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

Dissertação de Mestrado

Um Arcabouço Não Intrusivo para Provisionamento Automático de Recursos em Ambientes de IaaS

Fábio Jorge Almeida Morais

Campina Grande, Paraíba, Brasil
Fevereiro de 2013

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

Um Arcabouço Não Intrusivo para Provisionamento Automático de Recursos em Ambientes de IaaS

Fábio Jorge Almeida Morais

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: Metodologia e Técnicas da Computação

Francisco Vilar Brasileiro (Orientador)
Raquel Vigolvino Lopes (Orientadora)

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

M827a Morais, Fábio Jorge Almeida.
Um arcabouço não instrutivo para provisionamento automático de recursos em ambientes de IaaS. -- 2013.
112 f. : il. color.

Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática.

"Orientação: Prof. Dr. Francisco Vilar Brasileiro, Profa. Dra. Raquel Vigolvino Lopes".
Referências.

1. Computação na Nuvem. 2. Infraestrutura como Serviço.
3. Gerência de Capacidade. 4. Simulação. 5. Predição. I. Brasileiro, Francisco Vilar. II. Lopes, Raquel Vigolvino. III. Título.

CDU 004.7(043)

M ARCABOUÇO NÃO INTRUSIVO PARA PROVISIONAMENTO AUTOMÁTICO DE RECURSOS EM AMBIENTES DE IAAS"

FABIO JORGE ALMEIDA MORAIS

DISSERTAÇÃO APROVADA EM 26/02/2013

FRANCISCO VILAR BRASILEIRO, Ph.D, UFCG
Orientador(a)

RAQUEL VIGOLVINO LOPES, D.Sc, UFCG
Orientador(a)

ANDREY ELÍSIO MONTEIRO BRITO, Dr., UFCG
Examinador(a)

MARCO AURÉLIO STELMAR NETTO, Dr., IBM
Examinador(a)

CAMPINA GRANDE - PB

Resumo

O paradigma de *Computação na Nuvem* carrega consigo o conceito de elasticidade, que consiste na provisão de recursos computacionais sob demanda. Esse conceito é uma das principais propriedades usadas na redução de custos derivados da execução de serviços em ambientes de infraestrutura como serviço (IaaS). No entanto, essa propriedade só pode ser inteiramente explorada se os clientes dos serviços de IaaS forem capazes de estimar futuras demandas de suas aplicações no curto prazo, de forma que apenas a infraestrutura necessária para manter as aplicações é requisitada a cada instante de tempo. Deste modo, os acordos de nível de serviço (SLAs) firmados entre o cliente do serviço de IaaS e os usuários de suas aplicações são sempre honrados e o super provisionamento é evitado.

O provisionamento automático consiste no processo de modificar automaticamente a quantidade de recursos disponíveis para manter e executar uma aplicação em um ambiente de IaaS, dependendo da demanda da aplicação. O estado da prática apresenta soluções de provisionamento automático que utilizam abordagens reativas, que em geral são insuficientes para minimizar os custos de violações de SLA, embora possam reduzir os custos do super provisionamento. Para reduzir os custos devido a violações de SLA são necessárias abordagens proativas.

Este trabalho propõe um arcabouço para provisionamento automático de recursos não intrusivo. O arcabouço realiza o provisionamento a partir das abordagens reativa e proativa, baseadas no uso de um conjunto configurável de preditores de demandas dos serviços, além de usar um mecanismo de seleção que decide, periodicamente, o melhor preditor a ser usado. Também é proposta uma nova maneira de corrigir predições subestimadas, reduzindo por consequência o número de quebras de SLA.

O arcabouço proposto foi avaliado através de simulações baseadas em rastros de utilização de aplicações em produção de clientes da HP. Os resultados mostram que é possível obter uma economia de até 37% enquanto a probabilidade de quebra de SLA é mantida em média em $0,008\%$ e limitada superiormente a $0,036\%$. Além disso, a flexibilidade do arcabouço permite que, através da utilização de diferentes configurações, seja possível alcançar economias adicionais apenas com um pequeno aumento no número de violações de SLA.

Abstract

The paradigm of cloud computing has brought the concept of elasticity, which is the on demand provision of computational resources. This property is the key to reduce the costs derived from the execution of services in cloud systems that employ an infrastructure-as-a-service (IaaS) deployment model. However, this property can only be fully exploited if the users of IaaS services are able to estimate the short-term future demands of their own applications, so that only the necessary infrastructure to maintain the application is requested at each instant of time. Thus, the service level agreements (SLA), signed between the client of the IaaS service and the users of their applications, are always honored and over provisioning is avoided.

The auto-scaling is the process of dynamically modifying the amount of resources available to maintain and run an application on an IaaS system, depending on the load application. The state-of-practice provides solutions for auto-scaling using reactive approaches, which are not sufficient to minimize the costs of SLA violations, although they may reduce the costs of over provisioning. To reduce costs due to SLA violations proactive approaches are necessary.

This work proposes a flexible non-intrusive framework for auto-scaling services. The framework follows a hybrid, reactive and proactive, approache based on the use of a configurable set of predictors for the future demand of services and uses a selection mechanism that decides, over time, the best predictor to be used. Also a new way of correcting underestimations is proposed, which reduces the number of SLA violations.

The proposed framework had its performance evaluated through simulations using production utilization traces of HP customers. The results show that costs savings of as much as 37% can be achieved, while the probability of an SLA violation can be kept, on average, as small as 0.008%, and no larger than 0.036%. Moreover, the flexibility of the framework allows different configurations to be used, for which additional cost savings can be achieved with only a small increase on the number of SLA violations.

Não tenhamos pressa, mas não percamos tempo.

José Saramago

Agradecimentos

Ao CNPq por financiar meus estudos de pós-graduação. À Hewlett Packard (HP) cuja parceria viabilizou a realização deste trabalho. Aos meus orientadores, Fubica e Raquel Lopes, que por vezes me indicaram o bom caminho. Aos que fazem ou fizeram parte do Laboratório de Sistemas Distribuídos, os quais pude contar ao longo desses anos. Aos amigos com quem dividi ideias, questionamentos, momentos, cafés e cervejas. Aos meus pais, responsáveis pelo incentivo e fomento à minha formação. À minha esposa e filha, fontes infinitas de força e amor. Aos que duvidaram. Aos que nos deixaram. E aos que direta ou indiretamente fizeram parte desta etapa.

Sumário

Lista de Siglas e Abreviaturas

AC	*Auto-Correlation*
ACF	*Auto-Correlation Function*
API	*Application Programming Interfaces*
AR	*Auto-Regressive*
ARIMA	*Auto-Regressive Integrated Moving Average*
ARMA	*Auto-Regressive Moving Average*
EN	*Ensemble*
FDA	Função de Distribuição Acumulada
FFT	*Fast Fourier Transform*
HP	*Hewlett-Packard*
IaaS	*Infrastructure as a Service*
LR	*Linear Regression*
LW	*Last Window*
MA	*Moving Average*
PaaS	*Platform as a Service*
QoS	*Quality of Service*
SaaS	*Software as a Service*
SLA	*Service Level Agreement*
SLO	*Service Level Objective*
TDD	*Test-Driven Development*
TI	Tecnologia da Informação
VM	*Virtual Machine*
WMA	*Weighted Moving Average*

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Contextualização

A computação como serviço utilitário, conhecida por *Computação na Nuvem*, tornou-se uma realidade nos últimos anos [4], proporcionando flexibilidade e escalabilidade na provisão e na utilização de serviços de infraestruturas computacionais. Estes serviços, em geral, são apresentados na forma de máquinas virtuais (VM, do inglês *Virtual Machine*) implantadas em centros de processamento de dados dos provedores de *Computação na Nuvem*; são as chamadas infraestruturas como serviço (IaaS, do inglês *Infrastructure as a Service*) [45]. Na prática, este tipo de serviço consiste na obtenção de recursos computacionais, como poder de processamento, memória e disco, de um provedor e na utilização destes recursos para implantação e execução de aplicações de interesse [44].

Existem duas formas fundamentais de utilização e provisão de serviços de IaaS, a privada e a pública. Na privada, o paradigma de *Computação na Nuvem* é aplicado ao contexto de uma mesma empresa. Nesse caso, o principal objetivo não é vender recursos computacionais, mas prover aos usuários locais da empresa uma infraestrutura flexível para executar cargas de trabalho de serviços dentro de seus domínios administrativos [4]. Ou seja, tirar proveito dos conceitos de *Computação na Nuvem* para realizar a gerência dos recursos finitos da organização para seus clientes internos [43].

Por outro lado, na *Nuvem* pública os provedores de computação oferecem infraestrutura computacional sob demanda, normalmente empregando uma tarifação do serviço baseada no modelo “pague conforme utilização” (do inglês *pay-as-you-go*), no qual clientes idealmente

pagam apenas pelos recursos de fato utilizados [5]. Para o mercado público de *Computação na Nuvem*, o serviço de IaaS foi o segmento que presentou o crescimento mais acentuado no último ano [17]. Diversas empresas deste mercado oferecem esse tipo de serviço, como Amazon [3], GoGrid [18], Rackspace [32], dentre outras.

A adoção desse paradigma é decorrente das vantagens em termos de custo e confiabilidade na utilização de serviços de IaaS em comparação aos serviços tradicionais, que consistem na utilização de infraestruturas físicas dedicadas [20]. Uma vantagem se dá na redução dos custos computacionais e de operação, devido ao compartilhamento de recursos, à virtualização e ao baixo custo de manutenção e de investimento em infraestruturas de Tecnologia da Informação (TI) [49].

Outra vantagem, talvez a principal, oferecida pelos provedores de IaaS para os clientes é a elasticidade, que consiste na propriedade que permite a provisão sob demanda de recursos computacionais para uma aplicação [1, 26]. Através desta, uma dada aplicação pode utilizar diferentes quantidades de recursos para suprir variações de carga ao longo do tempo. A utilização dessa propriedade só pode ser considerada ótima se a política de provisionamento de recursos de uma aplicação, com demandas que variam no tempo, disponibiliza a cada instante de tempo a capacidade exata para manter o desempenho da aplicação. A consequência é o cumprimento dos objetivos de nível de serviço (SLO, do inglês *Service Level Objective*) e a satisfação dos acordos de nível de serviço (SLA, do inglês *Service Level Agreement*) estabelecidos entre os clientes do provedor de IaaS e os usuários de suas aplicações. Tal política de provisionamento gerencia automaticamente a capacidade, provendo mais recursos quando a demanda aumenta, e liberando recursos à medida que não são mais necessários.

Portanto, a adoção desse paradigma traz consigo certas complexidades ao processo de gestão de capacidade. Muitos dos serviços ou aplicações que precisam ser migrados para ambientes de *Nuvem* são críticos para as empresas. Por isso, o planejamento da capacidade apropriada para manter a qualidade de serviço (QoS, do inglês *Quality of Service*) em um nível desejado é uma atividade fundamental no processo de migração. Uma forma usual de lidar com esse problema é seguir uma estratégia conservadora que utiliza conhecimento sobre o serviço em execução para estimar a demanda máxima e superprover estaticamente a infraestrutura, aplicando uma margem de segurança, para atender variações abruptas na demanda. No contexto de uma infraestrutura privada, realizar superprovisionamento fere o

princípio de que a capacidade total necessária pode ser minimizada através da "multiplexação" de diferentes aplicações com demandas elevadas em instantes de tempo diferentes. De maneira similar, em infraestruturas públicas de IaaS, o superprovisionamento elimina a principal vantagem do modelo *pay-as-you-go*, visto que o cliente estará pagando por recursos que não foram de fato utilizados durante a execução do serviço. Além do mais, essa estratégia não garante que os níveis de qualidade de serviço serão mantidos, dado que, em geral, a carga máxima do serviço não é previamente conhecida.

Outra estratégia usada para lidar com o problema de planejamento de capacidade é a utilização de políticas de provisionamento automático de recursos a curto prazo. Estas políticas consistem naquelas em que o monitoramento e o provisionamento são realizados automaticamente pelo sistema da infraestrutura de *Computação na Nuvem*, de acordo com regras e configurações definidas pelo cliente ou especificadas no contrato de nível de serviço [16]. Uma vez que é tido como impossível conhecer com exatidão futuras demandas da aplicação, o objetivo desse tipo de política é minimizar o número de violações de SLA, que podem ocorrer quando a demanda da aplicação não consegue ser suprida pelo montante de recursos a ela reservados. Um segundo objetivo consiste na redução do custo decorrente da quantidade de recursos alocados à aplicação, através da diminuição de possíveis superdimensionamentos de capacidade.

As atuais soluções de provisionamento automático a curto prazo desenvolvidas e disponibilizadas pelo mercado utilizam uma abordagem reativa, como a API (do inglês *Application Programming Interfaces*) da Amazon [39] e as ferramentas Rightscale [33] e Scalr [37]. Esse tipo de abordagem é baseado em regras e gatilhos, isto é, quando todas as condições que especificam uma regra são satisfeitas uma ação de provisionamento é disparada automaticamente pelo sistema da infraestrutura em reação a alterações na carga da aplicação. Abordagens reativas também são utilizadas nas soluções propostas por Lim et al. [27], Meng et al. [30], Calheiros et al. [9], Fitó et al. [15], Vijayakumar et al. [48], Calcavecchia et al. [8], Marshall et al. [29] e Bonvin et al. [6]. Entretanto, esse tipo de abordagem, em geral, não consegue manter a qualidade de serviço em um nível desejável, dado que a ação de provisionamento é disparada em reação a uma mudança de carga. Ou seja, quando o provisionamento começa a ser executado uma quantidade incerta de violações de SLO podem ter ocorrido desde o disparo da ação.

Outras soluções, por outro lado, utilizam uma abordagem proativa, que baseia-se na utilização de heurísticas e técnicas de análise de dados para prever o comportamento da aplicação e baseado nessas predições decidir quando e como realizar o provisionamento de recursos [16]. Para que esse tipo de abordagem seja eficiente é necessário que o gerente de capacidade possua a habilidade de prever mudanças na demanda da aplicação. Realizar estimativas de demanda é uma tarefa não trivial, contudo, as soluções de planejamento de capacidade a curto prazo têm dado pouca importância a esse fato ou simplesmente estão abstraindo sua complexidade, por assumir a disponibilidade de um preditor sem de fato avaliar a influência da predição na performance do provisionamento automático em si. Os trabalhos de Dawoud et al. [14], Roy et al. [34], Gong et al. [19], Vasić et al. [47], Shen et al. [42] e Sharma et al. [41] fazem uso de abordagens proativas no provisionamento de recursos a curto prazo.

Além do mais, alguns trabalhos requerem a coleta de informações sobre a aplicação (ou serviço) em execução na infraestrutura (como tempo de resposta, tamanho da fila, taxa de chegada e demanda das requisições) [41, 46, 48], o que remonta questões de privacidade, dado que essas são informações críticas que nem sempre podem ser compartilhadas, mesmo com os provedores do serviço de IaaS.

Desta forma, fica evidente a necessidade de um estudo de planejamento de capacidade a curto prazo em ambientes de IaaS que contemple as limitações existentes nas soluções propostas tanto pela academia quanto pelo mercado. Um estudo que fomente e viabilize a utilização da elasticidade oferecida pelos provedores de IaaS sem comprometer a segurança e a privacidade dos dados das aplicações implantadas na infraestrutura de *Nuvem*.

1.2 Contribuições e Relevância

Este trabalho tem como objetivo principal abordar as limitações das soluções existentes no mercado e em trabalhos propostos pela academia para o provisionamento automático de recursos a curto prazo em infraestruturas de *Nuvem*. Para tal, é proposto um arcabouço para provisionamento automático em ambientes de IaaS que: (i) faz uso de um *mecanismo de provisionamento híbrido*, isto é, utiliza ao mesmo tempo uma abordagem proativa e reativa; (ii) funciona de forma *não intrusiva*, ou seja, necessita apenas do monitoramento dos recur-

sos da infraestrutura onde a aplicação está em execução; (iii) possui *configuração dinâmica e automática* em relação às atividades de predição e de monitoramento.

Em particular, o arcabouço proposto considera o uso de múltiplos preditores, um vez que não existe um único preditor que é o melhor em estimar a carga para todas as aplicações [24], dado a variedade e diversidade dos tipos de aplicação e dos padrões de carga. Além disso, o arcabouço utiliza um seletor que avalia periodicamente as opções de predição disponíveis e a partir de critérios previamente definidos decide o preditor que será utilizado no futuro próximo, possibilitando que o mecanismo de provisionamento se adapte automaticamente a alterações nos padrões de carga das aplicações em execução. Complementarmente, um novo método de correção de predição é proposto, baseado na correlação do histórico de erros de predição, que visa melhorar o desempenho do mecanismo de provisionamento, evitando erros de subprovisionamento e reduzindo a quantidade de possíveis violações de SLA.

1.3 Metodologia

A partir de um levantamento bibliográfico do estado da arte e da prática, que serviu de base para que fossem destacadas necessidades e lacunas existentes nas soluções de provisionamento automático de recursos em ambientes de *Computação na Nuvem*, foi produzida uma definição detalhada do arcabouço proposto nesse trabalho, bem como do mecanismo de provisionamento automático por ele utilizado. Em seguida, através da linguagem de programação R [11] o arcabouço foi implementado, visto que esta é um linguagem facilmente extensível por meio de bibliotecas e pacotes, além de possuir todo um ferramental base para realização de análises estatísticas e execução e avaliação de predições. Em pormenores, essa implementação consiste em um modelo analítico do ambiente de IaaS e em um simulador do mecanismo de provisionamento automático de recursos utilizado pelo arcabouço.

Após a implementação do arcabouço, o mesmo foi instanciado e configurado segundo valores característicos das soluções existentes no mercado e as simulações foram alimentadas a partir de dados de utilização de recursos de aplicações em produção de clientes da Hewlett-Packard[1] (HP). Por fim, foram realizadas avaliações das instanciações do arcabouço segundo

[1]Esses dados são provenientes da cooperação entre a Universidade Federal de Campina Grande e a Hewlett-Packard Brasil Ltda e não encontram-se disponíveis para acesso público.

métricas em nível de negócio, ou seja, métricas que impactam diretamente o cliente e o provedor do serviço de IaaS, métricas estas que são: (i) quantidade de violações de SLO, que corresponde ao número de vezes em que a demanda da aplicação não pôde ser suprida pelos recursos a ela alocados; (ii) custo de provisionamento, ou seja, a quantidade de horas-máquina necessárias para executar uma dada aplicação.

1.4 Organização do Documento

O restante deste documento encontra-se organizado da seguinte forma. No Capítulo 2 são definidos detalhadamente o problema de provisionamento automático abordado por esta pesquisa, incluindo as suposições consideradas, e descritos termos e notações utilizados ao longo do documento. Em seguida, no Capítulo 3, uma revisão da literatura é realizada, no que concerne o planejamento de capacidade e o provisionamento automático de recursos no contexto de infraestruturas virtualizadas, a fim de construir o embasamento teórico necessário para o melhor entendimento das contribuições deste trabalho.

O arcabouço para provisionamento automático de recursos que utiliza uma abordagem híbrida e não intrusiva é proposto no Capítulo 4, juntamente com o embasamento que justifica a utilização de múltiplos preditores, construído através da experimentação e avaliação do componente de seleção. Além do mais, neste mesmo capítulo são definidos os preditores contemplados neste trabalho, caracterizados os dados utilizados nas simulações e especificadas as métricas usadas tanto na avaliação do seletor como na avaliação do arcabouço proposto.

No Capítulo 5, o arcabouço proposto é instanciado e avaliado segundo as métricas definidas no capítulo anterior. Por fim, no Capítulo 6 são apresentadas as considerações finais, elencados os potenciais efeitos das simplificações realizadas no presente estudo e levantados possíveis trabalhos futuros.

Capítulo 2

Problema Investigado

Neste capítulo o problema investigado é definido em detalhes, através da abordagem dos temas de infraestrutura como serviço e de provisionamento de recursos a curto prazo e da elicitação das suposições e simplificações consideradas no contexto desta pesquisa.

2.1 Infraestrutura como Serviço

Infraestrutura como Serviço (IaaS) faz parte do conjunto de tipos de serviços pertencentes ao paradigma de *Computação na Nuvem*, tal como *Software* como Serviço (SaaS, *Software as a Service*), Plataforma como Serviço (PaaS, do inglês *Platform as a Service*), dentre outros. IaaS se caracteriza por oferecer recursos computacionais (poder de processamento, memória, disco, etc) como um serviço, através da disponibilização imediata de recursos, da tarifação sob demanda e da ausência de comprometimentos futuros entre cliente e provedor, ou seja, não existe duração preestabelecida para a relação de prestação de serviço. Desta forma, ao utilizar esse tipo de serviço o cliente delega ao provedor, por exemplo, as obrigações de: (i) compra de servidores físicos e equipamentos; (ii) aquisição de licenças, como licenças de gerenciadores de banco de dados e de sistemas operacionais; (iii) suporte a infraestrutura física, ou seja, gerenciamento de espaço físico, de energia elétrica, do sistema de arrefecimento, etc. Os atuais modelos de IaaS destacam-se pela elasticidade (capacidade de alocar e liberar máquinas virtuais de acordo com a demanda da aplicação), flexibilidade (diversidade de tipos de instância e sistemas operacionais), confiabilidade e segurança na provisão do serviço.

A relação de prestação de serviço estabelecida entre o cliente e o provedor de IaaS é mediada através de um contrato de nível de serviço que é responsável por referir as expectativas do cliente e a prestação de serviço do provedor de *Computação na Nuvem* [7]. O SLA firmado garante que a disponibilidade, escalabilidade, confiabilidade e segurança do serviço prestado serão mantidas.

Em particular, esse tipo de serviço é provido através de infraestruturas virtualizadas — máquinas virtuais — que são hospedadas nos servidores do centro de processamento de dados do provedor de *Computação na Nuvem* e que são utilizadas para executar os serviços e aplicações dos clientes do provedor. O centro de processamento de dados define um conjunto de tipos de instâncias — máquinas virtuais com diferentes características — que podem ser providas por esses servidores. Cada tipo de instância caracteriza uma máquina virtual em termos de capacidade de recursos (CPU, memória RAM, largura de banda, e assim por diante).

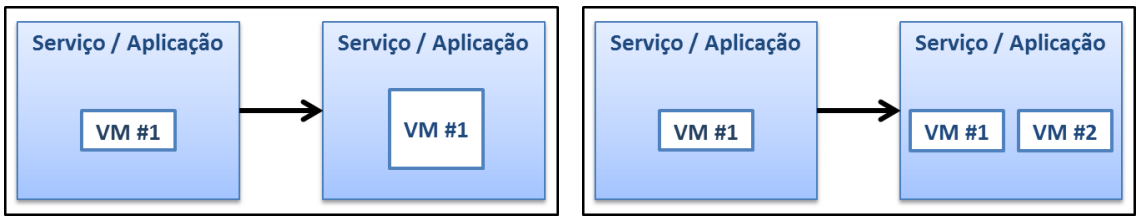
Portanto, quando um cliente contrata um serviço de IaaS de um provedor, ele na verdade está adquirindo um determinado tipo de instância de máquina virtual, a qual está hospedada em algum dos servidores do provedor, para executar uma aplicação de interesse do cliente, com os direitos e os deveres de ambas as partes da relação de negócio geridos por um SLA.

2.2 Provisionamento de Recursos a Curto Prazo

O provisionamento de recursos a curto prazo em ambientes de IaaS consiste no gerenciamento da quantidade de recursos alocados, nos próximos minutos, para um dado serviço ou aplicação, que está executando na infraestrutura, em resposta a mudanças na demanda do serviço. Essas mudanças na demanda podem ser monitoradas a partir de uma variedade de métricas, como taxa de utilização de recursos, tempos de resposta, tamanhos de filas, dentre outras, que determinam o nível de intrusão do modelo de provisionamento. Neste trabalho é considerado um modelo de provisionamento minimamente intrusivo, que considera apenas os dados de utilização dos recursos virtuais, ou seja, um modelo que não é intrusivo à aplicação que está sendo provida.

Essencialmente, esse tipo de provisionamento pode ser realizado de duas formas, como mostrado na Figura 2.1, que são: (i) vertical, quando variações na demanda da aplicação são

supridas modificando-se a capacidade da máquina virtual que está executando a aplicação; (ii) horizontal, quando a quantidade de máquinas virtuais alocadas é alterada com o objetivo de adaptar-se a mudanças na demanda da aplicação. É bem verdade que existem aplicações e serviços que não são horizontalmente escaláveis, no entanto, o foco desse trabalho é em aplicações que possuem essa propriedade. Desse modo, que quando empregado isoladamente, o termo provisionamento, possuirá o sentido de provisionamento horizontal. Serão descritas, em maiores detalhes, as características desse tipo de aplicação na Seção 2.3.



(a) Provisionamento vertical (b) Provisionamento horizontal

Figura 2.1: Diferentes formas de provisionamento de recursos.

2.3 O Perfil da Aplicação Provisionada

Neste trabalho, assume-se que as aplicações são provisionadas horizontalmente, como destacado na seção anterior, e que um serviço de balanceamento de carga é utilizado para distribuir e equilibrar a demanda da aplicação entre todas as máquinas virtuais alocadas para esse serviço. Além do mais, considera-se que estas aplicações que executam na infraestrutura podem ser compostas por múltiplas camadas, ou seja, uma ou mais camadas.

