C.2: Estatísticas dos erros (em %) usando carga sintética variando u e n

r	**Estratégia**	**1ºQrt**	**Mediana**	**Média**	**3ºQrt**	**Max**	**DP**	**IQR**
20	NoF-Based	1.49	3.62	9.26	13.90	59.84	11.45	12.41
20	NoF-Based.CF	1.50	3.73	9.17	13.55	59.84	11.33	12.04
20	Constant.Obt	25.12	50.76	45.56	67.08	87.91	24.08	41.96
20	Constant.OR	2.07	5.95	39.97	29.24	1539.00	124.80	27.17
20	RunningMean.Obt	3.02	16.86	24.90	39.27	92.96	24.92	36.25
20	RunningMean.OR	2.87	9.47	37.11	46.02	1381.00	100.40	43.15
40	NoF-Based	0.67	1.51	3.94	3.34	71.61	7.34	2.67
40	NoF-Based.CF	0.66	1.53	3.91	3.33	71.89	7.29	2.66
40	Constant.Obt	24.89	45.74	43.71	62.93	90.90	23.81	38.04
40	Constant.OR	0.88	2.34	12.63	6.73	1047.00	54.67	5.85
40	RunningMean.Obt	1.51	7.31	17.41	28.11	89.96	20.89	26.60
40	RunningMean.OR	0.99	2.25	14.00	6.57	914.70	50.38	5.58
60	NoF-Based	0.54	1.18	3.38	2.43	54.47	7.08	1.90
60	NoF-Based.CF	0.53	1.18	3.34	2.46	53.62	6.97	1.92
60	Constant.Obt	22.73	41.90	42.14	61.45	91.10	24.09	38.72
60	Constant.OR	0.74	1.92	8.12	5.65	413.60	28.22	4.92
60	RunningMean.Obt	1.09	5.69	16.89	28.62	87.17	20.74	27.54
60	RunningMean.OR	0.75	1.87	9.22	4.83	380.80	27.73	4.08
80	NoF-Based	0.41	0.96	2.44	1.88	54.68	5.72	1.47
80	NoF-Based.CF	0.41	0.99	2.43	1.90	54.73	5.68	1.49
80	Constant.Obt	20.04	37.76	39.74	58.76	92.59	24.19	38.72
80	Constant.OR	0.66	1.64	4.69	4.26	135.90	10.11	3.60
80	RunningMean.Obt	0.98	5.60	16.12	26.85	83.49	20.22	25.87
80	RunningMean.OR	0.61	1.45	5.95	3.21	134.70	14.02	2.60

Tabela C.3: Estatísticas dos erros (em %) usando rastros variando r