Cada uma destas camadas pode apresentar variações nas cargas de trabalho, o que significa que a quantidade de máquinas virtuais necessárias para executar uma camada da aplicação, com o nível de desempenho adequado, pode variar com o tempo. Logo, cada camada da aplicação é vista como um componente que deve ser provisionado independentemente. Aqui, por questões de simplificação, considera-se que cada camada da aplicação está associada a um tipo de instância, ou seja, uma camada da aplicação só pode ser executada em um único tipo de instância, enquanto camadas diferentes podem ser executadas por tipos diferentes.

Na Figura 2.2 é apresentada uma visão geral do provisionamento de recursos conside-

rado nesta pesquisa. Basicamente, existem dois componentes principais, a infraestrutura de *Nuvem* e o módulo de provisionamento. A infraestrutura é responsável por prover e gerenciar os recursos virtuais e realizar o balanceamento de carga entre as máquinas virtuais alocadas para cada camada de serviço, enquanto que o módulo de provisionamento realiza a provisão de recursos propriamente dita, sendo este módulo subdivido em dois componentes: (i) o monitor, que realiza periodicamente a coleta de dados dos serviços em execução; (ii) o controlador, que é responsável por realizar ações de provisionamento a partir dos dados recebidos do monitor.

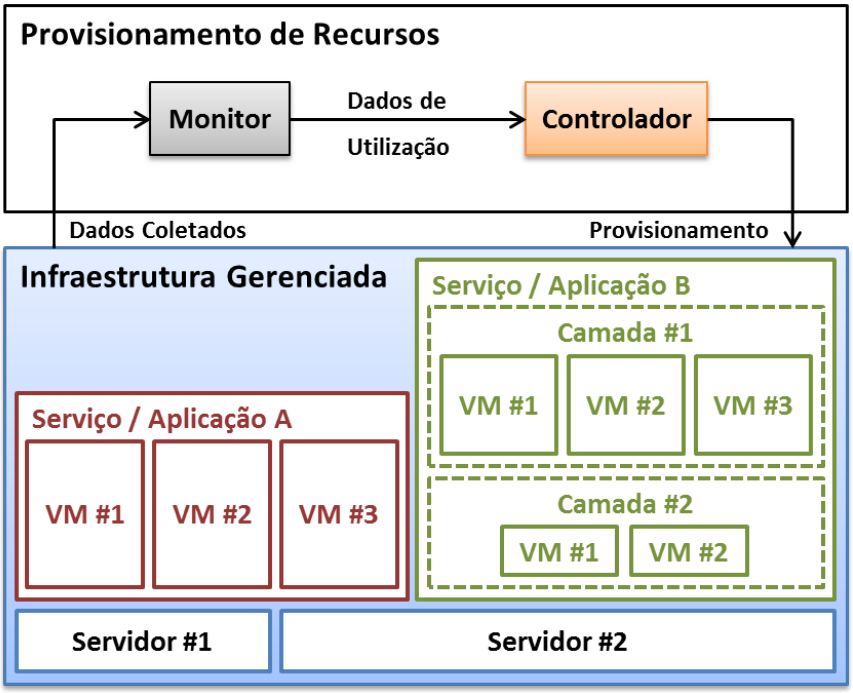


Figura 2.2: Visão geral do modelo de provisionamento de recursos.

As camadas da aplicação estão indiretamente relacionadas a SLAs firmados entre os usuários e o provedor da aplicação, que por consequência, também estão associadas a um ou mais objetivos de nível de serviço. Assume-se a existência de um mapeamento que relaciona a satisfação dos SLOs com a utilização de recursos das máquinas virtuais alocadas ao serviço, de tal forma que se a utilização do recurso for mantida abaixo de um valor alvo, então os SLOs da camada do serviço serão satisfeitos. Ou seja, quanto mais próxima do alvo a utilização de recursos for mantida, menor é o número de máquinas virtuais necessárias e menor é o custo para executar esta camada do serviço com o nível de desempenho desejado.

2.4 Definição do Problema

Idealmente, a melhor forma de proceder o provisionamento horizontal de recursos a curto prazo em ambientes de IaaS é através de um mecanismo automático que necessita do mínimo grau de intrusão. Além disso, esse mecanismo deve prover a quantidade exata de máquinas virtuais necessárias para manter em níveis aceitáveis o desempenho das camadas da aplicação em execução enquanto garante que os SLOs são mantidos. No entanto, um mecanismo que contemple toda essa eficiência consiste em um cenário não realista para as soluções e tecnologias existentes atualmente.

Por outro lado, um objetivo acessível e ao mesmo tempo uma lacuna no atual contexto de *Computação na Nuvem* e de serviços de IaaS consiste em um mecanismo de provisionamento que realiza automaticamente o planejamento de capacidade a curto prazo de forma não intrusiva e que seja capaz de reduzir os custos de provisionamento. Uma vez que, utiliza-se apenas dados de utilização de recursos e minimiza-se a quantidade de recursos ativos necessários para satisfazer os SLOs estabelecidos.

Capítulo 3

Trabalhos Relacionados e Fundamentação Teórica

Neste capítulo são descritos e analisados trabalhos relacionados com a gerência de capacidade de recursos em ambientes virtualizados. A partir de um levantamento do estado da arte, estes trabalhos são classificados e organizados segundo a natureza do funcionamento do mecanismo de gerência, considerando-se desde aspectos como o horizonte de planejamento de capacidade até o modo de operação e o nível de intrusão do mecanismo.

Em primeiro lugar, quando o tema de gerência de capacidade em ambientes de recursos virtuais é abordado por pesquisas desta área, em geral, são consideradas uma ou mais das seguintes atividades intrínsecas ao tema em questão:

- **Planejamento de capacidade a longo prazo**: responsável por decidir quantos recursos são necessários para compor uma infraestrutura computacional no longo prazo, ou seja, no período mínimo de um ano [28];

- **Planejamento de capacidade a médio prazo**: planejamento utilizado para estipular a quantidade de recursos físicos que devem estar operacionais nas próximas horas ou dias, tendo como objetivo principal a redução de custos provenientes do consumo de energia elétrica [24];

- **Planejamento de capacidade a curto prazo**: decide quantos recursos — máquinas virtuais — são necessários para implantar e executar uma aplicação ou serviço em específico durante os próximos minutos [8, 29, 41, 46];

- **Escalonamento de máquina virtual**: estabelece onde devem ser criadas, dentre os recursos físicos disponíveis, as máquinas virtuais necessárias, na tentativa de maximizar a utilização de recursos e reduzir os custos, sem comprometer a qualidade de serviço da aplicação provida [2, 22, 42].

Todas estas atividades, que também podem ser vistas como áreas de estudo, encontram-se fortemente relacionadas e são essencialmente dependentes. O gerente de capacidade a curto prazo, ao prover um dado serviço, precisa alocar recursos através de máquinas virtuais instanciadas sobre os recursos físicos disponíveis, o que é realizado pelo de gerente de escalonamento de máquina virtual, responsável por decidir onde as máquinas virtuais devem ser criadas. O gerente de escalonamento de máquina virtual utiliza os recursos físicos que encontram-se operacionais, que foram ligados a partir da decisão tomada pelo gerente de capacidade a médio prazo, que por sua vez, controla os recursos físicos disponíveis decorrentes da decisão tomada pelo gerente de capacidade a longo prazo. Entretanto, o foco deste trabalho de mestrado encontra-se no planejamento de capacidade a curto prazo.

Por outro lado, quando analisadas em profundidade, as soluções de planejamento de capacidade a curto prazo podem ser comparadas segundo dois aspectos fundamentais: o modo de operação e o nível de intrusão dos mecanismos. Alguns gerentes de capacidade destas soluções possuem modo de operação reativo, que apenas respondem a mudanças na demanda aumentando ou diminuindo a capacidade dos serviços caso seja necessário, e que, em geral, realizam provisionamento horizontal [8, 9, 15, 27, 29, 30], apesar de existirem trabalhos que exploram a provisão de recursos de forma vertical [48] ou que utilizam os dois tipos de provisionamento [6].

Outros gerentes de capacidade adotados por estas soluções utilizam uma abordagem proativa, que emprega técnicas de análise de dados e heurísticas com o intuito de estimar cargas futuras. Estes, por sua vez, também encontram-se classificados entre os que realizam provisionamento horizontal [34, 47], provisionamento vertical [14, 19] e os que fazem uso de ambos os mecanismos de provisionamento [41]. Existem ainda soluções que utilizam gerentes que seguem uma abordagem híbrida [46], proativa e reativa, realizando predições de carga e reagindo, o mais rápido possível, às más decisões de planejamento de capacidade, de forma similar à solução proposta nesta dissertação.

Quanto ao aspecto da intrusão, alguns gerentes a curto prazo são dependentes de aplica-

ção [29,40,46], no sentido de que eles precisam de informações específicas sobre os serviços que estão sendo providos, como: tempos de resposta; tamanhos de filas; taxas de chegada; demandas de requisições. Este requisito, além de suscitar possíveis problemas de privacidade, agrega maiores complexidades à estrutura de gestão de recursos, dado que o serviço provido deve estar devidamente instrumentado para que seja possível seu monitoramento. Por outro lado, existem gerentes que requerem conhecimento sobre o que está sendo provisionado. Por exemplo, Calcavecchia et al. [8] definem um gerente a curto prazo, que necessita apenas do monitoramento da carga das máquinas virtuais, mas assumem que os serviços seguem a estrutura característica de processamento em lote (do inglês *batch*).

O mecanismo de provisionamento a curto prazo proposto neste trabalho funciona de forma não intrusiva e não necessita de informações específicas sobre a aplicação, tampouco possui requisitos de que o serviço provido seja de um determinado tipo. O mecanismo proposto necessita apenas de dados de utilização da infraestrutura virtual, o que pode ser obtido, de forma trivial, no nível de gerência da máquina virtual.

Além do mais, o mecanismo realiza provisionamento horizontal e segue um modo de operação que pode agir de forma proativa e reativa. A abordagem proativa considera a utilização de múltiplos preditores, que são periodicamente selecionados, de forma criteriosa, para gerar estimativas de futuras demandas do serviço, o que concede ao mecanismo de provisionamento a capacidade de adaptação a mudanças no padrão das cargas do serviço. Deste modo, até onde se sabe, o mecanismo de provisionamento automático de recursos virtuais proposto neste trabalho é o primeiro que utiliza modo de operação híbrido, realiza provisionamento horizontal e é de fato não intrusivo. A Tabela 3.1 apresenta um resumo de trabalhos existentes na literatura que apresentam essas características.

Tabela 3.1: Classificação de soluções de planejamento de capacidade a curto prazo segundo o modo de operação, o tipo de provisionamento e o nível de intrusão.

Autores	**Modo de operação**	**Provisionamento**	**Nível de intrusão**
Lim et al. [27]	Reativo	Horizontal	Baixo
Meng et al. [30]	Reativo	Horizontal	Baixo
Calheiros et al. [9]	Reativo	Horizontal	Alto

Tabela 3.1 – Continuação

Autores	Modo de operação	Provisionamento	Nível de intrusão
Fitó et al. [15]	Reativo	Horizontal	Alto
Marshall et al. [29]	Reativo	Horizontal	Alto
Seung et al. [40]	Reativo	Horizontal	Alto
Calcavecchia et al. [8]	Reativo	Horizontal	Alto
Vijayakumar et al. [48]	Reativo	Vertical	Alto
Bonvin et al. [6]	Reativo	Horizontal e vertical	Alto
Roy et al. [34]	Proativo	Horizontal	Alto
Vasić et al. [47]	Proativo	Horizontal	Baixo
Gong et al. [19]	Proativo	Vertical	Baixo
Dawoud et al. [14]	Proativo	Vertical	Alto
Sharma et al. [41]	Proativo	Horizontal e vertical	Alto
Urgaonkar et al. [46]	Reativo e proativo	Horizontal	Alto

Capítulo 4

Arcabouço de Provisionamento Automático Não Intrusivo

Neste capítulo é apresentado um arcabouço de provisionamento automático de recursos não intrusivo para ambientes de infraestruturas como serviço e realizada a fundamentação, através de experimentos de simulação, da utilização de múltiplos preditores pelo mecanismo de provisionamento.

4.1 Arquitetura do Arcabouço de Provisionamento

O arcabouço apresentado neste capítulo baseia-se em elementos da teoria do controle (do inglês *control theory*). Esta teoria consiste na utilização de informações coletadas de um dado sistema monitorado a fim de atingir objetivos externamente definidos, através de mudanças nos parâmetros de entrada do sistema [23]. Ou seja, utilizar-se dos dados monitorados do sistema para dinamicamente alterar a sua configuração e assim manter o desempenho do sistema em um patamar desejado.

Visto que, os dados coletados são utilizados para gerenciar a parametrização do sistema e que essa parametrização afeta na resposta do mesmo, esse modelo pode ser chamado de laço de controle com retroalimentação (do inglês *feedback control loop*). Esse modelo de laço de controle está exemplificado na Figura 4.1.

Conceitualmente, o arcabouço está estruturado em quatro módulos principais, o módulo de monitoramento, de controle, de predição e de seleção, sendo estes os responsáveis

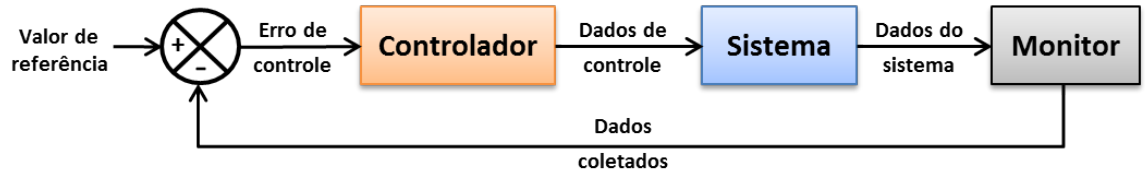


Figura 4.1: Visão geral do modelo de laço de controle.

por manter dinamicamente o desempenho do serviço ou camada de aplicação em um nível aceitável. A arquitetura do arcabouço, bem como, a distribuição de seus módulos podem ser encontradas na Figura 4.2. Esse arcabouço reúne algumas características e propriedades pontuais de soluções específicas de provisionamento automático de recursos propostas tanto pelo mercado com pela literatura, tais como: (i) mecanismo híbrido de provisionamento de recursos, isto é, que atua de forma proativa e reativa; (ii) provisionamento não intrusivo, ou seja, que utiliza apenas dados de utilização de recursos, com o mínimo de intrusão; (iii) configuração dinâmica e automática dos módulos de monitoramento e predição, tornando o modelo de provisionamento adaptável e flexível a aplicações com diferentes padrões de carga.

Ou seja, diferente de outras abordagens, esta solução de planejamento de capacidade a curto prazo baseia-se somente em informações históricas da infraestrutura usada pelas camadas do serviço e não requer nenhuma outra informação sobre o serviço em si. Nesse sentido, a solução não é intrusiva ao serviço em execução na infraestrutura, além de prover mais flexibilidade na forma com a qual as predições são feitas, dado que é possível incorporar quantos preditores se queira sem a necessidade de alterar o funcionamento do modelo de provisionamento.

Realizando-se uma correspondência entre o modelo de teoria do controle e o modelo de provisionamento automático de recursos considerado pelo arcabouço, tem-se que: (i) os módulos de coleta de dados, os **monitores**, de ambos os modelos, possuem em essência a mesma função, de coletar dados do sistema, ou seja, dados de utilização de recursos; (ii) o **controlador** consiste no módulo que calcula os erros de controle, a partir dos dados de utilização e dos valores de referência, e que decide, a partir dos erros de controle, quais ações de provisionamento devem ser tomadas para manter o desempenho do serviço em torno de um valor aceitável; (iii) o **sistema** do modelo de laço de controle corresponde a uma camada da aplicação que está sendo automaticamente provisionada; (iv) os **valores de referência**

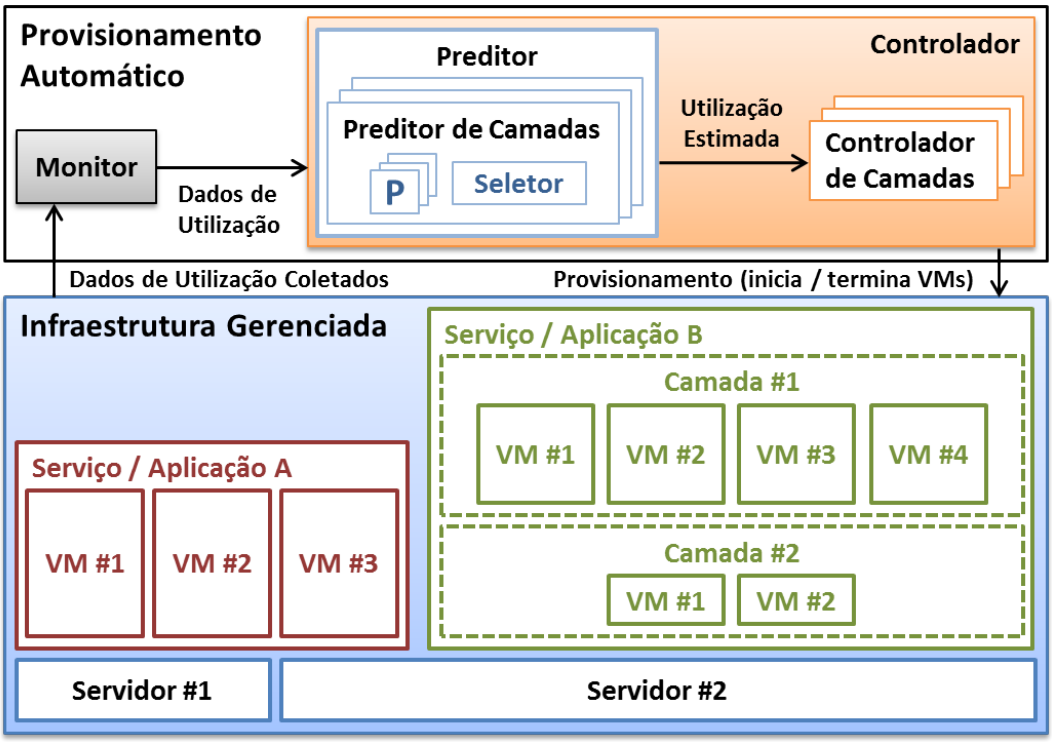


Figura 4.2: Arquitetura do arcabouço para provisionamento automático.

são os níveis de utilização de recursos desejados para esta camada de serviço em específico.

4.1.1 Monitor

O módulo de monitoramento tem como função primordial a coleta de informações sobre a utilização de recurso no nível de máquina virtual. Ou seja, coletar informações sobre a utilização de cada máquina virtual ativa. Considera-se que este módulo possui a habilidade de monitorar um grupo restrito de recursos como consumo de CPU, memória, largura de banda, etc. Além de que, informações sobre diferentes tipos de recursos podem ser coletadas individualmente para cada camada do serviço em execução, com periodicidade configurável.

As características apresentadas por esse módulo são compatíveis com as atuais soluções de monitoramento existentes no mercado de IaaS. Um exemplo desse tipo de solução é o serviço *CloudWatch* da Amazon [38], que apresenta métricas de monitoramento de máquinas virtuais [12] em conformidade com as do arcabouço proposto.

4.1.2 Preditor

O provisionamento de recursos a curto prazo funciona proativamente através de preditores de demanda, de modo que, para cada camada dinamicamente provisionada da aplicação e tipo de recurso monitorado em uma camada, existe um conjunto pré-definido de preditores que tem por objetivo estimar a demanda futura de recursos para aquela camada.

Os preditores são alimentados com as informações de utilização de recursos coletadas pelo módulo de monitoramento e, a partir destas informações, realizam predições da demanda futura da camada para o tipo de recurso em particular. O conjunto de preditores considerado para o provisionamento de cada camada pode ser flexivelmente definido, dado que os preditores diferem entre si apenas na estratégia de predição usada, ou seja, na forma como o preditor explora a informação recebida a fim de estimar a carga futura. De forma geral, cada camada provisionada pode possuir um conjunto diferente de preditores.

4.1.3 Seletor

Considerando qualquer um dos tipos de recurso usado na gerência de capacidade de uma camada de serviço em particular, tem-se que, para cada iteração do laço de controle, um único preditor é utilizado para estimar a demanda futura desse recurso para essa camada provisionada. Para tal, periodicamente, um componente de seleção é utilizado para encontrar dentre o conjunto de preditores disponíveis para uma camada em específico a opção de predição mais apropriada para ser usada nas próximas predições, baseado em simulações de desempenho dos preditores.

Essa contínua seleção faz-se necessária pois, como mostrado na próxima seção, não existe um único preditor que seja eficiente para estimar a ampla variedade de padrões de carga existentes na prática, tanto ao comparar diferentes camadas, como para uma única camada, especialmente ao considerar camadas de serviço que executam por longos períodos de tempo.

4.1.4 Controlador

O controlador é composto por controladores de camada que executam periodicamente, em intervalos configuráveis, para realizar planejamento de capacidade a curto prazo de cada ca-

mada. Os controladores de camada operam de forma reativa e proativa. O comportamento reativo é implementado definindo ações associadas a cada tipo de recurso usado em uma determinada camada. Essas ações são executadas toda vez que a utilização dos recursos atinge um dado limite. Enquanto que o comportamento proativo funciona através da utilização de preditores que geram estimativas de demanda, como descrito na Subseção 4.1.2, que serão usadas pelo controlador de camada para efetuar a gerência de capacidade.

As estimativas derivadas do módulo de predição são periodicamente confrontadas com os valores de referência (por exemplo, nível desejado de utilização) definidos pelo administrador da infraestrutura e/ou pelo cliente. Estas estimativas, juntamente com os valores de referência, são utilizadas pelo controlador para decidir como atuar no sistema para fazer com que o mesmo permaneça em, ou passe para, um estado desejado. Ou seja, o controlador pode atuar no sistema iniciando novas máquinas virtuas ou parando máquinas virtuais em execução para manter a utilização dos recursos o mais próximo possível dos valores de referência associados a esses tipos de recurso.

Cada controlador estipula o número de máquinas virtuais necessárias, no próximo intervalo de controle, para manter a camada em execução, baseado nos valores de referência de utilização e nas estimativas para futuros valores de demanda dos recursos usados pelo serviço. Se a quantidade necessária de máquinas virtuais é maior do que a atualmente alocada, máquinas virtuais adicionais são iniciadas. Caso contrário, a ação a ser tomada depende do tipo da infraestrutura provida. Em um modelo público de IaaS, por exemplo, os provedores cobram geralmente um preço fixo por máquina virtual usada em uma unidade de tempo pré-definida, por exemplo US$ 0.10 por hora. Nesse caso, o controlador deve somente desligar máquinas virtuais que estejam próximas de completar a hora. Por outro lado, em um modelo privado de IaaS, máquinas virtuais devem ser desligadas o mais cedo possível. Independente do modelo de IaaS, a remoção de máquinas virtuais é limitada à quantidade mínima, previamente definida, de máquinas virtuais que devem estar alocadas para cada camada do serviço.

Obviamente, deve haver algum tipo de orquestração entre os controladores de camadas, tal que decisões tomadas por um controlador não afetem outras camadas do mesmo serviço e que pedidos simultâneos de criação de máquinas virtuais sejam devidamente satisfeitos pela infraestrutura física. Isto, no entanto, encontra-se fora do escopo desta pesquisa.

4.2 Decidindo por Múltiplos Preditores

A decisão de usar múltiplos preditores selecionados dinamicamente ao longo do tempo baseia-se em evidências empíricas de que essa abordagem apresenta melhores resultados do que quando é usado um único preditor, mesmo que este preditor combine características de diversos outros, como quando é utilizada uma abordagem que pondera a relevância entre preditores [24].

Essas evidências foram conseguidas através de experimentos de simulação que verificam os efeitos do emprego de diferentes preditores para uma mesma camada de serviço e para diferentes camadas de serviço. Maiores detalhes sobre esses experimentos são apresentados nas próximas subseções.

4.2.1 Modelo de Simulação

Um modelo de simulação do arcabouço apresentado na seção anterior foi implementado através da linguagem de programação R [1]. As simulações e execuções deste modelo foram guiadas a partir de rastros de utilização de recursos de serviços *web*, em produção, de clientes da HP. Cada um desses rastros de utilização corresponde a uma camada diferente do serviço em execução. Visto que, cada controlador de camada funciona de forma autônoma, ou seja, é independente de outros controladores de camada, e que o planejamento de capacidade da infraestrutura do provedor de IaaS não pertence ao escopo deste trabalho, tem-se que cada controlador de camada pode ser simulado de forma individual.

Os rastros de utilização provêm itens coletados com periodicidade de 5 minutos. Cada um destes itens consiste em uma dupla $\langle u, c \rangle$, onde u é a utilização média de CPU no intervalo de 5 minutos e c é o número de núcleos de CPU alocados à aplicação durante esse tempo. A leitura de uma nova dupla permite computar a utilização real das máquinas virtuais no sistema simulado para os próximos 5 minutos. O modelo considera máquinas virtuais com um único núcleo de CPU, ou seja, a quantidade de núcleos alocados corresponde à quantidade de máquinas virtuais disponíveis para uma camada do serviço.

A utilização real de cada máquina virtual entre o tempo t, instante de tempo no qual uma nova dupla $\langle u, c \rangle$ foi lida, e $t + 5$ é computada como sendo o mínimo entre 100% e

[1] O código-fonte encontra-se disponível no endereço http://www.lsd.ufcg.edu.br/∼fabio/autoflex.

$u \times \frac{c}{a}$, onde a é o número de máquinas virtuais alocadas para executar a camada do serviço no experimento de simulação. Além do mais, se $u \times \frac{c}{a} > 100\%$, então o excesso da demanda que não pôde ser alocado no tempo t, dado por $(u \times \frac{c}{a}) - 100\%$, é adicionado à demanda do próximo intervalo de tempo, que ocorre em $t + 5$ minutos.

O controlador de camada pode decidir, de forma proativa, adicionar ou remover máquinas virtuais a cada 5 minutos ou, de forma reativa, toda vez que uma determinada condição ocorrer. Por exemplo, o controlador reage quando a utilização de CPU está acima de um dado limite preestabelecido. As decisões de provisionamento são baseadas na estimativa do número de VMs que devem estar alocadas no próximo período de controle, dado por $\left\lceil \frac{\hat{u}}{\rho} \right\rceil$, onde $\hat{u}$ é a estimativa de utilização para o próximo período e ρ é o valor de referência de utilização.

No entanto, visto que é necessário um tempo até que o provisionamento das novas VMs tenha efeito, no modelo de simulação considera-se que se um controlador de camada decidir no tempo t que uma nova máquina virtual deve ser alocada, tal VM estará de fato operacional apenas no tempo $t + 5$. Dessa forma, os preditores no controlador de camada usam informações históricas coletadas até o tempo t para estimar a demanda no tempo $t + 5$ minutos e usam essa estimativa para decidir, no tempo t, quantas máquinas virtuais devem estar executando no tempo $t + 5$ minutos. Portanto, qualquer quantidade extra de máquinas virtuais necessária no tempo $t + 5$ minutos é requisitada no tempo t.

Particularmente, o modelo de simulação considera apenas o cenário público de IaaS e um formato de tarifação que realiza cobrança por hora pelo uso de máquinas virtuais. Ou seja, máquinas virtuais são terminadas somente em horas completas, como discutido na Subseção 4.1.4.

Algoritmos de Predição

As simulações fizeram uso de 5 preditores, exaustivamente difundidos na literatura, para a realização de estimativas de demandas de serviço. Estes preditores são baseados em: (i) medições de demandas anteriores; (ii) autocorrelação; (iii) regressão linear; (iv) autorregressão; (v) autorregressão com média móvel integrada. Também foi avaliado um sexto preditor que usa os outros 5 preditores de forma combinada, como proposto por Jiang et al. [24].

Medições de Demandas Anteriores Esse algoritmo realiza predições de futuras demandas de utilização através da repetição de valores de utilização coletados do sistema na última janela de tempo (LW, do inglês *Last Window*).

$$\widehat{U}(t+5) = U(t-5) \tag{4.1}$$

Onde $\widehat{U}(t)$ corresponde ao valor estimado de utilização para o tempo t e $U(t)$ é a utilização real durante a janela de tempo t.

Autocorrelação A técnica de autocorrelação (AC, do inglês *Auto-Correlation*) visa encontrar um padrão de repetição nos dados coletados. Para tal, o algoritmo calcula a correlação dos dados de utilização originais com estes mesmos dados deslocados, de forma crescente, no tempo. O algoritmo realiza n deslocamentos nos dados e verifica o tamanho p do deslocamento que gerou a maior correlação, dado que p varia de 1 até n e que n corresponde a metade do comprimento dos dados. O valor de predição é baseado no tamanho p do padrão de repetição, ou seja, o valor de utilização da p-ésima janela de tempo é repetido para o próximo valor de predição [19]. Se nenhum padrão de repetição é encontrado, o algoritmo usa o preditor LW para realizar a predição.

Regressão Linear O modelo de regressão linear (LR, do inglês *Linear Regression*) estima futuros valores de utilização a partir de uma função derivada através de uma regressão linear dos valores históricos de utilização das últimas p janelas de tempo. O valor de p é o maior tamanho de deslocamento aplicado pela função de autocorrelação que possui um coeficiente de correlação aceitável. Os dados de utilização são considerados previsíveis, com uma precisão adequada, se o coeficiente de autocorrelação dos dados é maior ou igual a 0.3 [10], visto que esse é o limiar para avaliar se um coeficiente de correlação é aceitável. Previsões são feitas usando a função apresentada na Equação 4.2.

$$\widehat{U}(t+5) = lm(U(t) \sim x)(t+5), \text{ para } x = t-5, t-10, \cdots, t-(p \times 5) \tag{4.2}$$

Onde, $lm(U(t) \sim x)$ é um modelo linear do valores passados de utilização em função de

suas janelas de tempo.

Autoregressão Um modelo de autoregressão (AR, do inglês *Auto-Regressive*) prevê a utilização da demanda no tempo $t + 5$ com base em uma combinação linear ponderada dos p valores anteriores de dados históricos de utilização [10, 13, 21], como apresentado na Equação 4.3.

$$\widehat{U}(t+5) = a_1 \cdot U(t-5) + a_2 \cdot U(t-10) + \cdots + a_p \cdot U(t-(p \times 5)) \tag{4.3}$$

A ordem p do modelo AR é definida por meio de um teste estatístico com base na função de autocorrelação parcial, chamado de critério de informação de Akaike (AIC, do inglês *Akaike Information Criterion*) [35], e pode assumir valores inteiros no intervalo $[2, p.max]$, onde $p.max$ corresponde à quantidade de dados históricos recebidos pelo modelo. Os coeficientes a_1, a_2, ... a_n consistem na solução do sistema linear composto por equações construídas a partir dos coeficientes de autocorrelação dos dados históricos de utilização [21].

Autoregressão com Média Móvel Integrada As previsões desse modelo de autoregressão com média móvel integrada (ARIMA, do inglês *Auto-Regressive Integrated Moving Average*) são obtidas através de d diferenciações da sequência não estacionária de dados de utilização do passado e do ajuste de um modelo de autoregressão com média móvel (ARMA, do inglês *Auto-Regressive Moving Average*), que é composto pelo modelo AR em conjunto com um modelo de média móvel (MA, do inglês *Moving Average*) [10]. As predições realizadas por esse modelo são dadas a partir da Equação 4.4.

$$\begin{aligned}\widehat{U}(t+5) = {} & a_1 \cdot U(t-5) + a_2 \cdot U(t-10) + \cdots + a_p \cdot U(t-(p \times 5)) \\ & + b_1 \cdot e(t-5) + b_2 \cdot e(t-10) + \cdots + b_q \cdot e(t-(q \times 5))\end{aligned} \tag{4.4}$$

A média móvel é uma combinação linear dos ruídos, $e(t-5, \cdots, e(t-(q \times 5))$, ponderados pelos coeficientes lineares b_1, ..., b_q. O modelo ARIMA é caracterizado por três parâmetros, o número p de pontos do passado que são considerados, calculado a partir do mesmo

método usado no modelo AR, o número q de valores residuais e o número d de diferenciações. Nos experimentos realizados foi utilizada uma implementação do ARIMA que calcula automaticamente os parâmetros p e q e define o valor de d em zero, por considerar-se que a distribuição de dados de utilização é um processo estacionário, no qual as características estatísticas não se modificam no tempo [21].

Combinação de Preditores O algoritmo de combinação de preditores (EN, do inglês *Ensemble*) utiliza uma estratégia de combinação linear ponderada de um conjunto pré-determinado de preditores para realizar estimativas de demanda por meio da Equação 4.5 [24].

$$\widehat{U}(t) = \sum_{p \in P} W_p(t) \cdot \widehat{U}_p(t), \text{ sujeito a } \sum_{p \in P} W_p(t) = 1 \tag{4.5}$$

$$W_p(t) = \frac{e_p(t)}{\sum_{x \in P} e_x(t)} \tag{4.6}$$

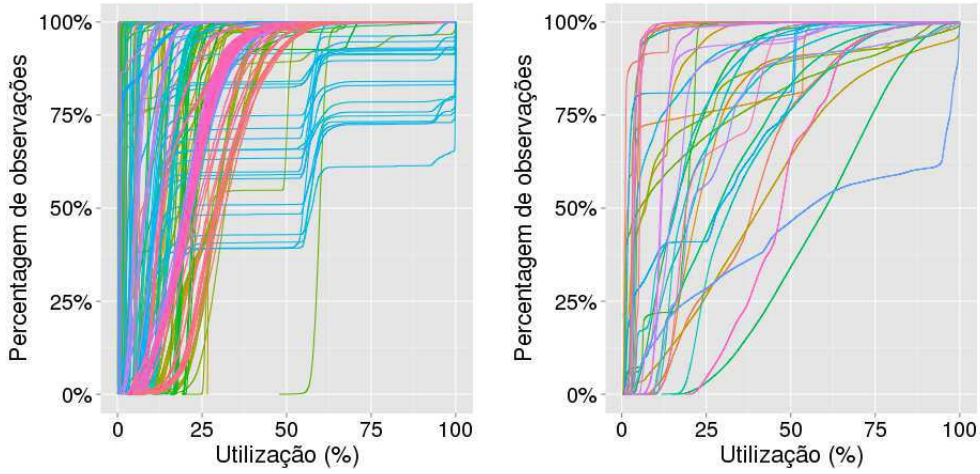
Onde P é o conjunto de preditores considerados, $\widehat{U}_p(t)$ é o valor estimado pelo preditor p no tempo t, $W_p(t)$ é o peso do preditor p no tempo t (Equação 4.6) e $e_p(t)$ é o erro relativo causado pelo preditor p no tempo t.

Rastros de Utilização de Recursos

Nos experimentos de simulação foram usados dois conjuntos de rastros de utilização CPU de recursos de aplicações em produção, um contendo 256 arquivos com duração de um mês e outro com 33 arquivos com duração média de 8 meses. Com o intuito de caracterizar a ampla diversidade de padrões de utilização presentes nos dados utilização considerados foram realizadas duas análises diferentes para avaliar quão representativos são esses dados.

Primeiramente, foi calculada a função distribuição acumulada (FDA) dos dados de utilização de CPU a fim de tornar possível a análise da variação de amplitudes das demandas. A partir desses resultados, conclui-se que os dados apresentam uma ampla variedade de distribuições de utilização para ambos os conjuntos de dados. Ou seja, existem rastros, nos dois conjuntos de dados considerados, que na maior parte do tempo possuem um nível de

utilização que é diferente dos apresentados pelos demais rastros. Esse comportamento pode ser visto na Figura 4.3.



(a) (b)

Figura 4.3: FDA da utilização de CPU para: (a) 265 arquivos de rastros com duração de um mês e (b) 33 arquivos de rastros com 8 meses de duração média.

Em uma segunda análise, a FDA foi gerada para as variações de utilização de CPU, definidas como a diferença entre as utilizações no tempo t e no tempo $t + 5$ minutos, para todos os valores de t. Através dos resultados da FDA é possível observar que cada um dos rastros, dos dois conjuntos de dados, apresenta uma grande diversidade de pequenas variações de utilização de CPU e um pequeno percentual de variações intensas de utilização, positivas e negativas. Os resultados da FDA podem ser observados nas Figuras 4.4 e 4.5.

Desta forma, a partir das análises realizadas, é possível verificar que os dados de utilização usados para alimentar os experimentos de simulação são de fato representativos para os estudos desenvolvidos nesse trabalho de dissertação.

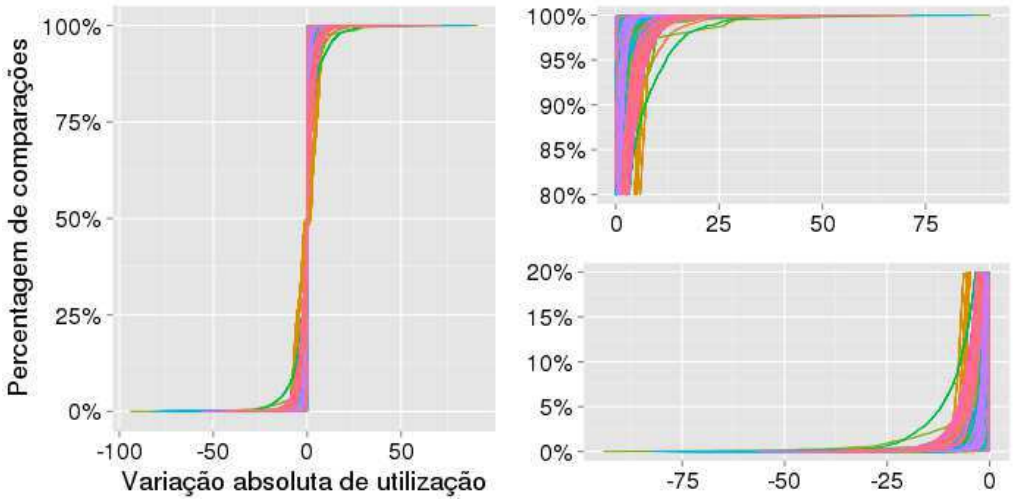


Figura 4.4: FDA da variação absoluta de utilização de CPU para 265 arquivos de rastros com duração de um mês.

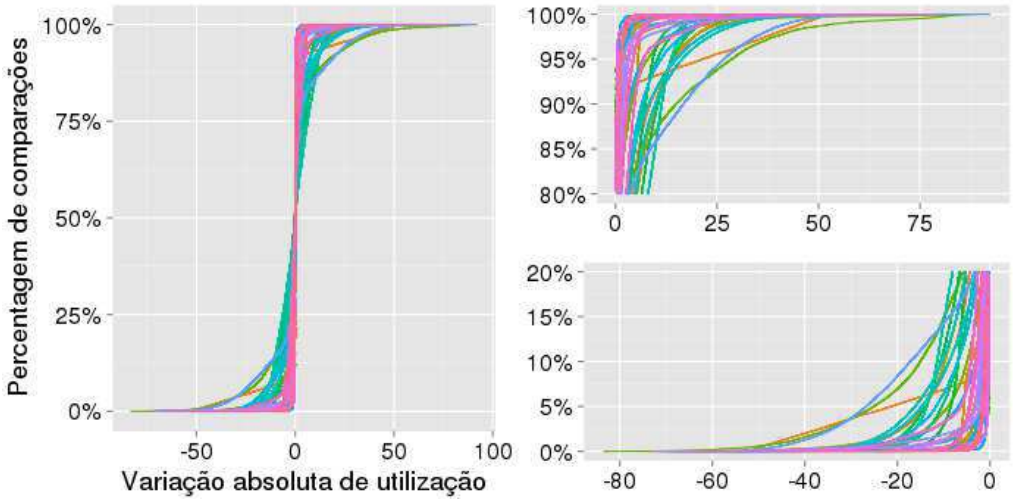


Figura 4.5: FDA da variação absoluta de utilização de CPU para 33 arquivos de rastros com 8 meses de duração média.

Métricas de Avaliação

Durante os experimentos de simulação realizados foram consideradas como métricas independentes de avaliação de desempenho: o número de **violações de SLO** do serviço e a **economia de custos** de provisionamento alcançada. Violações de SLO consistem no número de

intervalos de tempo onde a capacidade provida não foi suficiente para suprir a demanda do serviço, ou seja, os intervalos de 5 minutos em que a utilização de CPU foi de 100% e havia demanda que não pôde ser atendida, e foi, portanto, deslocada para o próximo intervalo de tempo[2].

Por outro lado, considerando mais o nível de negócio, tem-se a segunda métrica considerada, que é definida como a economia conseguida, em termos de quantidade de recursos, quando comparado o provisionamento automático com uma abordagem que realiza superprovisionamento perfeito. Ou seja, uma abordagem não realística que conhece a demanda mais elevada de todo o período simulado e estaticamente aloca recursos tal que a utilização seja sempre menor que um dado valor limite, por exemplo 70%. [3].

4.2.2 Experimentos de Simulação

A fim de fundamentar a utilização de múltiplos preditores foram realizados inicialmente dois experimentos de simulação. Estes experimentos evidenciam a necessidade de utilização de diferentes preditores para diferentes padrões de carga, tanto para camadas distintas como para uma mesma camada de serviço.

Diferentes Preditores para Diferentes Camadas de Serviço

Em um primeiro experimento o modelo de simulação foi instanciado com um único preditor, ou seja, cada controlador de camada utilizou apenas um preditor durante a experimentação do modelo. Essa simplificação na instanciação do modelo torna o mecanismo de seleção uma tarefa trivial, visto que apenas um preditor é usado para estimar as demandas de uma camada de serviço. Particularmente, o controlador foi configurado para manter a utilização de CPU abaixo do limite de 70%. A parametrização do modelo de simulação pode ser encontrada de forma detalhada na Tabela 4.1

[2]Essa métrica só pode ser computada em experimentos de simulação, visto que em ambientes não simulados não há como medir a demanda real quando a utilização está em 100%.

[3]Esse limite é considerado no cenário perfeito com o intuito de fazer referência ao superprovisionamento real, que busca manter a utilização sempre abaixo de um limiar preestabelecido.

Tabela 4.1: Parâmetros utilizados na execução do modelo de simulação para análise de diferentes preditores para diferentes camadas de serviço.

Parâmetro	**Valor**
Quantidade de rastros de utilização	265 arquivos
Duração média dos rastros de utilização	1 mês
Conjunto de preditores	LW, LR, AC, AR, ARIMA e EN
Tamanho do histórico de dados usado por predição	1 semana
Tamanho do histórico de dados usado por seleção	—
Periodicidade de monitoramento	5 minutos
Periodicidade de predição	5 minutos
Periodicidade de seleção	—
Periodicidade de controle	5 minutos
Valor de referência de utilização	70%
Valor limite de superprovisionamento	70%
Quantidade mínima de VMs alocadas	1
Número de núcleos por VM	1
Tamanho da fatia da tarifação	60 minutos

O modelo foi simulado para cada uma das duplas $\langle camada, preditor \rangle$ e em seguida as métricas de quantidade de violações de SLO e de economia de custos foram calculadas. A partir da análise das métricas foi verificado a percentagem de camadas de serviço para as quais um dado preditor foi avaliado como o melhor[4], considerando ambas as métricas em separado. Os resultados desta verificação podem ser observados na Figura 4.6.

Como pode ser observado, quando considerado o número de violações de SLO, o preditor com melhor desempenho responde apenas por 44% dos casos, enquanto esse número aumenta para 64% quando a métrica de economia é usada para avaliar os preditores. Dessa forma, é possível perceber que, uma vez que não existe um único preditor que apresenta

[4]A soma das partes no gráfico pode ser maior que 100% dado que, para algumas das camadas de serviço, vários preditores foram igualmente bons.

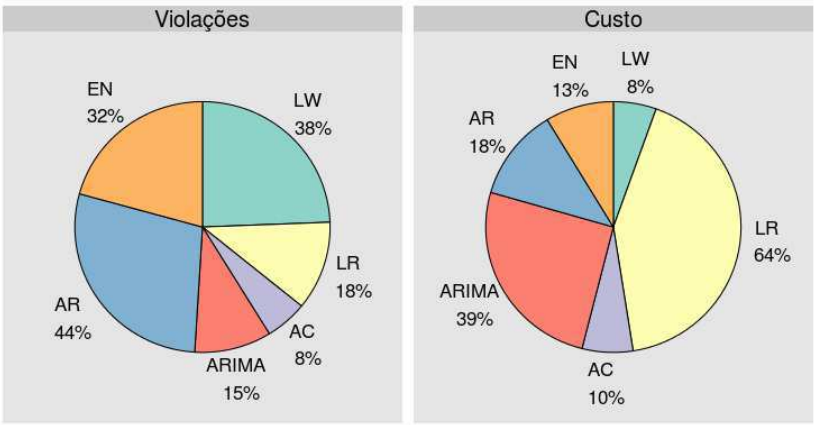


Figura 4.6: Percentagem de vezes em que um dado preditor é avaliado como o melhor.

o melhor desempenho na maioria dos casos, utilizar múltiplos preditores tem um impacto direto no desempenho alcançado com o modelo de provisionamento proposto.

Diferentes Preditores para Uma Mesma Camada de Serviço

Um outro experimento de simulação foi realizado a fim de avaliar como a qualidade de um preditor varia ao longo do tempo ao estimar demandas de uma mesma camada de serviço. O experimento foi alimentado com rastros de utilização de longa duração, em média 8 meses, e configurado para atuar com os mesmos preditores utilizados no primeiro experimento. As demais parametrizações estão detalhadas na Tabela 4.2.

Tabela 4.2: Parâmetros utilizados na execução do modelo de simulação para análise de diferentes preditores para uma mesma camada de serviço.

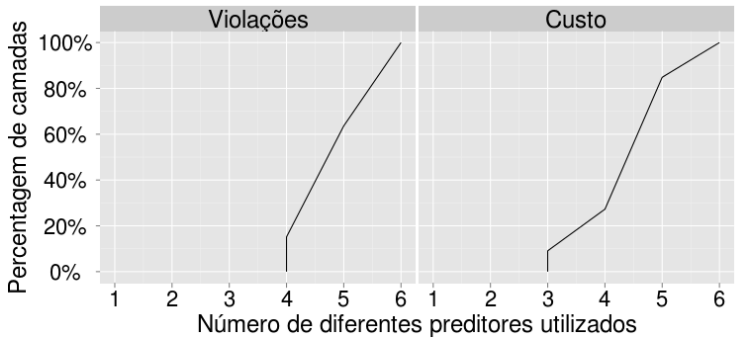
Parâmetro	Valor
Quantidade de rastros de utilização	33 arquivos
Duração média dos rastros de utilização	8 meses
Conjunto de preditores	LW, LR, AC, AR, ARIMA e EN
Tamanho do histórico de dados usado por predição	2 semanas
Tamanho do histórico de dados usado por seleção	24 horas
Periodicidade de monitoramento	5 minutos

Tabela 4.2 – Continuação

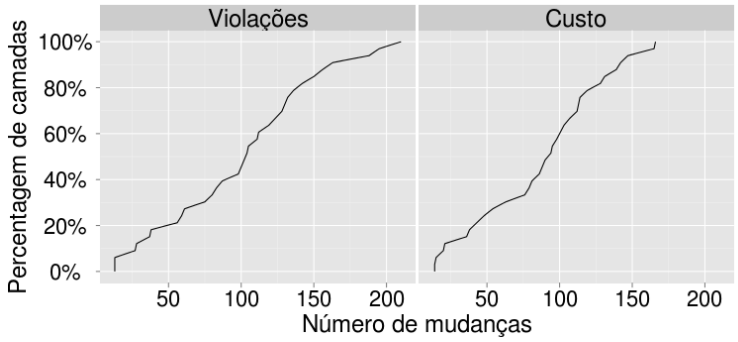
Parâmetro	**Valor**
Periodicidade de predição	5 minutos
Periodicidade de seleção	24 horas
Periodicidade de controle	5 minutos
Valor de referência de utilização	70%
Valor limite de superprovisionamento	70%
Quantidade mínima de VMs alocadas	1
Número de núcleos por VM	1
Tamanho da fatia da tarifação	60 minutos

Neste experimento um seletor perfeito é usado para escolher o preditor mais apropriado a cada dia. Para tal, os preditores são alimentados com dados de utilização das duas semanas anteriores e a utilização do próximo dia é usada para medir o desempenho de cada preditor e para definir o melhor preditor segundo as métricas estabelecidas. As Figuras 4.7(a) e 4.7(b) apresentam, respectivamente, as funções distribuição acumulada do número de diferentes preditores escolhidos por rastro de utilização e do número de vezes que o preditor a ser usado é trocado pelo seletor ao longo do tempo.

A partir dos resultados obtidos, percebe-se que no mínimo 3 diferentes preditores são sempre utilizados. Além disso, é possível observar que o número de vezes que o seletor opta por trocar o preditor utilizado varia de dezenas a centenas de vezes durante o período de duração dos dados, e que em cerca de 40% dos rastros de utilização existem mais de 100 trocas de preditor por rastro de utilização. Dado que não existe um único preditor que apresenta o melhor desempenho para as diferentes variações de carga dos rastros de utilização no tempo, os resultados indicam que o desempenho do mecanismo pode ser melhorado se os preditores forem dinamicamente selecionados para uso à medida que o tempo avança e o padrão de utilização muda.



(a)



(b)

Figura 4.7: FDA (a) do número de diferentes preditores escolhidos pelo seletor e (b) da quantidade de vezes que o seletor troca de preditor.

Capítulo 5

Instanciação e Avaliação do Modelo de Provisionamento de Recursos

Utilizando o mesmo modelo de simulação definido na Seção 4.2.1 do capítulo anterior foram realizados três novos experimentos a fim de avaliar e explorar melhorias no arcabouço de provisionamento automático de recursos proposto neste estudo. Primeiramente o modelo de simulação foi exercitado com o intuito de avaliar o desempenho do arcabouço de provisionamento quando considerada a seleção dinâmica de preditores. Em seguida, um novo algoritmo de correção de predições é apresentado e avaliado em relação a outro relevante mecanismo de correção presente na literatura [42]. Esse algoritmo de correção tem por objetivo minimizar o subprovisionamento e por consequência reduzir substancialmente o número de violações de SLO, mas sem comprometer demasiadamente a economia de recursos que pode ser alcançada através do provisionamento automático. Por fim é realizada uma avaliação geral dos efeitos da utilização de correções de predição no arcabouço de provisionamento, tanto na abordagem proativa quanto na reativa, através da avaliação de diversos subconjuntos de preditores, com diferentes graus de correção, em comparação ao caso base sem correção.

No entanto, para que seja possível realizar tais experimentos faz-se necessário definir com precisão alguns parâmetros de funcionamento do controlador, ou seja, estabelecer ações essenciais à instanciação do arcabouço, como: (i) definir os limites e as ações associadas ao comportamento reativo do controlador; (ii) prover implementações para o conjunto de preditores a serem utilizados por cada controlador de camada; (iii) fornecer a implementação de um mecanismo de seleção para dinamicamente selecionar o preditor a ser usado em cada

instante de tempo; (iv) estabelecer os valores apropriados para os parâmetros do arcabouço, tais como a periodicidade de execução dos controladores de camada e dos seletores e a frequência com que as informações monitoradas são coletadas.

Desta forma, além das avaliações efetuadas neste capítulo, é proposto um mecanismo de seleção de preditores, são elaborados preditores que tentam reduzir os erros que conduzem ao subprovisionamento e são estabelecidas ações triviais de provisionamento reativo.

5.1 Seleção Dinâmica de Preditores

O mecanismo de seleção de preditores empregado pelo arcabouço de provisionamento de recursos usa dados do histórico recente de utilização para avaliar o desempenho dos preditores disponíveis e decidir qual dentre os preditores será utilizado nos próximos ciclos de controle. Para tal, são utilizadas, pelo mecanismo de seleção, três semanas de histórico de utilização, da seguinte forma: (i) as duas semanas de dados mais antigos são utilizadas para alimentar os preditores; (ii) a semana mais recente de dados é utilizada como base para avaliar o desempenho dos preditores.

O preditor avaliado que apresenta o melhor desempenho, em termos das métricas descritas no capítulo anterior (Subseção 4.2.1), é selecionado para ser utilizado no módulo de provisionamento de recursos até que uma nova seleção seja realizada, seja em decorrência da periodicidade de execução do seletor ou em reação à ocorrência de uma violação de SLO. Essa avaliação é dada inicialmente pela comparação do desempenho dos preditores em relação à quantidade de violações de SLO e em seguida, em caso de empates, uma nova comparação é realizada em termos da economia de custos obtida pelos diferentes preditores. No entanto, dependendo do modo de atuação do modelo de provisionamento (proativo ou reativo) esta avaliação pode apresentar diferentes níveis de conservadorismo.

Nos experimentos de simulação executados foram realizadas seleções a cada dia e em casos de reação a uma violação de SLO o preditor que levou ao provisionamento mais custoso, mais conservador, foi o preditor utilizado até o fim do período corrente de um dia. Outros valores de periodicidade de seleção foram explorados, no entanto, a partir do teste estatístico de Kruskal-Wallis [25] não foram constatadas diferenças significativas nos resultados obtidos para o número de violações de SLO, quando foi considerado seleções a cada hora, dia e

semana.

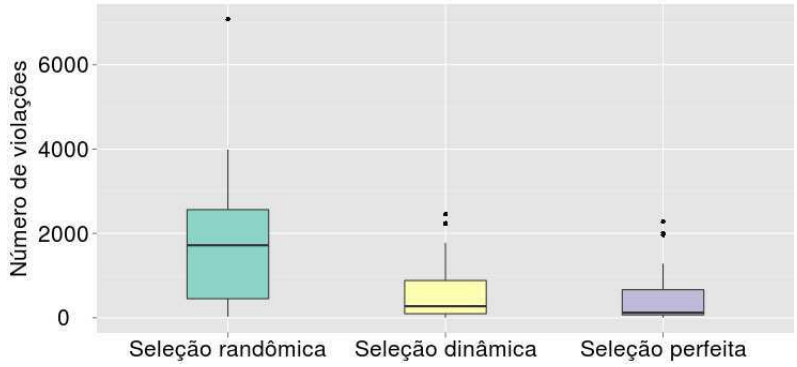
A fim de avaliar o desempenho do mecanismo de seleção, foram executadas simulações usando uma parametrização semelhante a do segundo experimento descrito no capítulo anterior, como pode ser visto na Tabela 5.1. O desempenho do mecanismo é comparado ao de dois outros seletores: o seletor perfeito, não realista, que também encontra-se descrito nesse experimento do capítulo anterior (Subseção 4.2.2) e um seletor randômico, que aleatoriamente seleciona o preditor para ser usado a cada período de controle. Estes dois seletores são vistos como os limites superior e inferior, respectivamente, sobre o desempenho que pode ser obtido pelo mecanismo de seleção apresentado no início desta seção.

Tabela 5.1: Parâmetros utilizados na execução do modelo de simulação para análise do mecanismos de seleção de preditores.

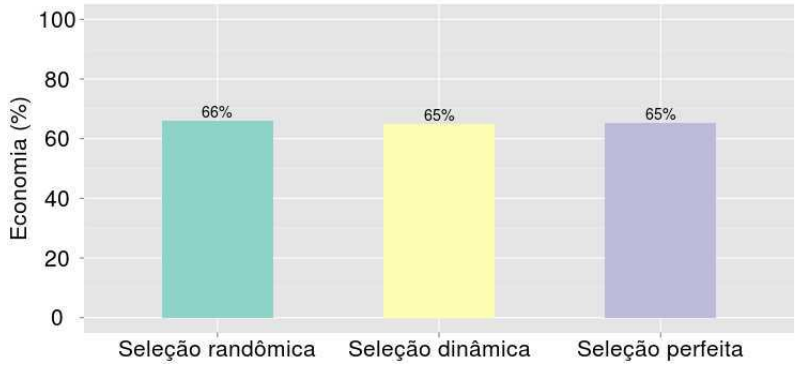
Parâmetro	**Valor**
Quantidade de rastros de utilização	33 arquivos
Duração média dos rastros de utilização	8 meses
Conjunto de preditores	LW, LR, AC, AR, ARIMA e EN
Tamanho do histórico de dados usado por predição	2 semanas
Tamanho do histórico de dados usado por seleção	1 semana
Periodicidade de monitoramento	5 minutos
Periodicidade de predição	5 minutos
Periodicidade de seleção	24 horas
Tempo máximo de ação em modo conservador	24 horas
Periodicidade de controle	5 minutos
Valor de referência de utilização	70%
Valor limite de superprovisionamento	70%
Quantidade mínima de VMs alocadas	1
Número de núcleos por VM	1
Tempo da fatia da tarifação	60 minutos

Nas Figuras 5.1(a) e 5.1(b) são apresentados, respectivamente, os diagramas de caixa

do número de violações e a economia de custos obtida ao simular as 33 cargas de trabalho de longa duração. Como é possível observar, o método de seleção proposto é ligeiramente pior que o perfeito e muito melhor que a seleção randômica em termos do número de violações de SLO e para todos os casos a redução de custos quando comparada com um sistema superprovisionado é em torno de 65%.



(a)



(b)

Figura 5.1: Avaliação do método de seleção dinâmica, de acordo com o número de violações de SLO (a) e à economia de recursos (b), em comparação com a seleção randômica e a seleção perfeita.

O comportamento observado para a quantidade de violações deve-se ao fato de que ao se utilizar um critério mais elaborado para a seleção de preditores eleva-se a qualidade das predições e por consequência reduz-se o número de violações de SLO. Por outro lado, os resultados para a economia de custos de provisionamento necessitam de um estudo mais detalhado para serem devidamente compreendidos. No entanto, à primeira vista, acredita-se que o desempenho semelhante dos três seletores avaliados está relacionado com a distribui-

ção dos erros de seleção no tempo, que podem gerar subestimativas ou super estimativas em diferentes momentos e contribuir para que os custos de provisionamento tendam a convergir em valores próximos.

5.2 Algoritmo de Correção de Predição

Através do experimento e das avaliações realizadas na seção anterior é possível verificar que mesmo se o seletor perfeito for instanciado com o conjunto padrão de 6 preditores considerado nos experimentos executados, as violações de SLO ainda ocorrem em um número bastante elevado. É sabido que violações de SLO podem vir a gerar quebras de SLA, que em geral elevam os custos associados ao provisionamento do serviço oferecido pelo cliente da *Nuvem*. Desta forma, em alguns casos, é importante reduzir, tanto quanto possível, o número de quebras de SLO, mesmo que para tal seja necessário um aumento no custo final de provisionamento.

O objetivo de reduzir quebras de SLO no provisionamento automático pode ser alcançado através da utilização de preditores que atuam de forma mais conservadora. Uma vez que, a partir de análises preliminares dos resultados de predição, foi possível identificar padrões nos erros de predição, foi assumido que a captura desses padrões poderia ser utilizada na correção das predições, com o intuito de gerar preditores mais conservadores. Ou seja, preditores que tentam corrigir suas estimativas com base em erros de predição cometidos anteriormente. Em particular, as abordagens de predição mais conservadoras consideram os erros que conduzem à subestimativa da capacidade necessária e consequentemente à ocorrência de violações de SLO [42].

Desta forma, foi elaborada uma nova abordagem para corrigir predições, de modo a reduzir o número de subestimativas. O processo de correção calcula o histórico de erros de predição e tenta correlacionar a sequência recente de erros de predição com sequências de erros de predição no passado, através de uma função de autocorrelação (ACF, do inglês *Auto-Correlation Function*). O ponto no passado em que a correlação é máxima é utilizado como um ponto central de uma janela de valores de erro de predição. Então, o maior erro negativo dentro desta janela é utilizado para corrigir a predição seguinte, somando-se o erro ao valor estimado. A Figura 5.2 ilustra o algoritmo de correção de predição aqui descrito.

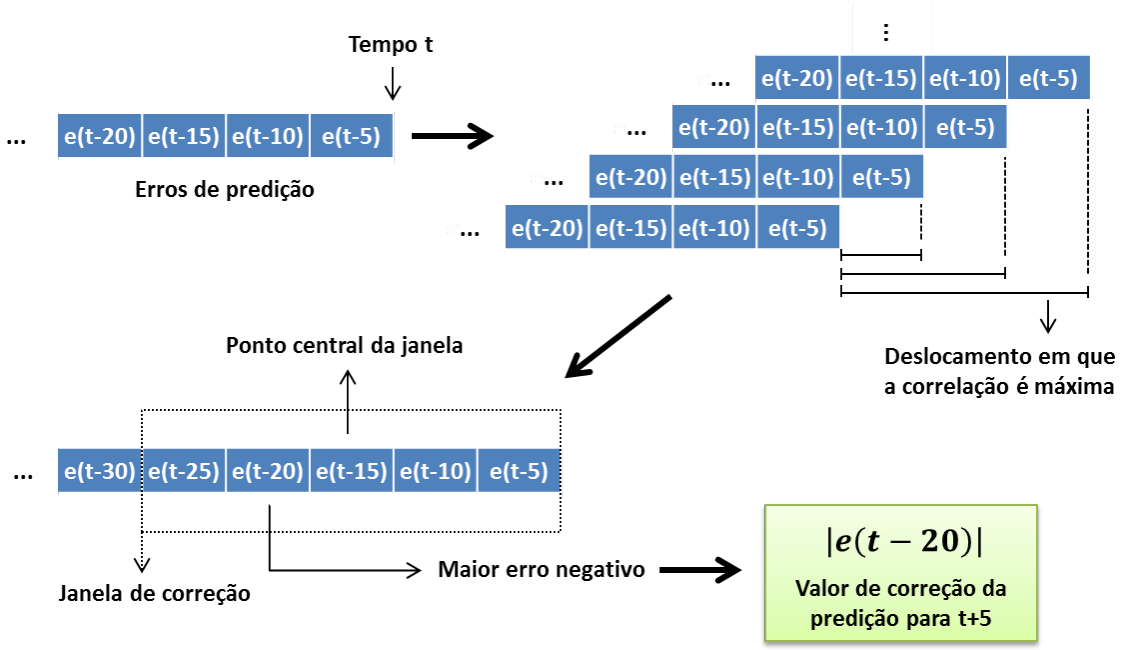


Figura 5.2: Visão ilustrativa do algoritmo de correção de predição baseado na correlação do histórico de erros de predição.

O tamanho da sequência de erros de predição e o tamanho da janela de correção são parâmetros do mecanismo de correção de predição. Neste experimento foi utilizado um tamanho de sequência de erros de 2 semanas do passado e valores para o tamanho da janela que são variados de 4 horas a 2 dias. Os valores do tamanho da janela de correção foram configurados de forma a considerar quantidades iguais de dados em torno do ponto de correlação máxima, por exemplo, para uma janela de 4 horas, ou com 49 intervalos de 5 minutos, considera-se 2 horas antes e 2 horas depois do ponto central da janela.

O experimento de simulação realizado para avaliar o desempenho do mecanismo de correção de predição foi alimentado com as cargas de trabalho de longa duração e utilizou o algoritmo de seleção proposto na seção anterior (Seção 5.1) e uma parametrização semelhante ao do experimento da mesma seção. Contudo, além dos seis preditores originais usados nos outros experimentos, a simulação do algoritmo de correção incluiu versões de todos os preditores com o método de correção proposto acima e instanciado com diferentes tamanhos de janela (referenciados como "ACF-x", onde x é o tamanho da janela em intervalos de 5 minutos).

Além do mais, o experimento de simulação também fez uso de um outro mecanismo de

correção, que foi proposto por Shen et al. [42] (referenciado como "padding"). Esse mecanismo calcula o valor da correção de predição através da combinação de dois métodos. O primeiro método utiliza a transformada rápida de Fourier (FFT, do inglês *Fast Fourier Transform*) para extrair padrões de rajadas de utilização dos últimos x valores de utilização coletados (x corresponde ao tamanho da janela de correção em intervalos de 5 minutos) e em seguida verifica se a densidade de padrões de rajada positivos é superior ao valor de referência de densidade de rajadas. Em caso positivo seleciona-se o a-ésimo percentil do padrão extraído como valor de correção e no caso contrário o b-ésimo percentil é selecionado. O segundo método calcula o valor absoluto da média móvel ponderada (WMA, do inglês *Weighted Moving Average* dos últimos x erros negativos de predição. Ao final, o mecanismo aplica o valor de correção que é o maior entre os valores calculados pelos dois métodos. Embora esse mecanismo tenha sido proposto em um contexto diferente, ele foi utilizado nesta avaliação para comparação, pois não foi encontrado outro método de correção de predição que tenha sido usado no mesmo contexto deste trabalho. A parametrização do modelo de simulação que encontra-se detalhada na Tabela 5.2 foi realizada a partir de valores colhidos do trabalho de referência.

Tabela 5.2: Parâmetros utilizados na execução do modelo de simulação para análise de mecanismos de correção de predição no contexto de provisionamento de recursos.

Parâmetro	**Valor**
Quantidade de rastros de utilização	33 arquivos
Duração média dos rastros de utilização	8 meses
Conjunto de preditores	LW, LR, AC, AR, ARIMA e EN
Tamanho do histórico de dados usado por predição	2 semanas
Tamanho do histórico de dados usado por seleção	1 semana
Periodicidade de monitoramento	5 minutos
Periodicidade de predição	5 minutos
Periodicidade de seleção	24 horas
Tempo máximo de ação em modo conservador	24 horas
Periodicidade de controle	5 minutos

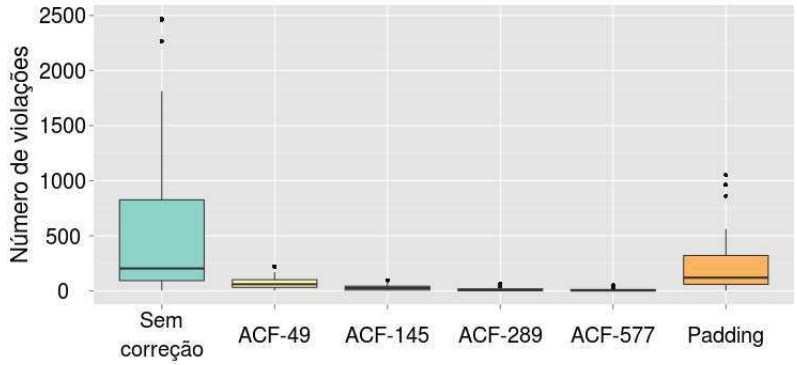
Tabela 5.2 – Continuação

Parâmetro	**Valor**
Valor de referência de utilização	70%
Valor limite de superprovisionamento	70%
Quantidade mínima de VMs alocadas	1
Número de núcleos por VM	1
Tempo da fatia da tarifação	60 minutos
Mecanismos de correção	ACF e Padding
Tamanhos de janela de correção do ACF	49, 145, 289 e 577
Tamanho da janela de correção do Padding	100
Valor de referência de densidade de rajadas	50%
a-ésimo percentil do padrão de rajadas	100%
b-ésimo percentil do padrão de rajadas	80%

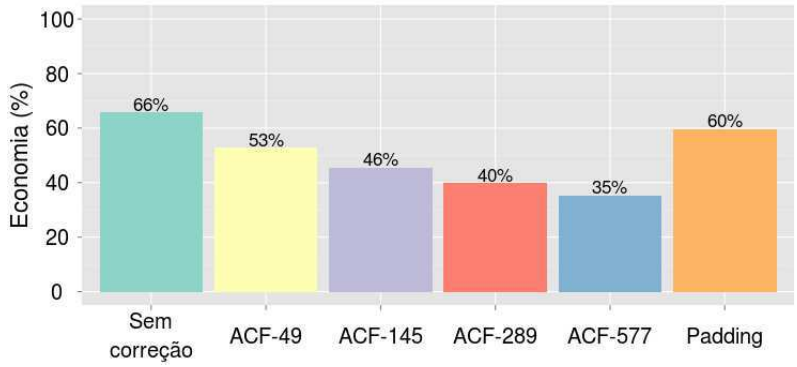
As Figuras 5.3(a) e 5.3(b) mostram, respectivamente, o diagrama de caixa do número de violações e o gráfico de barras das economias de custo obtidas em comparação ao cenário superdimensionado nas simulações realizadas.

Através destes resultados é possível observar que todos os métodos de correção são capazes de reduzir substancialmente o número de violações, sendo o método "ACF-x" mais eficiente para a configuração considerada. Possivelmente, o método "padding" não obteve bons resultados por ter sido proposto para um contexto diferente ao do experimento realizado.

Os resultados apresentados pelo método "ACF-x" mostram que a quantidade de violações de SLO é inversamente proporcional ao tamanho da janela de correção, ou seja, quanto maior a janela de correção menor é o número de violações de SLO. Esse comportamento é esperado, pois ao se aumentar o tamanho da janela de correção elevam-se as chances de que maiores erros de predição sejam encontrados dentro da janela, o que pode proporcionar maiores correções, levando a predições mais conservadoras e consequentemente a uma redução na ocorrência de violações de SLO. Ademais, é possível observar que a redução no número de violações ocorre às custas de um aumento no custo final de provisionamento, proveniente do conservadorismo implícito ao processo de correção de predição.



(a)



(b)

Figura 5.3: Comparação dos dois métodos de correção de predição com relação (a) ao número de violações e (b) à economia de recursos.

5.3 Efeitos da Correção de Predição

As correções de predição abordadas na seção anterior podem ser encaradas como novos tipos de preditores. Ao fazer isso, é possível considerar as correções de predição no arcabouço proposto, simplesmente adicionando ao conjunto pré-definido de preditores disponíveis alguns preditores que fazem uso de procedimentos de correção. Logo, resultados diferentes podem ser obtidos quando se é mais ou menos conservador na configuração da instância do arcabouço de provisionamento automático.

Foi realizado um experimento de simulação a fim de analisar como o nível de conservadorismo na configuração do arcabouço proposto pode impactar na redução do número de violações de SLO e na economia do custo de provisionamento. O experimento executado realiza o provisionamento automático através do seletor apresentado na Seção 5.1 e utilizando diferentes variações do conjunto de preditores. O modelo de provisionamento efetua

o provisionamento proativo diretamente através do emprego das predições, enquanto que o provisionamento reativo é disparado sempre que uma violação de SLO ocorre.

No provisionamento reativo, o arcabouço foi configurado de forma que o preditor que apresenta os maiores custos, e consequentemente realiza as maiores correções, seja utilizado até que o período corrente de 24 horas termine e uma nova seleção seja realizada pelo mecanismo de provisionamento automático proativo. Essa configuração permite que reações às violações de SLO sejam tratadas de modo conservador, enquanto que as ações proativas são realizadas de forma não conservadora. No entanto, os modos de operação do mecanismo de provisionamento são ortogonais ao conservadorismo da configuração do arcabouço.

Além do mais, a decisão do emprego de uma abordagem reativa configurada de forma conservadora foi fundamentada em um micro experimento de simulação que compara o desempenho do arcabouço quando configurado seguindo duas diferentes abordagens de provisionamento: uma apenas proativa e não conservadora e outra híbrida, com o modo de operação reativo configurado de forma conservadora e o proativo atuando de forma não conservadora. Os resultados mostram que, para os 33 rastros de utilização e uma configuração que considera um conjunto com todos os preditores e suas devidas variações com correção ACF, o número total de violações de SLO é cerca de 10% menor quando se utiliza a abordagem híbrida.

Seja P qualquer um dos preditores do conjunto {AC, LR, AR, ARIMA, LW, EN}, e P-x um preditor que utiliza o preditor P e realiza correções através do método "ACF-x" com uma janela de tamanho x, foram utilizados os seguintes conjuntos de preditores na simulação dos efeitos da correção de predição: C_1={P, P-49}, C_2={P, P-49, P-145}, C_3={P, P-49, P-145, P-289}, C_4={P, P-49, P-145, P-289, P-577}, C_5={P-49, P-145, P-289, P-577}, C_6={P-145, P-289, P-577}, C_7={P-289, P-577}, C_8={P-577}. A Tabela 5.3 descreve em detalhes a parametrização utilizada no experimento de simulação.

Tabela 5.3: Parâmetros utilizados na execução do modelo de simulação para análise de eficiência do mecanismo de correção de predição no contexto de provisionamento de recursos.

Parâmetro	**Valor**
Quantidade de rastros de utilização	33 arquivos

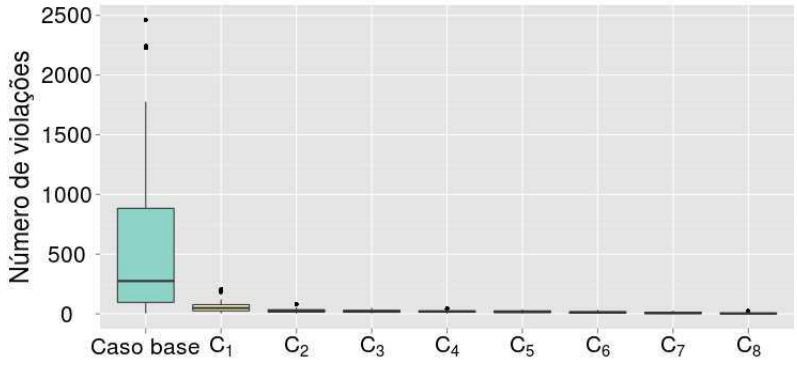
Tabela 5.3 – Continuação

Parâmetro	**Valor**
Duração média dos rastros de utilização	8 meses
Conjuntos de preditores	C_1, C_2, C_3, C_4, C_5, C_6, C_7 e C_8
Tamanho do histórico de dados usado por predição	2 semanas
Tamanho do histórico de dados usado por seleção	1 semana
Periodicidade de monitoramento	5 minutos
Periodicidade de predição	5 minutos
Periodicidade de seleção	24 horas
Tempo máximo de ação em modo conservador	24 horas
Periodicidade de controle	5 minutos
Valor de referência de utilização	70%
Valor limite de superprovisionamento	70%
Quantidade mínima de VMs alocadas	1
Número de núcleos por VM	1
Tempo da fatia da tarifação	60 minutos

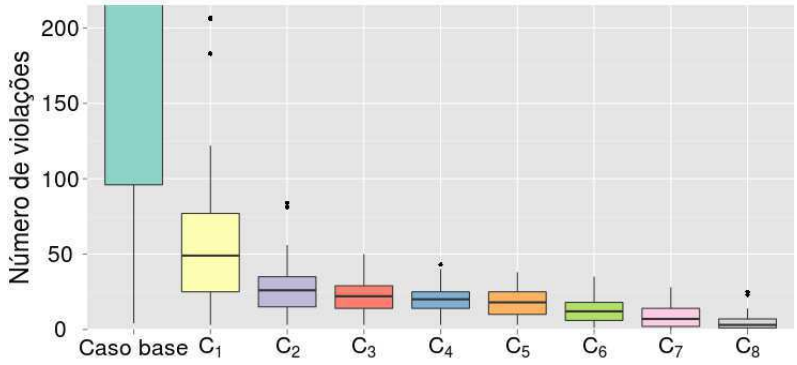
O modelo de simulação foi exercitado com as 33 cargas de trabalho de longa duração para cada um dos subconjuntos de preditores. Também foi avaliado o desempenho da configuração do modelo que não realiza correção de erros de predição, para servir como caso base para comparação. Os resultados do desempenho das diferentes configurações com relação ao número de violações podem ser observados nas Figuras 5.4(a) e 5.4 e com relação à economia de custo de provisionamento na Figura 5.5.

Através dos resultados de desempenho observa-se que as configurações que contêm P-577 (C_4, C_5, C_6, C_7 e C_8), configurações mais conservadoras, são aquelas que proporcionam as maiores reduções no número de violações. Esse comportamento é explicado pelo mesmo princípio da relação entre o tamanho da janela de correção e a quantidade de violações. Ou seja, uma configuração do arcabouço com preditores que utilizam grandes janelas de correção permite a seleção desses preditores e possibilita, por consequência, uma redução no número de violações de SLO.

Essas configurações conservadoras, quando comparado com o caso base, produzem um



(a)



(b)

Figura 5.4: Comparação de diferentes configurações do modelo de provisionamento com relação ao número de violações, apresentada em diferentes escalas.

número médio de violações que é entre 31 e 122 vezes menor. Isto vem com uma redução na economia média dos custos alcançados que varia de 29% a 43%, quando comparado com a redução na economia média dos custos obtidos no cenário do caso base, mas que ainda rende substanciais economias de custo, variando de 37% a 46%. Por outro lado, as outras configurações apresentam menores reduções do número de violações, variando de 10 a 27 vezes, com reduções na economia média dos custos alcançados de não mais que 28%.

Quando considerado apenas o cenário em que são usados todos os preditores com todas as janelas de correção (C_4) a probabilidade média de ocorrência de uma violação de SLO é de aproximadamente 0.03% para uma economia média de 46%. Onde, no pior caso, quando há uma maior frequência de violações de SLO, a probabilidade de ocorrer uma violação não é maior que 0.06%. Por outro lado, o cenário mais conservador (C_8), apresenta os melhores resultados quanto ao número de violações de SLO. Os resultados deste cenário mostram que

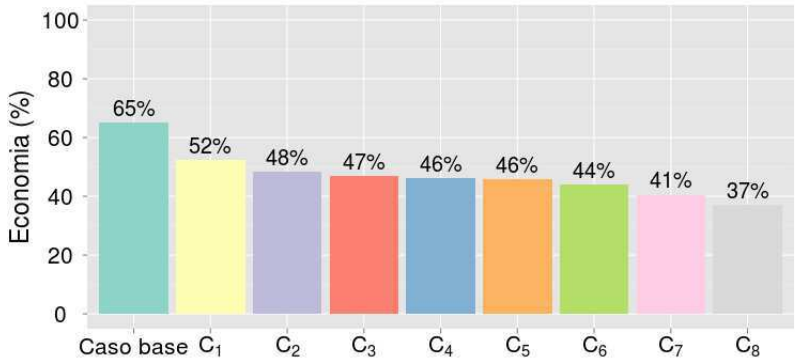


Figura 5.5: Comparação de diferentes configurações do modelo de provisionamento com relação à redução de custos.

uma configuração mais conservadora do arcabouço é capaz de reduzir a probabilidade média de ocorrência de violações para algo um pouco inferior a $0,008\%$, com uma economia média de custos de 37%, quando comparado a um sistema superprovisionado. Além disto, para esse caso a probabilidade de ocorrer violações é sempre mantida abaixo de $0,036\%$.

Capítulo 6

Conclusão

Este capítulo apresenta, na Seção 6.1, uma sumariação do que foi realizado neste trabalho e descrito neste documento. Na Seção 6.2, são feitas as considerações acerca dos resultados obtidos a partir do arcabouço proposto e de sua instanciação. Por fim, na Seção 6.3, são descritas potenciais limitações apresentadas por este trabalho de dissertação e enumerados possíveis trabalhos futuros que podem vir a sanar tais fraquezas.

6.1 Sumário

Neste trabalho de dissertação foi proposto um arcabouço de provisionamento automático de recursos a curto prazo para infraestruturas de *Computação na Nuvem*, especificamente ambientes de IaaS, que não é intrusivo à aplicação que está sendo provida. O arcabouço proposto funciona segundo uma abordagem híbrida, que realiza dinamicamente ações de provisionamento tanto de forma proativa quanto de forma reativa, por meio do uso de múltiplos preditores. Através desse arcabouço é possível desenvolver e instanciar novas soluções de provisionamento automático de recursos devido a sua flexibilidade e adaptatividade, visto que ele permite a configuração dinâmica e automática dos módulos de monitoramento e predição.

Complementarmente ao arcabouço de provisionamento, também foi apresentado um mecanismo para correção de predições baseado no histórico de predições e de dados de utilização visando minimizar a ocorrência de subestimativas no planejamento de capacidade, que podem vir a reduzir o número de violações de SLO sem impactar significantemente no custo

de provisionamento.

O arcabouço proposto foi implementado na linguagem R e as métricas de avaliação de desempenho foram definidas como sendo o número de violações de SLO e a economia de recursos obtida em relação a um cenário superprovido. A implementação do arcabouço foi instanciada e simulada a partir de dados de utilização de recursos de aplicações em produção de clientes da HP.

Por fim, a partir dos resultados de simulação das instâncias do arcabouço foram realizadas um série de avaliações de desempenho. Visto que não se conhece solução de provisionamento automático de recursos a curto prazo para ambientes de IaaS, tanto no mercado como na literatura, que seja totalmente compatível como o modelo de provisionamento empregado pelo arcabouço, estas avaliações foram realizadas em comparação a cenários perfeitos e não realistas ou a casos simplistas, que serviram de contexto base de referência. Desta forma, as avaliações serviram para analisar o arcabouço segundo três aspectos principais: (1) a eficácia do método de seleção de preditores em contrapartida a um seletor perfeito e outro aleatório; (2) o desempenho do mecanismo de correção de predições em comparação a outro significativo mecanismo presente na literatura; (3) os efeitos da utilização do mecanismo de correção em relação ao cenário base sem correção.

Uma versão resumida deste trabalho de dissertação será publicada nos anais do 13° Simpósio Internacional sobre *Cluster*, *Cloud* e *Grid Computing* (CCGrid 2013) [31].

6.2 Conclusões

A partir da diversidade de experimentos de simulação realizados foi possível constatar que o arcabouço de provisionamento automático de recursos proposto é flexível e adaptável, visto que foram conseguidas diferentes instanciações do modelo através de mudanças triviais na configuração do arcabouço. Da mesma forma, a eficiência do arcabouço e do mecanismo de provisionamento por ele utilizado ficou demonstrada por meio das avaliações realizadas a partir das simulações executadas.

Ao se observar os resultados do último experimento de simulação realizado (vide Seção 5.3 do capítulo anterior) é possível verificar que quanto mais conservadora é a configuração do arcabouço menor é o número de violações de SLO verificadas e menor é a

economia de recursos. Ou seja, ao se utilizar configurações do arcabouço com preditores mais conservadores aumenta-se a probabilidade de um preditor desse tipo ser selecionado e indiretamente reduz-se a chance de ocorrência de violações de SLO. Dado que, preditores com maiores janelas de correção estão mais propensos a realizar maiores correções e consequentemente gerar maiores valores de estimativa de demanda.

Os resultados do cenário mais conservador desse experimento, que considera apenas preditores com a maior janela de correção, mostram que com uma configuração adequada do arcabouço de provisionamento é possível obter uma economia média de custos de 37% com uma probabilidade média de ocorrência de violações em torno de $0,008\%$, comparando-se a um sistema superprovisionado. Além do mais, para este cenário conservador a probabilidade de ocorrer violações é no pior caso mantida abaixo de $0,036\%$.

Portanto, através dos experimentos de simulação foi possível comprovar que a utilização do arcabouço proposto pode trazer um nível significativo de economia de custos sujeito a uma fração desprezível de violações de SLO quando comparado a um cenário superprovido. Além do mais, os experimentos evidenciam a eficiência do mecanismo de seleção proposto e do algoritmo de correção de predição desenvolvido neste trabalho.

6.3 Limitações e Trabalhos Futuros

O arcabouço de provisionamento automático de recursos proposto nesse trabalho apresenta algumas limitações que são decorrentes tando de simplificações realizadas durante sua construção e simulação como de fraquezas de validação do modelo conceitual e de simulação. Algumas destas simplificações consistem em constantes do modelo que não foram devidamente estimadas por meio de experimentos em ambientes reais de IaaS, como:

- O tempo despendido para que uma máquina virtual esteja ativa e pronta para ser utilizada, dado que a inicialização da máquina e a implantação da aplicação podem requerer um tempo considerável. Por outro lado, caso esse tempo seja mínimo, o modelo de provisionamento puramente reativo passa a ser uma solução viável;

- O tempo gasto para que uma máquina virtual seja desativada, considerando a interferência do balanceamento da carga entre as máquinas virtuais que continuam alocadas

para a aplicação. Esse fator também influencia a viabilidade de abordagens reativas, dado que reduções consideráveis desse tempo podem tornar o provisionamento reativo uma abordagem aceitável;

- O possível erro propagado durante o balanceamento de carga, por exemplo, quando uma nova máquina virtual é alocada a carga atual da aplicação não é igualmente dividida entre todas as máquinas virtuais alocadas, o sistema de balanceamento necessita de um tempo para realizar a operação e para estabilizar a carga entre as máquinas virtuais;

- A periodicidade de monitoramento, pois talvez seja necessário monitorar os recursos alocados à aplicação em períodos menores de tempo, visto que periodicidades menores podem permitir que os dados coletados representem de forma mais confiável os padrões de carga apresentados pela aplicação.

- O tamanho do histórico de dados necessários para o provisionamento. Devido ao tamanho do histórico considerado na avaliação do arcabouço é possível haver restrições do tipo da aplicação que pode ser provisionada pelo modelo. Ou seja, em geral, o arcabouço só pode ser empregado para o provisionamento de aplicações de longa duração ou para aplicações de curta duração que são executadas com uma frequência considerável e que possuem características de carga semelhantes entre si. Uma vez que se acredita que, para aplicações de curta duração com esse perfil, é possível usar os dados de execução das aplicações como histórico de utilização para o provisionamento de uma nova aplicação do mesmo tipo.

Todos essas características são fatores que devem ter sua sensibilidade analisada e experimentada para que sejam, de fato, confiáveis e representativos do comportamento do sistema real. Além disso, outras características, como a utilização de um mesmo tipo de recurso virtual por camada de aplicação, o fato de considerar-se a existência de um gerente de recursos do ambiente de IaaS e que esses recursos são ilimitados e a realização de monitoramento e agrupamento de dados de forma a não interferir no funcionamento do sistema, são também limitações inerentes ao trabalho aqui apresentado. No entanto, estas limitações podem servir de base para possíveis expansões do trabalho, dado que tangenciam um leque de complexidades e lacunas relativas ao gerenciamento de infraestruturas de *Computação na Nuvem*.

Por outro lado, as fraquezas na validação dos modelos conceitual e de simulação são decorrentes da ausência de um sistema real que apresente as mesmas características que o arcabouço proposto e das aplicações que geraram os dados de carga utilizados para alimentar as simulações. Apesar de existir uma validação aparente, que avalia se o comportamento do modelo apresenta uma lógica razoável em comparação a um sistema real ou a um problema definido [36], pouco pôde ser realizado para validar o modelo conceitual e o modelo de simulação.

Desta forma, um possível trabalho futuro, que viria para ajudar a sanar algumas destas limitações, consiste na implementação do arcabouço em uma linguagem de programação que permita com maior facilidade o uso de ferramentas de desenvolvimento dirigido por testes (TDD, do inglês *Test-Driven Development*) e de testes automáticos, o que contribuiria para a verificação da implementação do modelo conceitual. Além disso, a implantação da implementação do arcabouço em um ambiente real de IaaS permitiria que fosse realizada uma validação operacional do modelo de simulação, através da comparação do comportamento do sistema real e do sistema simulado. Visto que ambos os sistemas podem ser estimulados pelos mesmos dados de utilização de recursos.

Referências Bibliográficas

[1] D. Agrawal, A. Abbadi, S. Das, and A.J. Elmore. Database scalability, elasticity, and autonomy in the cloud. In *Database Systems for Advanced Applications*, volume 6587 of *Lecture Notes in Computer Science*, pages 2–15. Springer Berlin Heidelberg, 2011.

[2] J. Almeida, V. Almeida, Ardagna D., I. Cunha, C. Francalanci, and M. Trubian. Joint admission control and resource allocation in virtualized servers. *Journal of Parallel and Distributed Computing*, 70(4):344 – 362, 2010.

[3] Amazon. Amazon web services. `http://aws.amazon.com`. Online; Nov., 2012.

[4] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. *ACM Communications*, 53(4):50–58, Apr. 2010.

[5] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica, et al. Above the clouds: A berkeley view of cloud computing. *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28*, 2009.

[6] N. Bonvin, T.G. Papaioannou, and K. Aberer. Autonomic sla-driven provisioning for cloud applications. In *Cluster, Cloud and Grid Computing, 11th IEEE/ACM International Symposium on*, CCGRID '11, pages 434 –443, Newport Beach, CA, USA, May. 2011.

[7] R. Buyya, Chee Shin Yeo, and S. Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In *High Performance Computing and Communications, 10th IEEE International Conference on*, HPCC '08, pages 5 –13, Dalian, China, Sep. 2008.

[8] N.M. Calcavecchia, B.A. Caprarescu, E. Di Nitto, D.J. Dubois, and D. Petcu. Depas: a decentralized probabilistic algorithm for auto-scaling. *Computing*, 94:701–730, 2012.

[9] R.N. Calheiros, C. Vecchiola, D. Karunamoorthy, and R. Buyya. The aneka platform and qos-driven resource provisioning for elastic applications on hybrid clouds. *Future Generation Computer Systems*, 28(6):861 – 870, 2012.

[10] S. Casolari and M. Colajanni. Short-term prediction models for server management in internet-based contexts. *Decision Support Systems*, 48(1):212 – 223, 2009.

[11] John Chambers. The R project for statistical computing. `http://www.r-project.org/`. Online; Nov., 2012.

[12] Amazon CloudWatch. Amazon cloudwatch metrics. `http://docs.amazonwebservices.com/AmazonCloudWatch/latest/DeveloperGuide/CW_Support_For_AWS.html`. Online; Dec., 2012.

[13] J.D. Cryer and K. Chan. Trends. In *Time Series Analysis*, Springer Texts in Statistics, pages 27–54. Springer New York, 2008.

[14] W. Dawoud, I. Takouna, and C. Meinel. Elastic vm for cloud resources provisioning optimization. In *Advances in Computing and Communications*, volume 190 of *Communications in Computer and Information Science*, pages 431–445. Springer Berlin Heidelberg, 2011.

[15] J.O. Fito, I. Goiri, and J. Guitart. Sla-driven elastic cloud hosting provider. In *Parallel, Distributed and Network-Based Processing, 18th Euromicro International Conference on*, PDP '10, pages 111 –118, Pisa, Italy, Feb. 2010.

[16] G. Galante and L.C.E. de Bona. A survey on cloud computing elasticity. In *Utility and Cloud Computing, 5th IEEE/ACM International Conference on*, UCC '12, pages 263 –270, Chicago, IL, USA, Nov. 2012.

[17] Gartner. Gartner says worldwide cloud services market to surpass $109 billion in 2012. `http://www.gartner.com/it/page.jsp?id=2163616`. Online; Dec., 2012.

[18] GoGrid. Gogrid cloud hostin. `http://www.gogrid.com`. Online; Nov., 2012.

[19] Z. Gong, X. Gu, and J. Wilkes. Press: Predictive elastic resource scaling for cloud systems. In *Network and Service Management, 6th International Conference on*, CNSM '10, pages 9 –16, Ontario, Canada, Oct. 2010.

[20] M. Hajjat, X. Sun, Y.E. Sung, D. Maltz, S. Rao, K. Sripanidkulchai, and M. Tawarmalani. Cloudward bound: planning for beneficial migration of enterprise applications to the cloud. *ACM SIGCOMM Computer Communication Review*, 40(4):243–254, Aug. 2010.

[21] J.D. Hamilton. *Time series analysis*, volume 2. Cambridge Univ Press, 1994.

[22] R. Han, L. Guo, M.M. Ghanem, and Y. Guo. Lightweight resource scaling for cloud applications. In *Cluster, Cloud and Grid Computing, 12th IEEE/ACM International Symposium on*, CCGRID '12, pages 644 –651, Ottawa, Canada, May. 2012.

[23] J. Hellerstein, S. Parekh, Y. Diao, and D.M. Tilbury. *Feedback control of computing systems*. Wiley-IEEE Press, 2004.

[24] Y. Jiang, C. Perng, T. Li, and R. Chang. Self-adaptive cloud capacity planning. In *Services Computing, 9th IEEE International Conference on*, SCC '12, pages 73 –80, Honolulu, HI, USA, Jun. 2012.

[25] W.H. Kruskal and W.A. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621, 1952.

[26] F. Leymann and D. Fritsch. Cloud computing: The next revolution in it. *Proceedings of the 52th Photogrammetric Week*, pages 3–12, 2009.

[27] H.C. Lim, S Babu, J.S. Chase, and S.S. Parekh. Automated control in cloud computing: challenges and opportunities. In *Automated control for datacenters and clouds, 1st Workshop on*, ACDC '09, pages 13–18, Barcelona, Spain, Jun. 2009.

[28] P.D. Maciel, F. Brasileiro, R. Lopes, M. Carvalho, and M. Mowbray. Evaluating the impact of planning long-term contracts on the management of a hybrid it infrastructure.

In *Integrated Network Management, 12th IFIP/IEEE International Symposium on*, IM '11, pages 89 –96, Dublin, Ireland, May. 2011.

[29] P. Marshall, K. Keahey, and T. Freeman. Elastic site: Using clouds to elastically extend site resources. In *Cluster, Cloud and Grid Computing, 10th IEEE/ACM International Conference on*, CCGRID '10, pages 43–52, Melbourne, Victoria, Australia, May. 2010.

[30] S. Meng, L. Liu, and V. Soundararajan. Tide: achieving self-scaling in virtualized datacenter management middleware. In *Industrial track, 11th International Middleware Conference on*, Middleware '10, pages 17–22, Bangalore, India, Nov.-Dec. 2010.

[31] F. Morais, F. Brasileiro, R. Lopes, R. Araújo, W. Satterfield, and L. Rosa. Autoflex: Service agnostic auto-scaling framework for iaas deployment models. In *Cluster, Cloud and Grid Computing, 13th IEEE/ACM International Symposium on*, CCGRID '13, Delft, Netherlands, May. 2013. (to appear).

[32] Rackspace. Rackspace cloudservers. `http://www.rackspace.com`. Online; Nov., 2012.

[33] RightScale. Right scale cloud management. `http://www.rightscale.com`. Online; Dec., 2012.

[34] N. Roy, A. Dubey, and A. Gokhale. Efficient autoscaling in the cloud using predictive models for workload forecasting. In *Cloud Computing, 4th IEEE International Conference on*, CLOUD '11, pages 500 –507, Washington DC, USA, Jul. 2011.

[35] Y. Sakamoto and G. Kitagawa. *Akaike information criterion statistics*. Kluwer Academic Publishers, 1987.

[36] R.G. Sargent. Verification and validation of simulation models. In *Winter simulation, 37th Conference on*, WSC '05, pages 130–143, Orlando, FL, USA, Dec. 2005.

[37] Scalr. Scalr cloud management. `http://scalr.net`. Online; Dec., 2012.

[38] Amazon Web Services. Amazon cloudwatch documentation. `http://aws.amazon.com/pt/documentation/cloudwatch/`. Online; Dec., 2012.

[39] Amazon Web Services. Amazon elastic compute cloud api reference. `http://docs.amazonwebservices.com/AWSEC2/latest/APIReference/Welcome.html`. Online; Nov., 2012.

[40] Y. Seung, T. Lam, L.E. Li, and T. Woo. Cloudflex: Seamless scaling of enterprise applications into the cloud. In *Computer Communications, 30th IEEE International Conference on*, INFOCON '11, pages 211 –215, Shanghai, China, Apr. 2011.

[41] U. Sharma, P. Shenoy, S. Sahu, and A. Shaikh. A cost-aware elasticity provisioning system for the cloud. In *Distributed Computing Systems, 31st International Conference on*, ICDCS '11, pages 559 –570, Minneapolis, MN, USA, Jun. 2011.

[42] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes. Cloudscale: elastic resource scaling for multi-tenant cloud systems. In *Cloud Computing, 2nd ACM Symposium on*, SOCC '11, pages 5:1–5:14, Cascais, Portugal, Oct. 2011.

[43] B. Sotomayor, R.S. Montero, I.M. Llorente, and I. Foster. Virtual infrastructure management in private and hybrid clouds. *Internet Computing, IEEE*, 13(5):14 –22, Sep.-Oct. 2009.

[44] I. Sriram and A. Khajeh-Hosseini. Research agenda in cloud technologies. *Computing Research Repository*, abs/1001.3259, 2010.

[45] K. Stanoevska-Slabeva and T. Wozniak. Cloud basics - an introduction to cloud computing. In *Grid and Cloud Computing*, pages 47–61. Springer Berlin Heidelberg, 2010.

[46] B. Urgaonkar, P. Shenoy, A. Chandra, P. Goyal, and T. Wood. Agile dynamic provisioning of multi-tier internet applications. *ACM Transactions on Autonomous and Adaptive Systems*, 3(1):1:1–1:39, Mar. 2008.

[47] N. Vasić, D. Novaković, S. Miučin, D. Kostić, and R. Bianchini. Dejavu: accelerating resource allocation in virtualized environments. In *Architectural Support for Programming Languages and Operating Systems, 17th International Conference on*, ASPLOS '12, pages 423–436, London, England, UK, Mar. 2012.

[48] S. Vijayakumar, Q. Zhu, and G. Agrawal. Dynamic resource provisioning for data streaming applications in a cloud environment. In *Cloud Computing Technology and Science, 2nd IEEE International Conference on*, CloudCom '10, pages 441 –448, Indianapolis, IN, USA, Dec. 2010.

[49] S. Zardari and R. Bahsoon. Cloud adoption: a goal-oriented requirements engineering approach. In *Software Engineering for Cloud Computing, 2nd International Workshop on*, SECLOUD '11, pages 29–35, Waikiki, Honolulu , HI, USA, May. 2011.

Apêndice A

Distribuição das Cargas de Trabalho

Neste capítulo é apresentada, de forma gráfica, uma visão geral da distribuição de utilização de CPU ao longo do tempo, considerando os dois conjuntos de dados de utilização de recursos: 265 rastros de utilização com duração de um mês e 33 rastros de utilização com duração média de 8 meses.

A.1 Cargas de Trabalho de Curta Duração

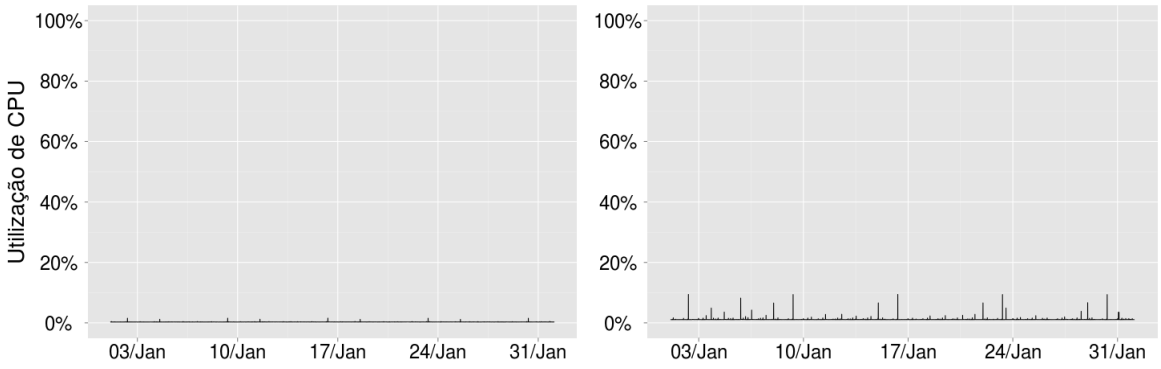


Figura A.1: Rastros 1 e 2 de curta duração.

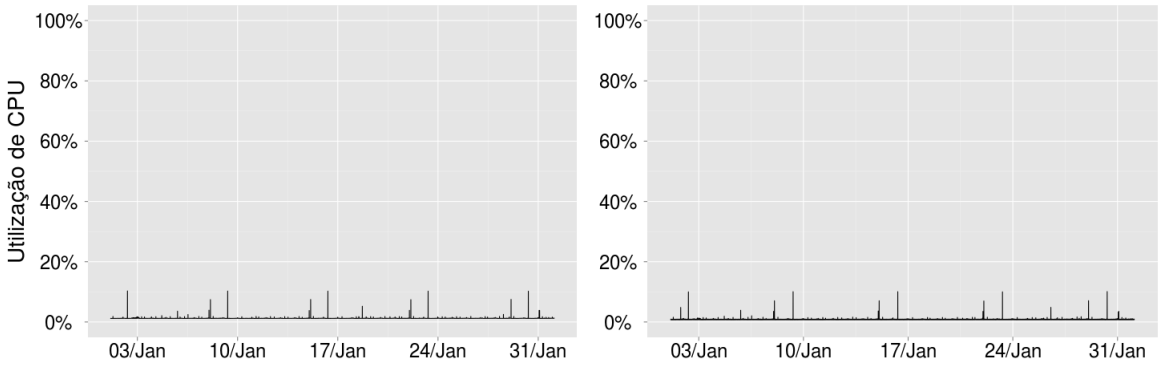


Figura A.2: Rastros 3 e 4 de curta duração.

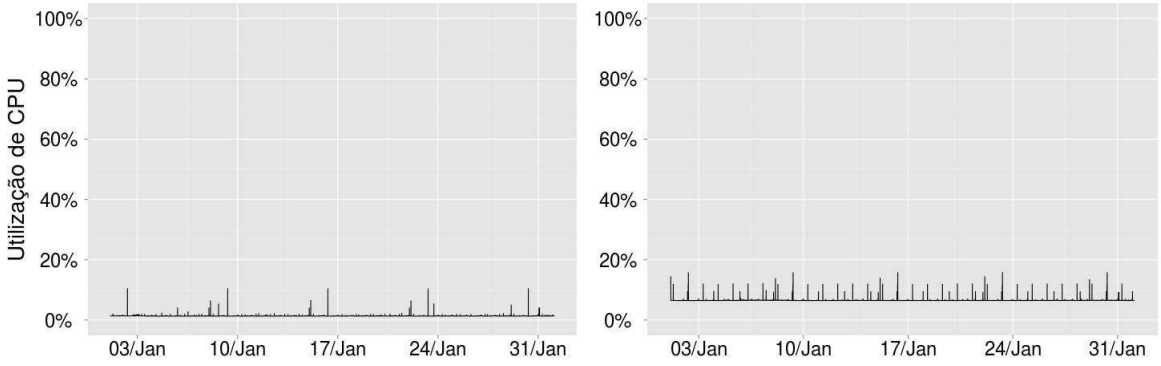


Figura A.3: Rastros 5 e 6 de curta duração.

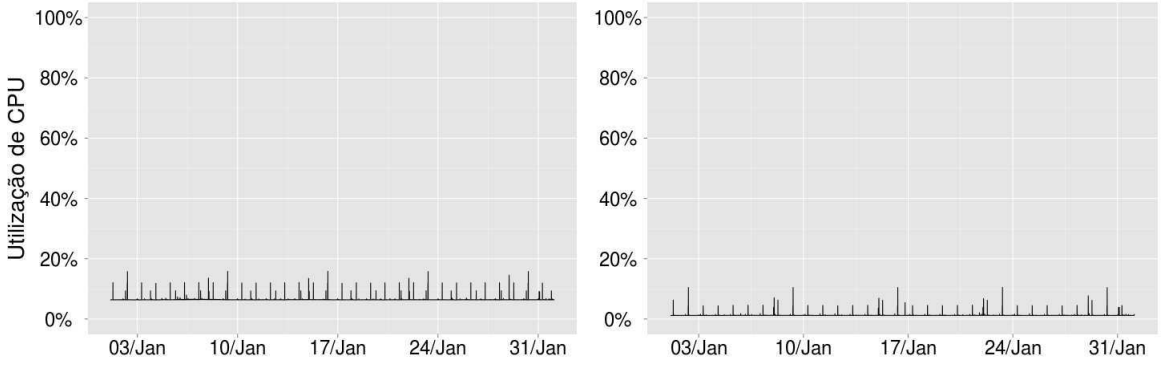


Figura A.4: Rastros 7 e 8 de curta duração.

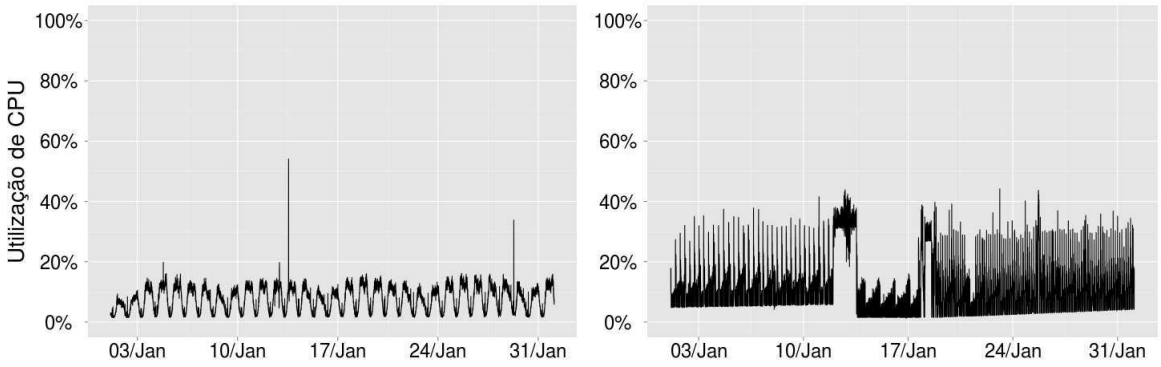


Figura A.5: Rastros 9 e 10 de curta duração.

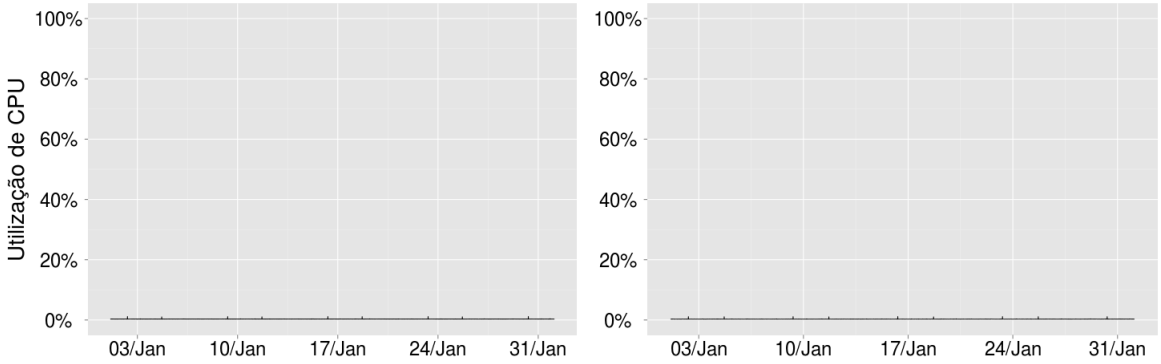


Figura A.6: Rastros 11 e 12 de curta duração.

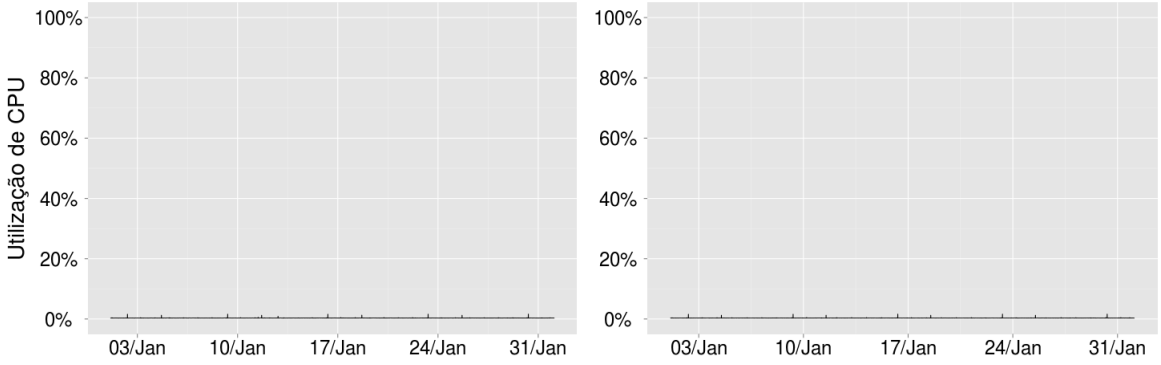


Figura A.7: Rastros 13 e 14 de curta duração.

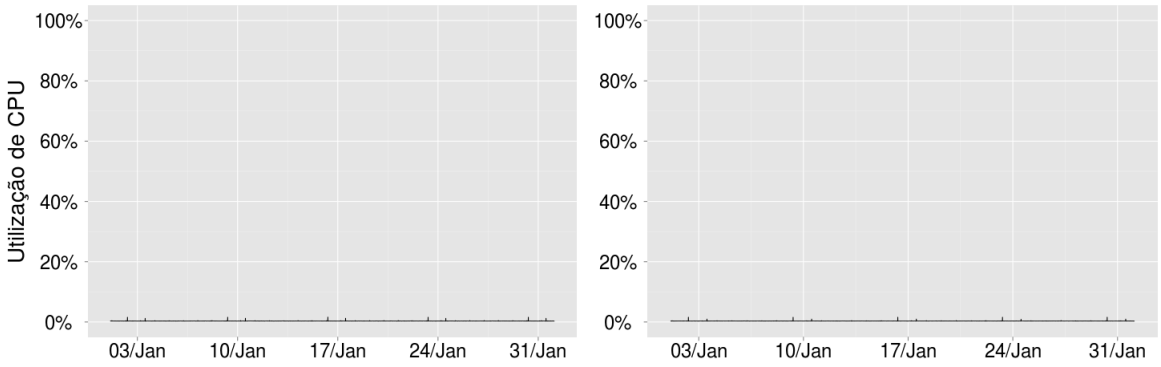


Figura A.8: Rastros 15 e 16 de curta duração.

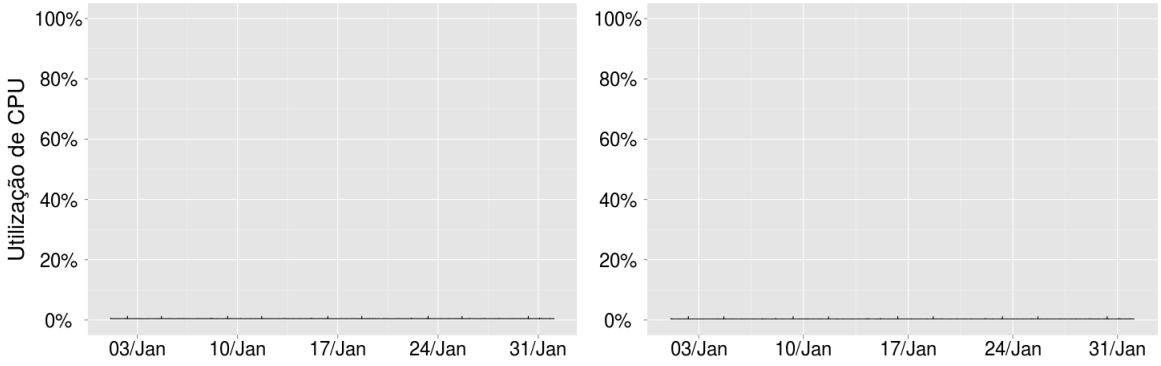


Figura A.9: Rastros 17 e 18 de curta duração.

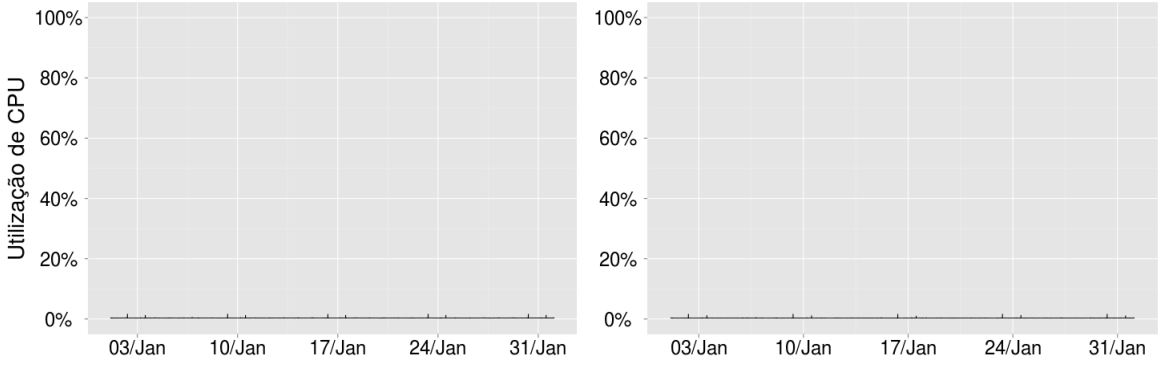


Figura A.10: Rastros 19 e 20 de curta duração.

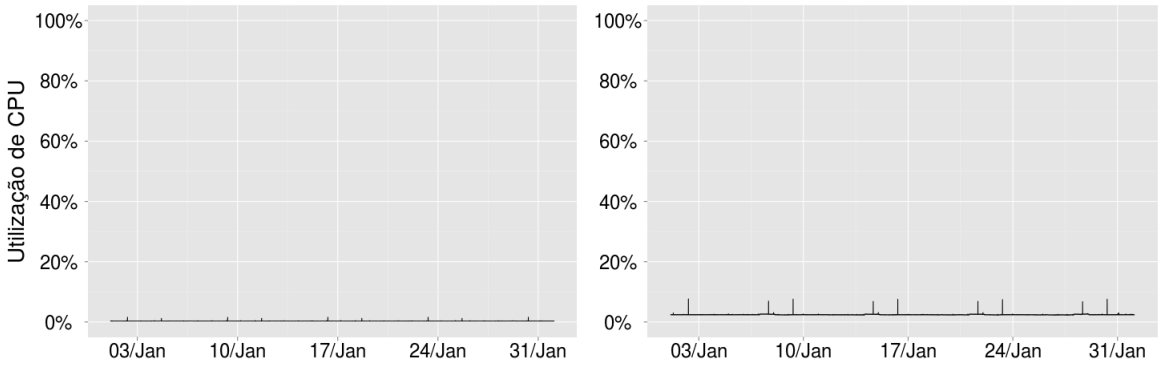


Figura A.11: Rastros 21 e 22 de curta duração.

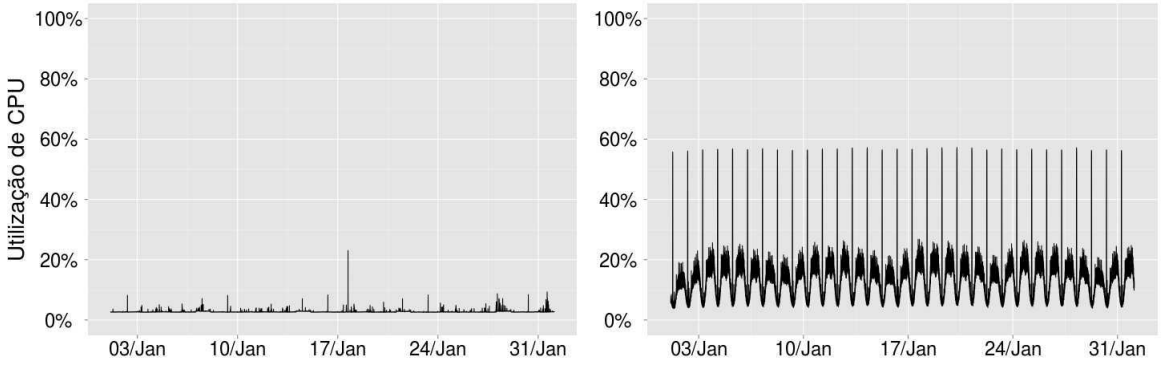


Figura A.12: Rastros 23 e 24 de curta duração.

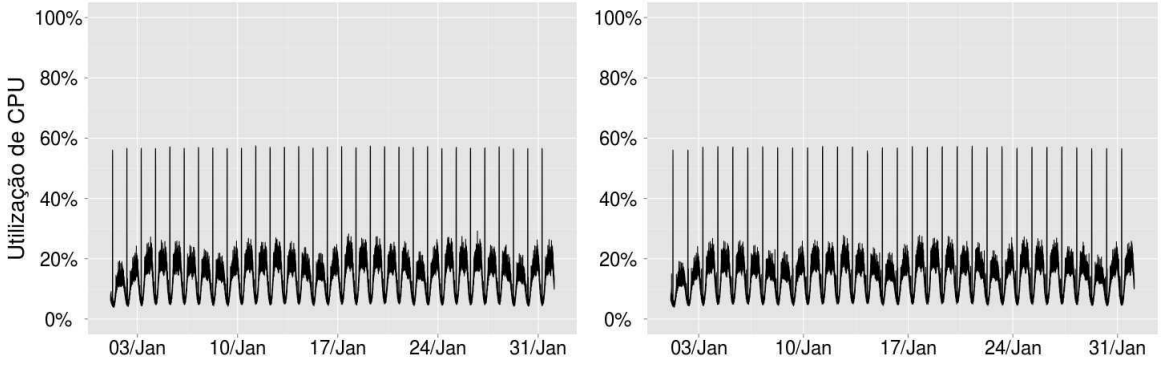


Figura A.13: Rastros 25 e 26 de curta duração.

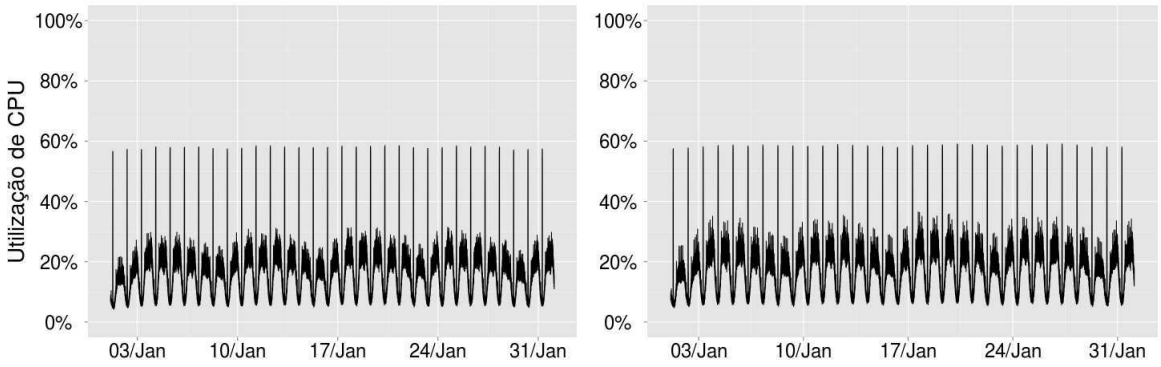


Figura A.14: Rastros 27 e 28 de curta duração.

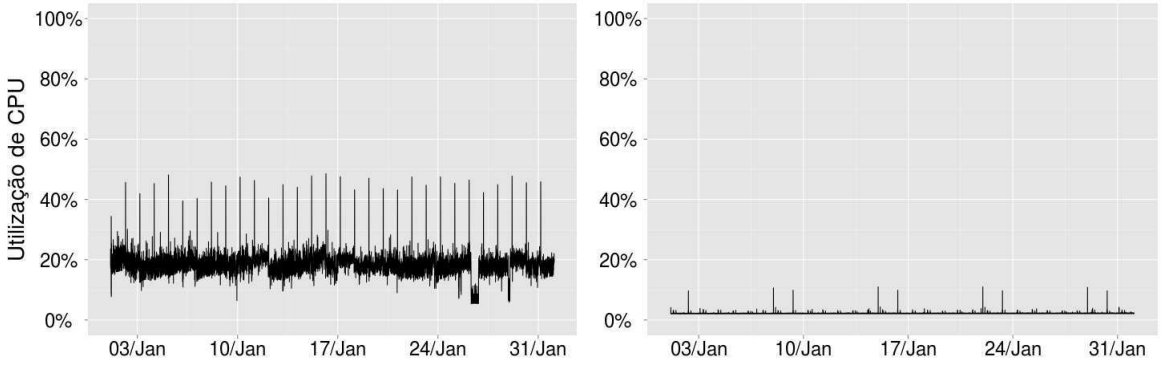


Figura A.15: Rastros 29 e 30 de curta duração.

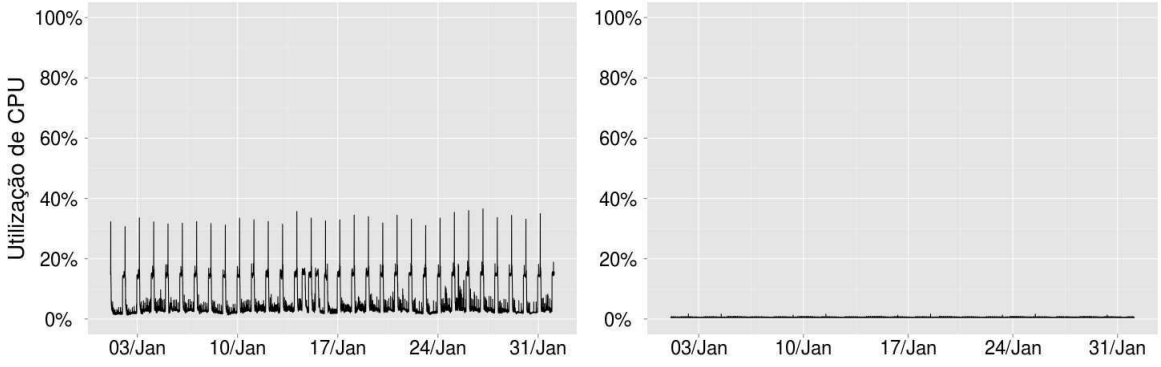


Figura A.16: Rastros 31 e 32 de curta duração.

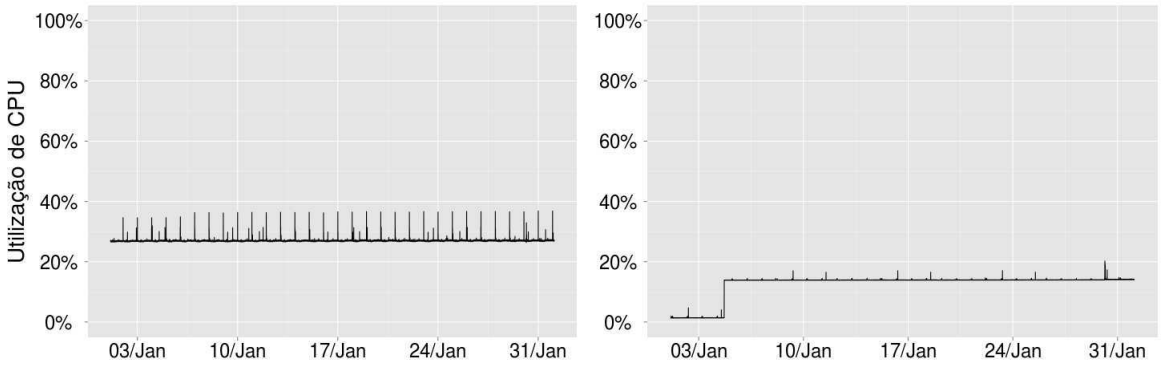


Figura A.17: Rastros 33 e 34 de curta duração.

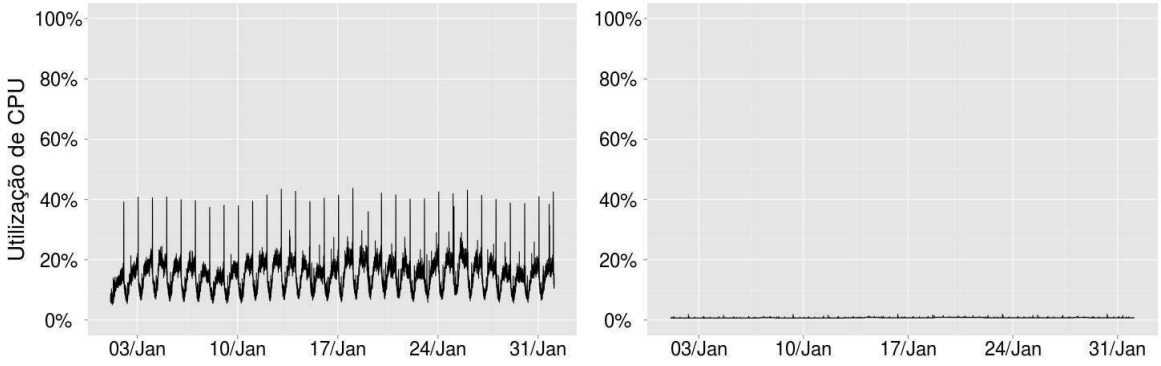


Figura A.18: Rastros 35 e 36 de curta duração.

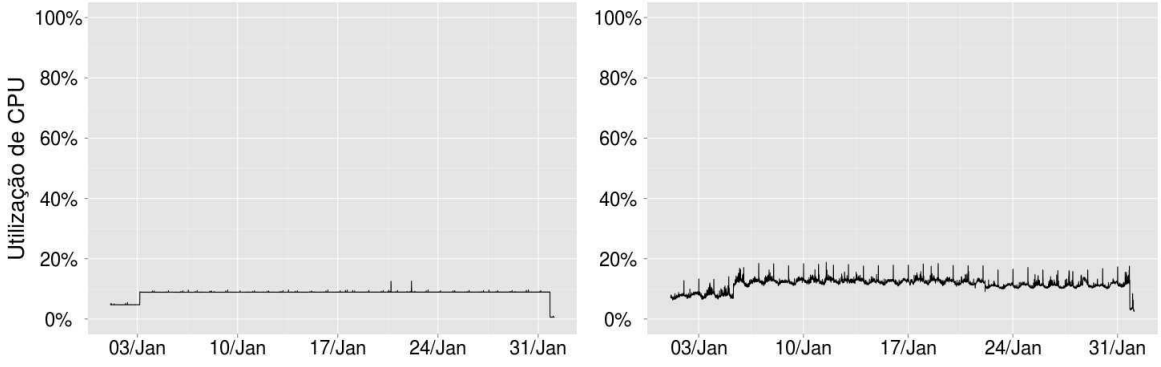


Figura A.19: Rastros 37 e 38 de curta duração.

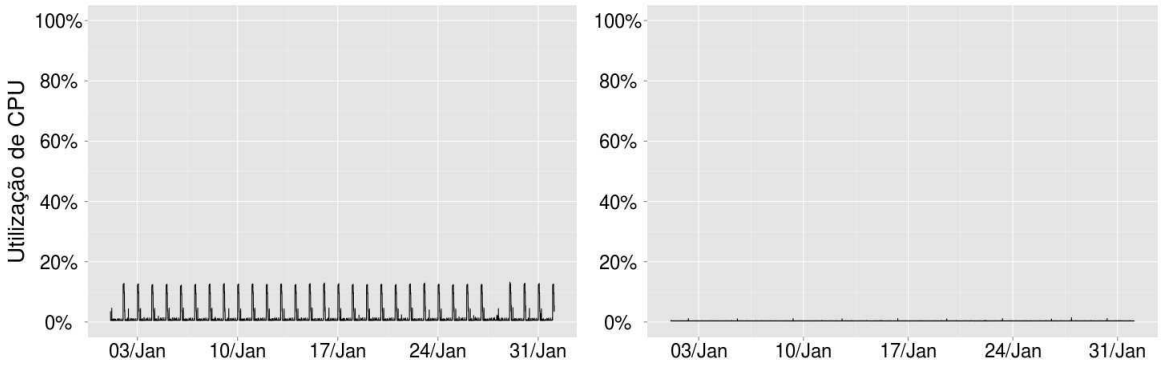


Figura A.20: Rastros 39 e 40 de curta duração.

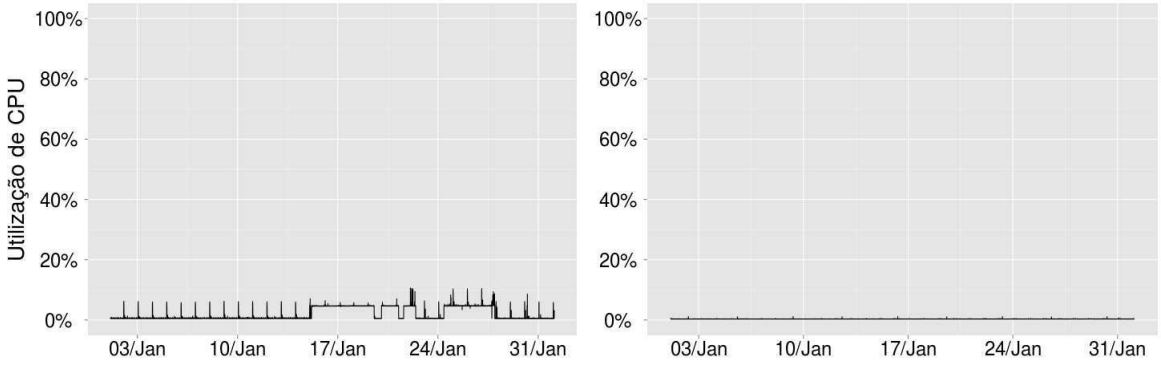


Figura A.21: Rastros 41 e 42 de curta duração.

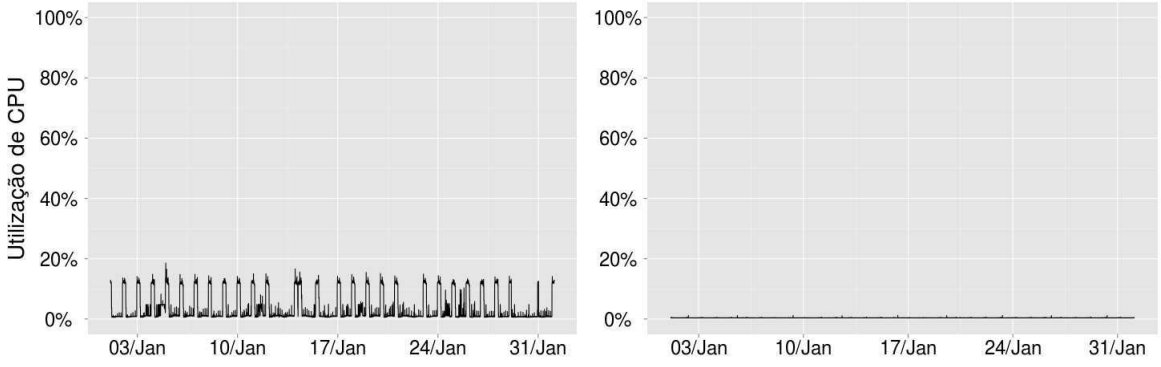


Figura A.22: Rastros 43 e 44 de curta duração.

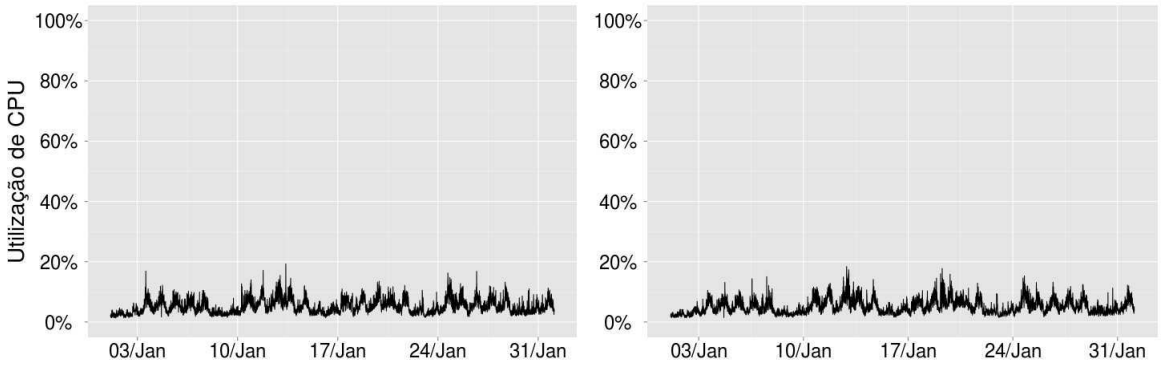


Figura A.23: Rastros 45 e 46 de curta duração.

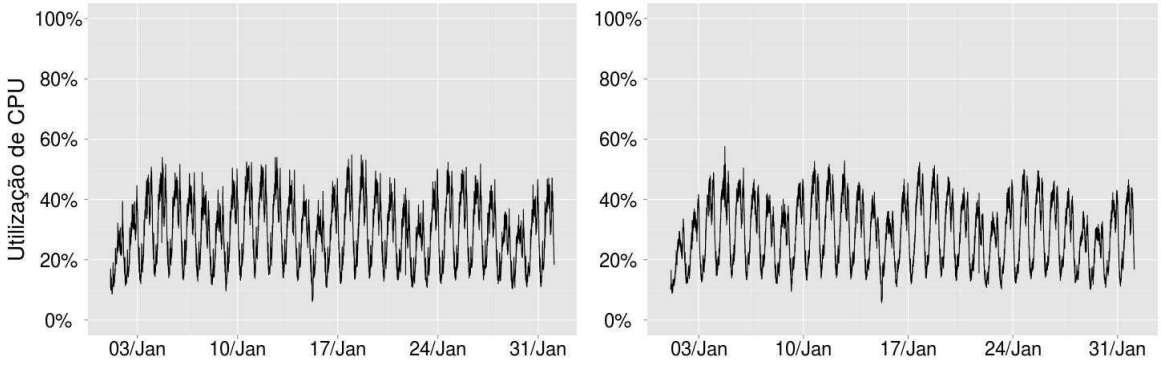


Figura A.24: Rastros 47 e 48 de curta duração.

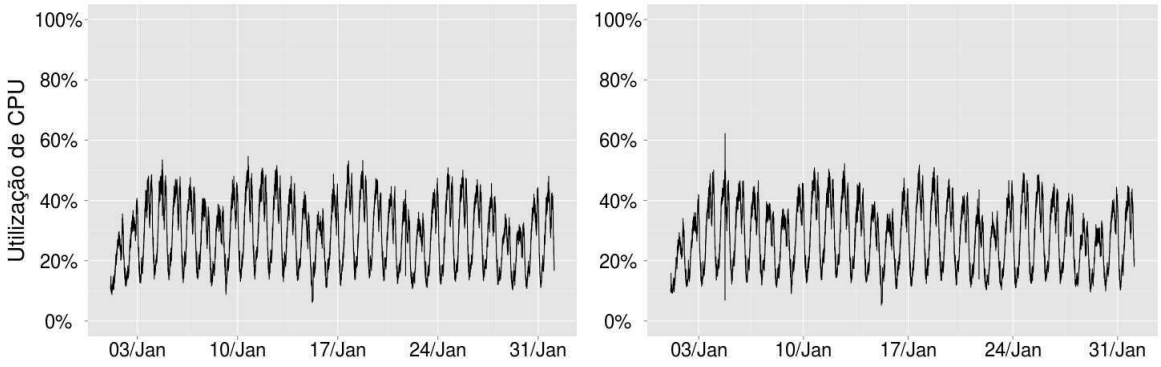


Figura A.25: Rastros 49 e 50 de curta duração.

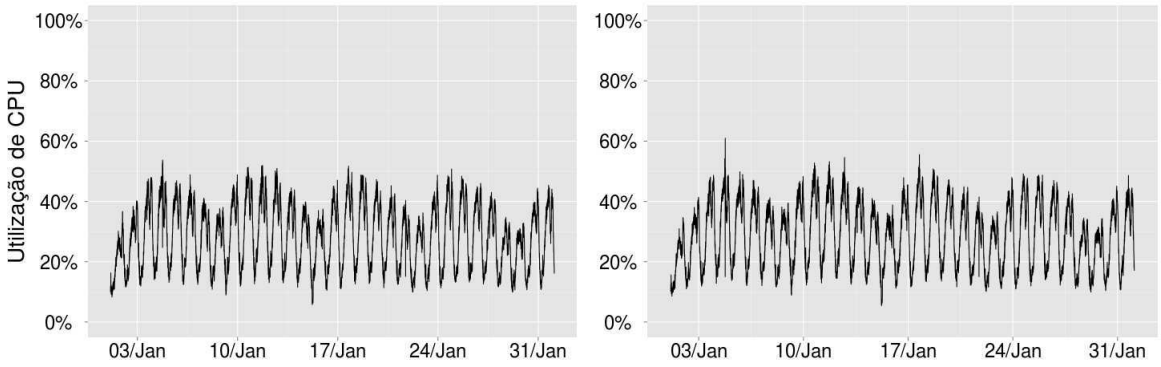


Figura A.26: Rastros 51 e 52 de curta duração.

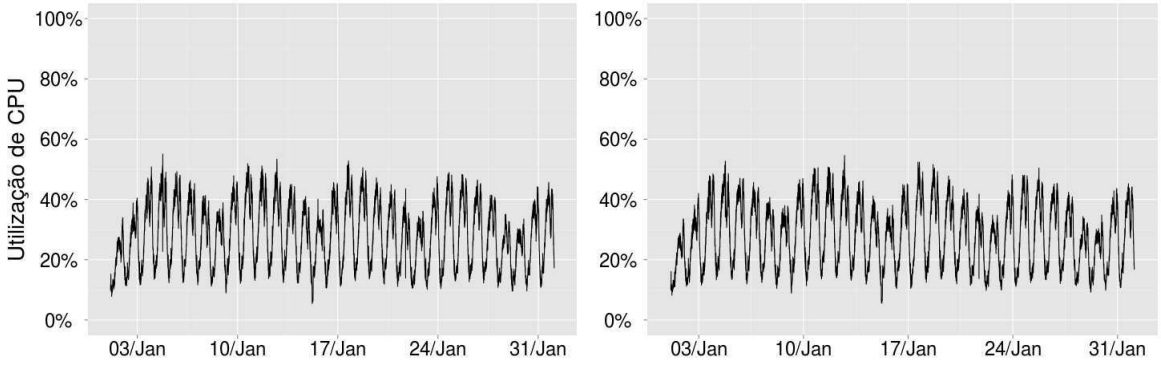


Figura A.27: Rastros 53 e 54 de curta duração.

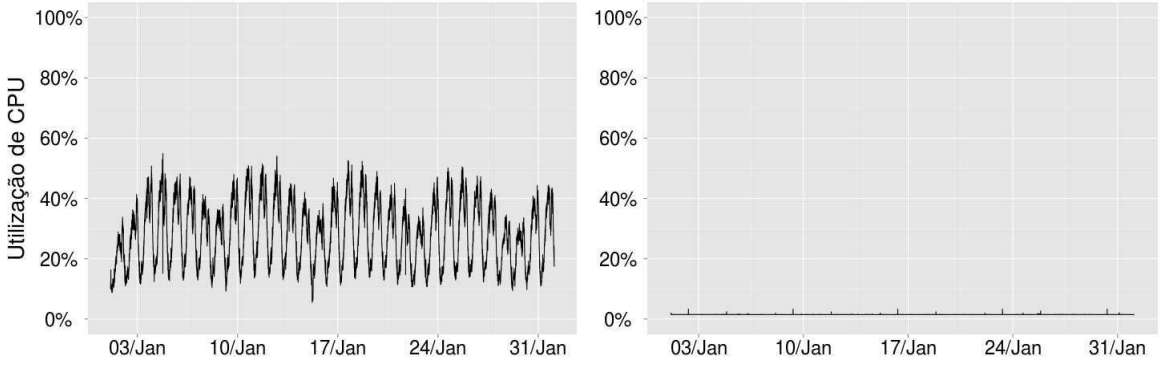


Figura A.28: Rastros 55 e 56 de curta duração.

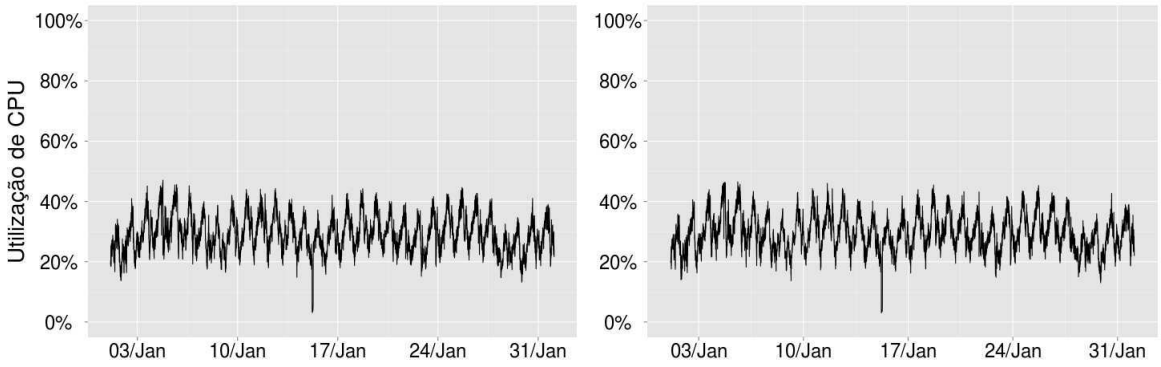


Figura A.29: Rastros 57 e 58 de curta duração.

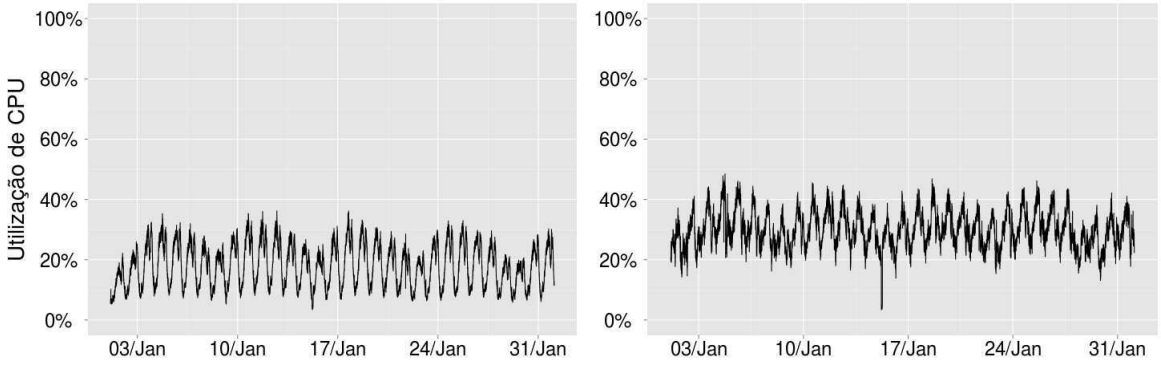


Figura A.30: Rastros 59 e 60 de curta duração.

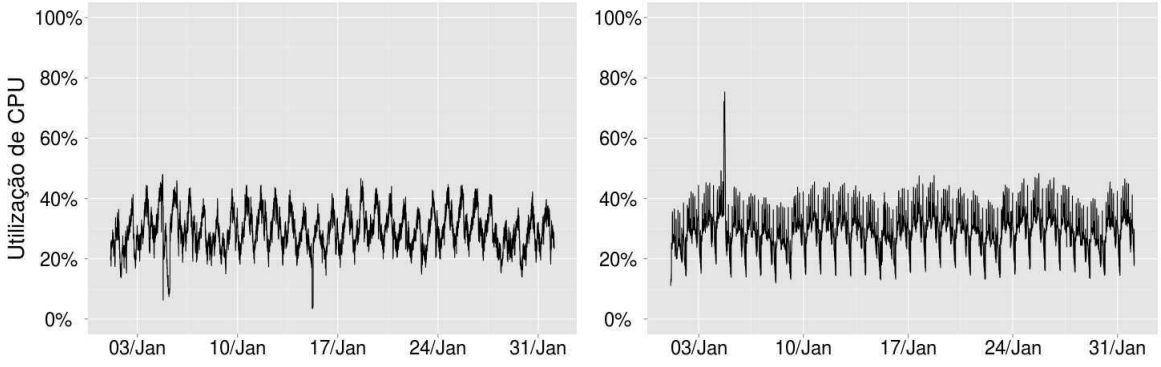


Figura A.31: Rastros 61 e 62 de curta duração.

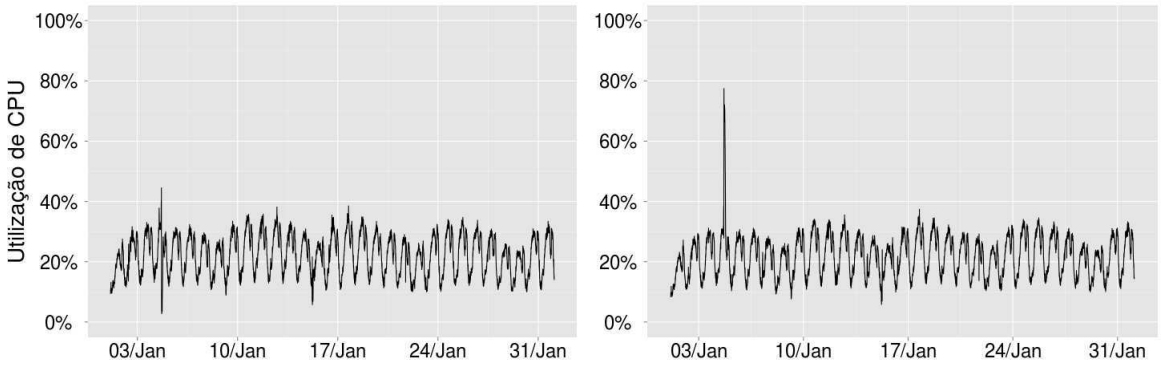


Figura A.32: Rastros 63 e 64 de curta duração.

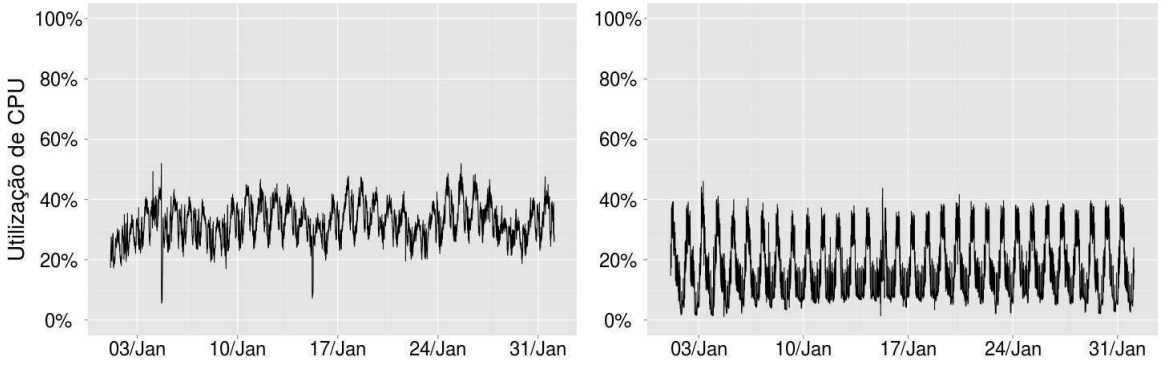


Figura A.33: Rastros 65 e 66 de curta duração.

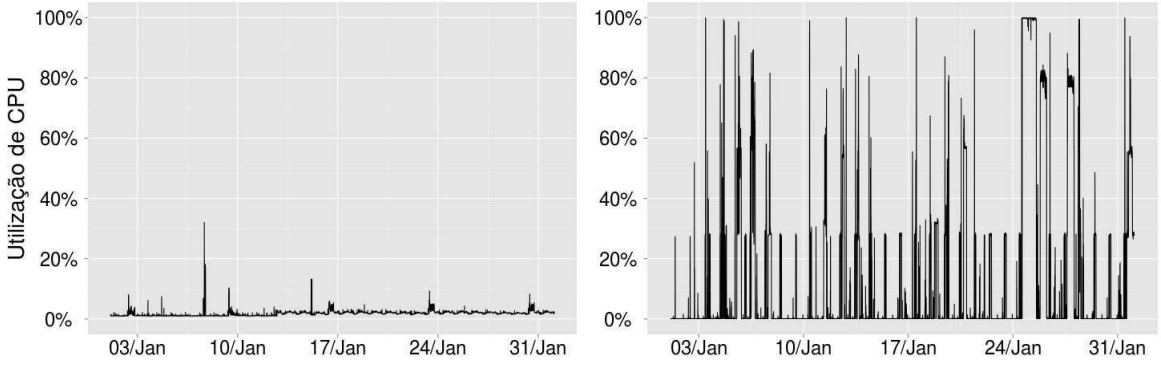


Figura A.34: Rastros 67 e 68 de curta duração.

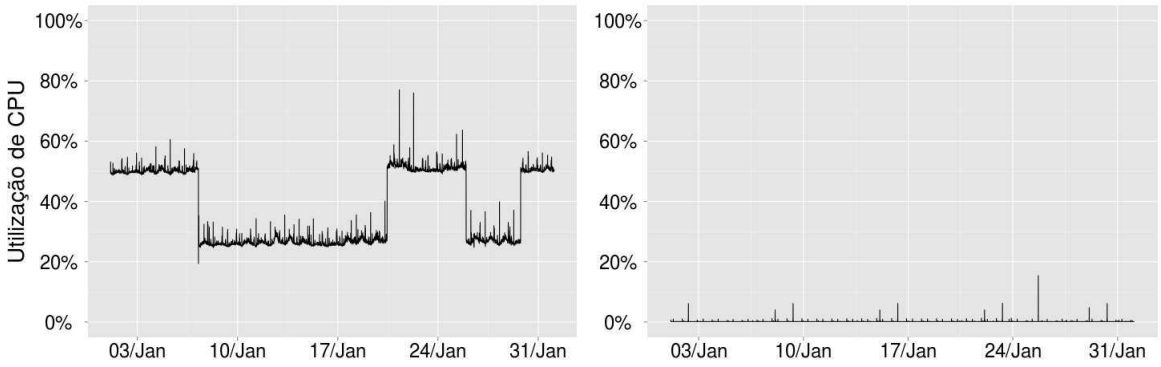


Figura A.35: Rastros 69 e 70 de curta duração.

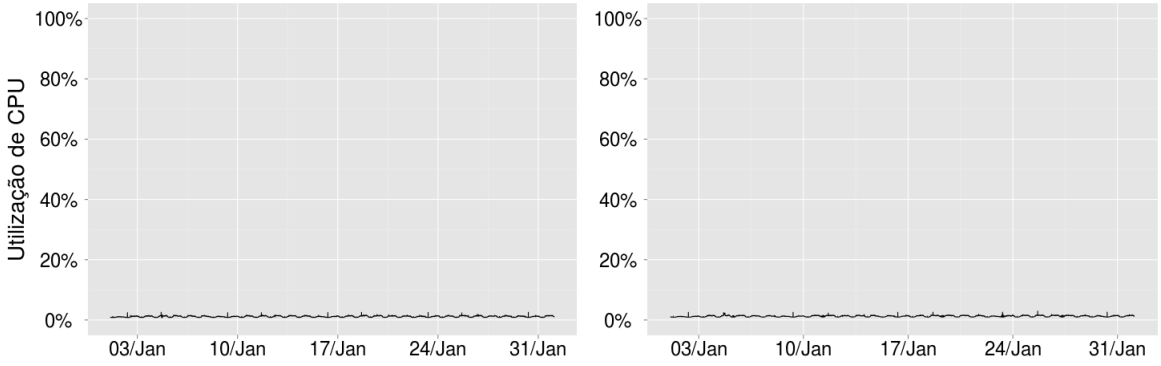


Figura A.36: Rastros 71 e 72 de curta duração.

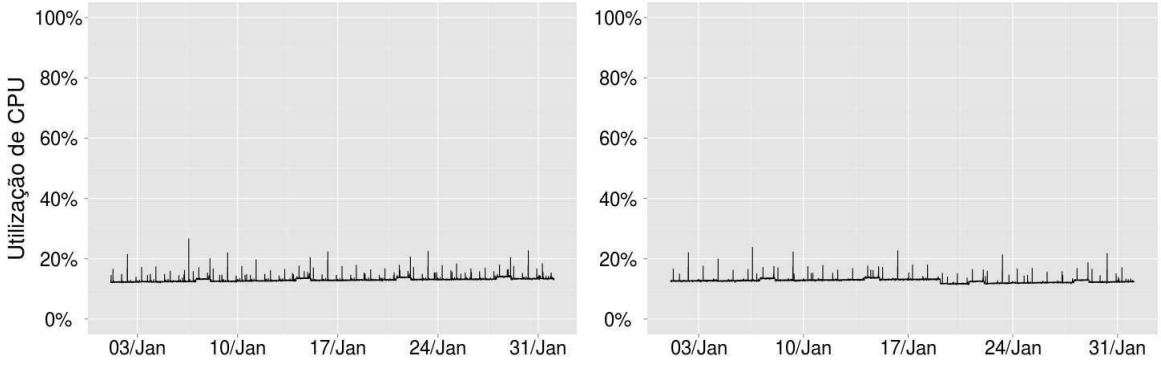


Figura A.37: Rastros 73 e 74 de curta duração.

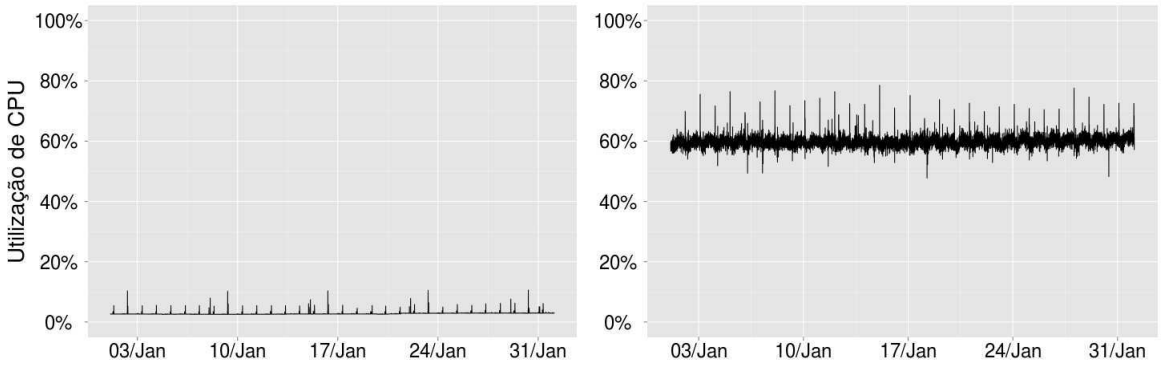


Figura A.38: Rastros 75 e 76 de curta duração.

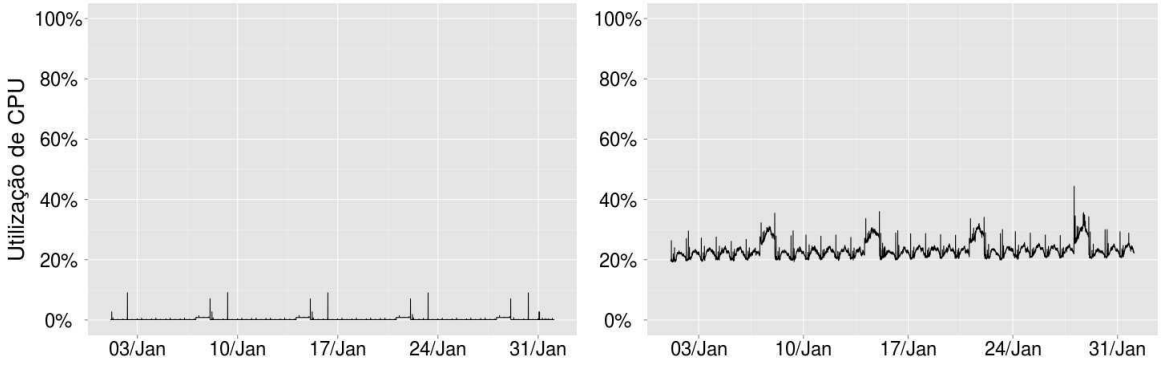


Figura A.39: Rastros 77 e 78 de curta duração.

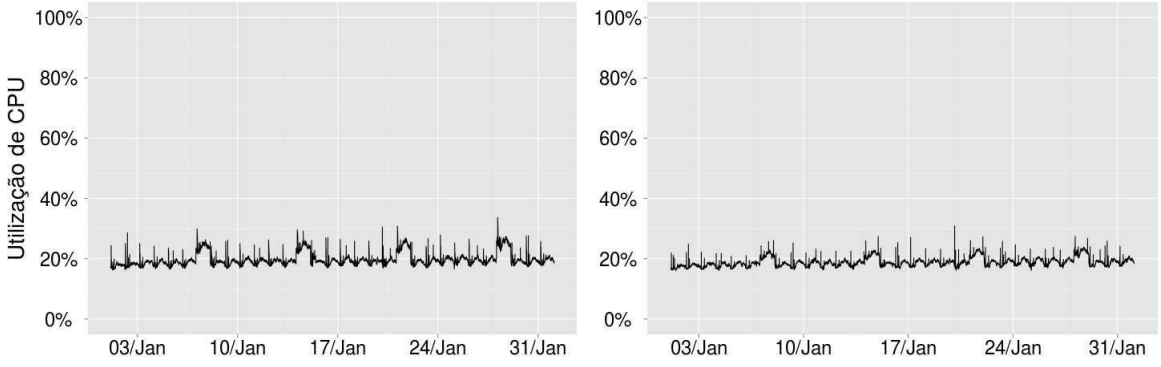


Figura A.40: Rastros 79 e 80 de curta duração.

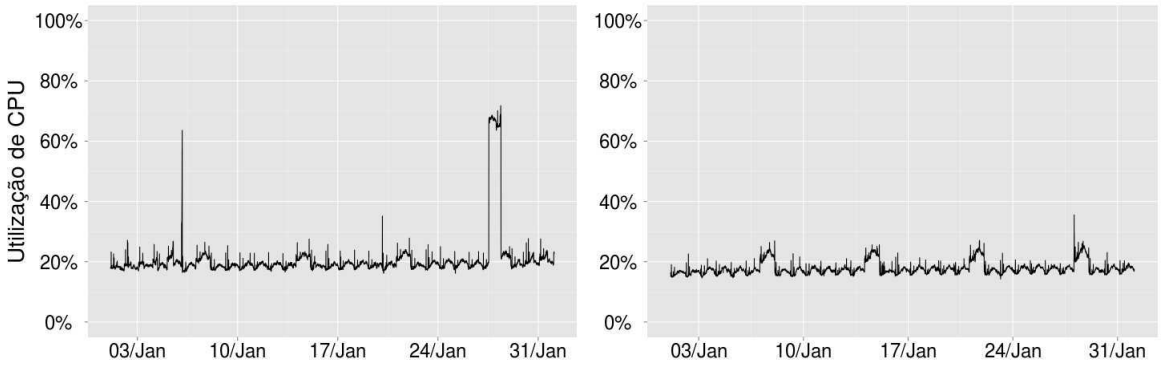


Figura A.41: Rastros 81 e 82 de curta duração.

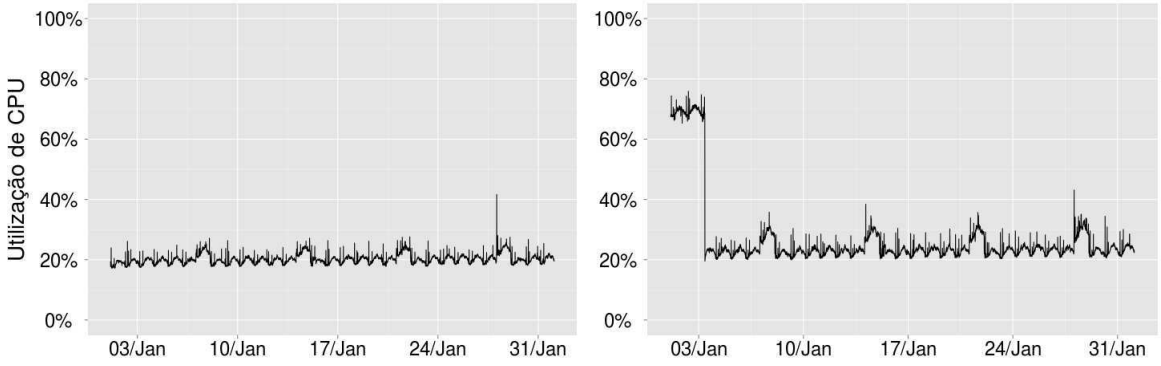


Figura A.42: Rastros 83 e 84 de curta duração.

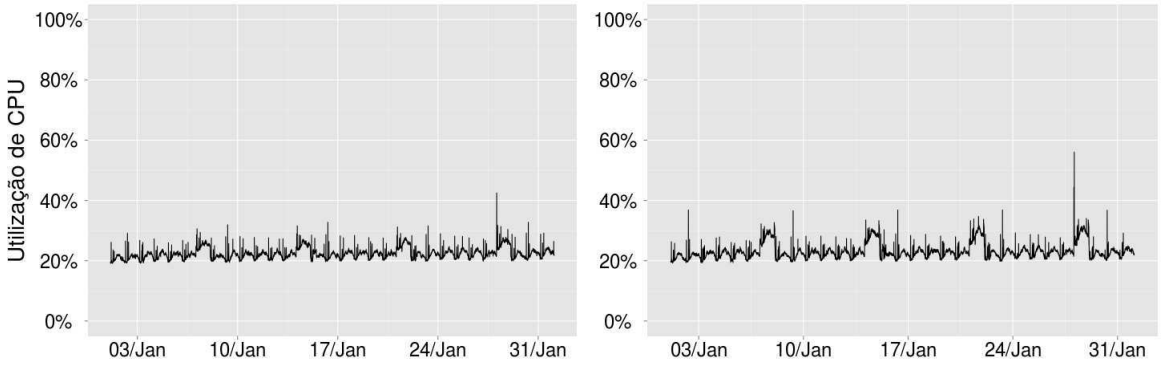


Figura A.43: Rastros 85 e 86 de curta duração.

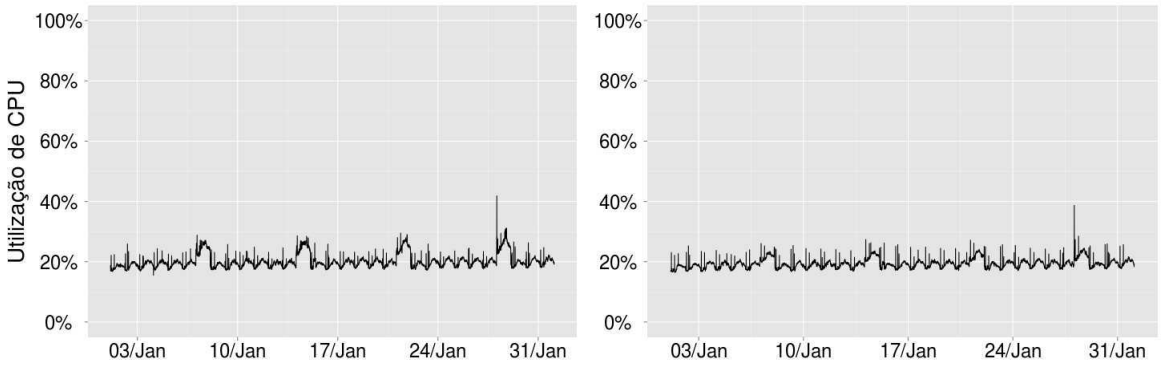


Figura A.44: Rastros 87 e 88 de curta duração.

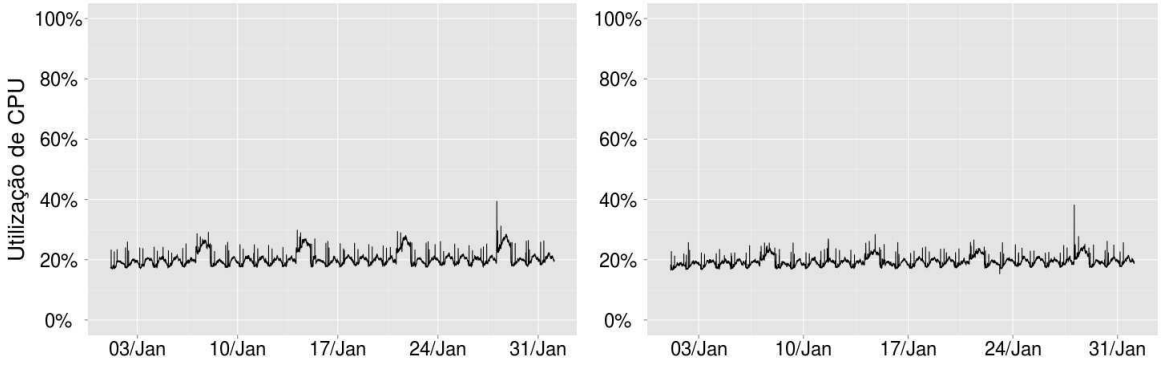


Figura A.45: Rastros 89 e 90 de curta duração.

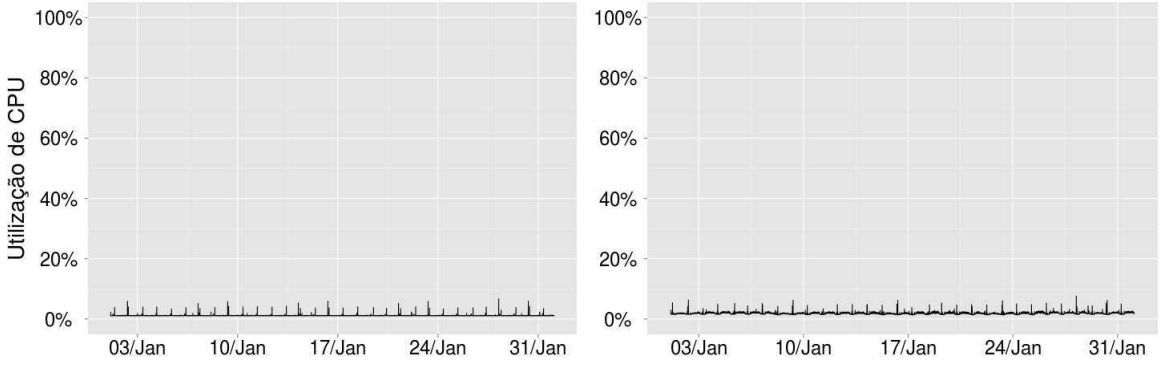


Figura A.46: Rastros 91 e 92 de curta duração.

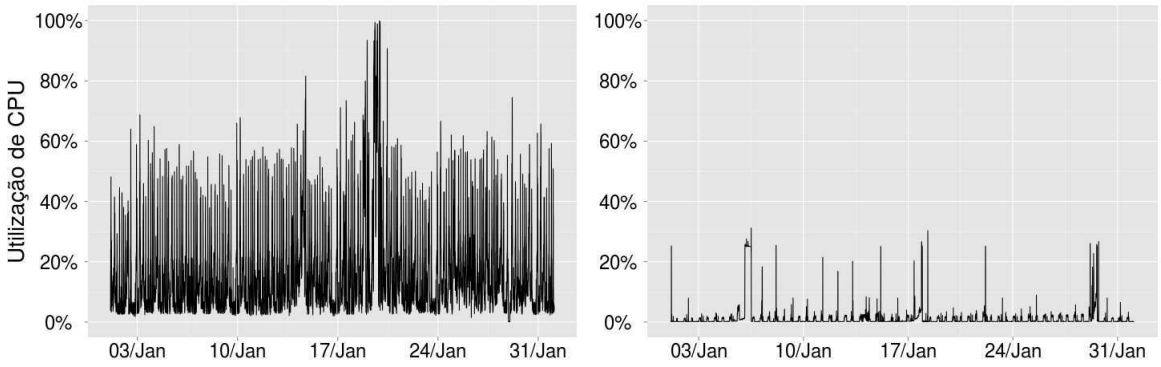


Figura A.47: Rastros 93 e 94 de curta duração.

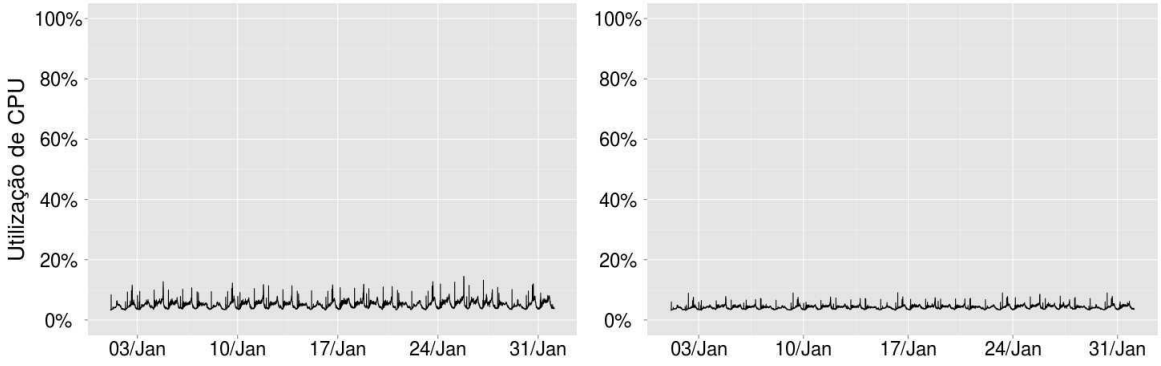


Figura A.48: Rastros 95 e 96 de curta duração.

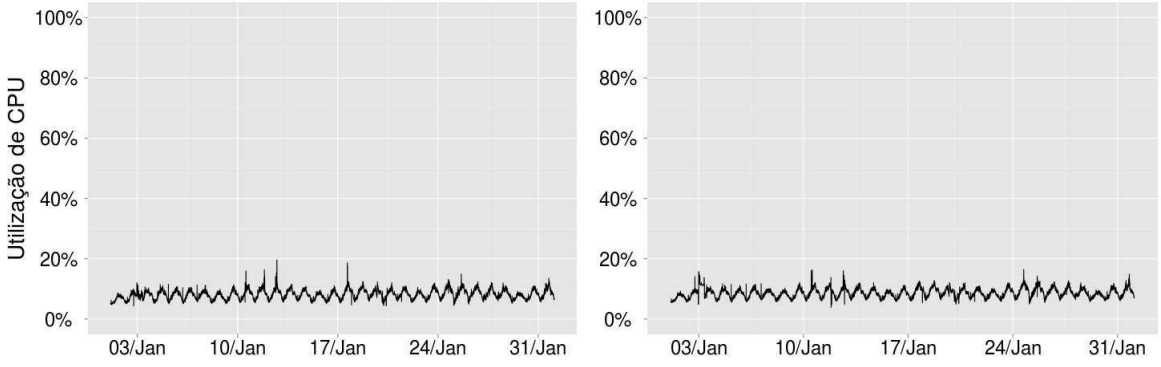


Figura A.49: Rastros 97 e 98 de curta duração.

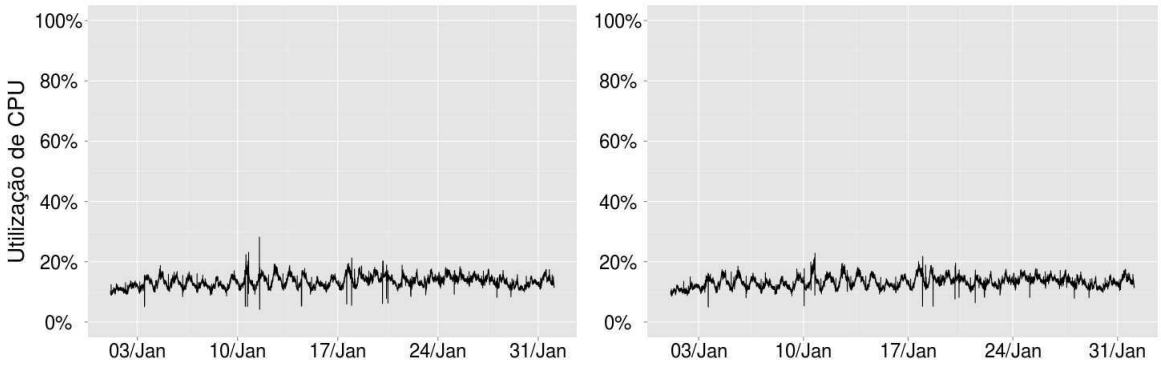


Figura A.50: Rastros 99 e 100 de curta duração.

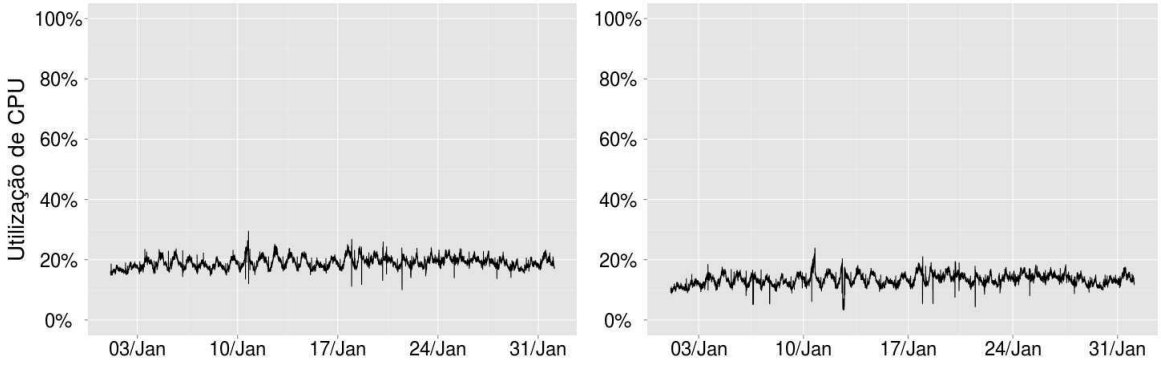


Figura A.51: Rastros 101 e 102 de curta duração.

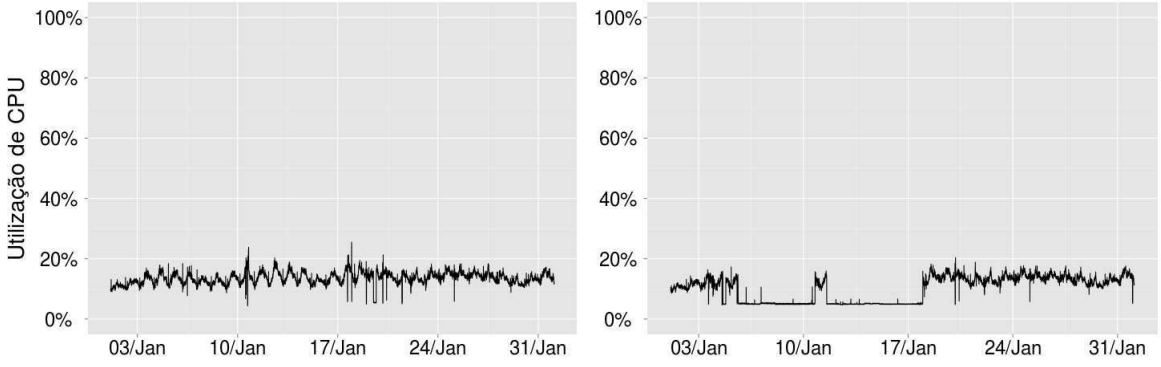


Figura A.52: Rastros 103 e 104 de curta duração.

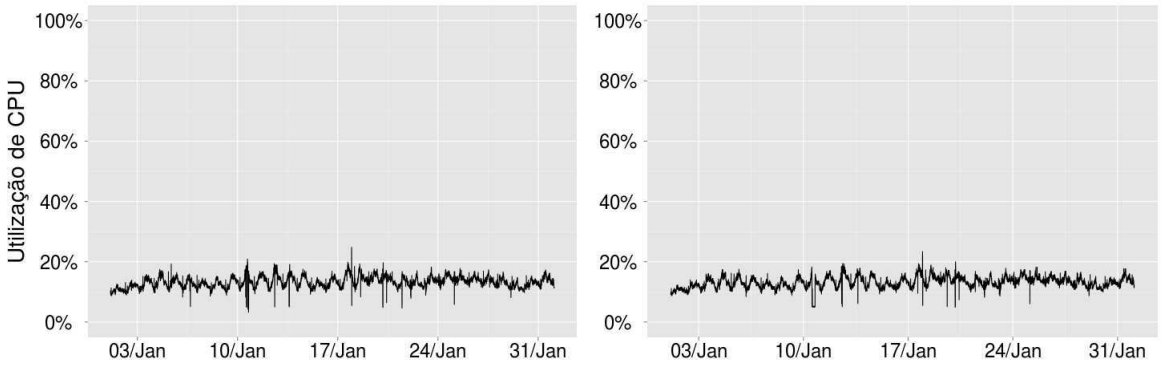


Figura A.53: Rastros 105 e 106 de curta duração.

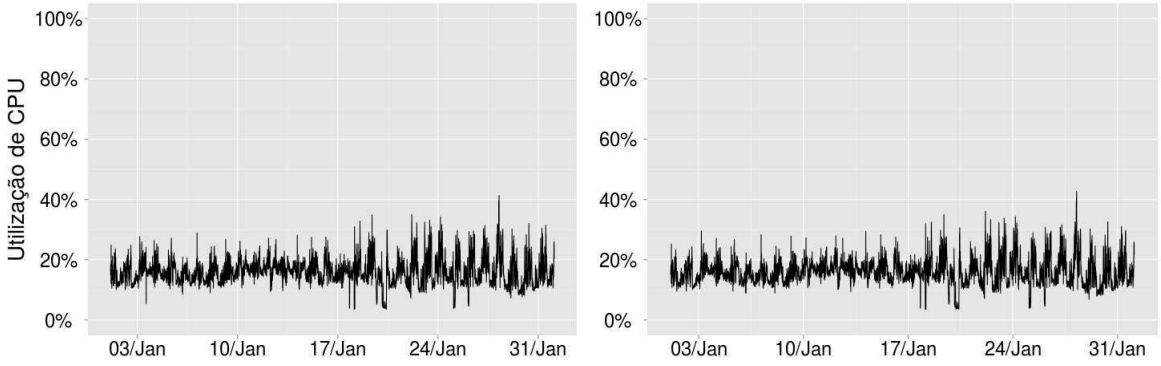


Figura A.54: Rastros 107 e 108 de curta duração.

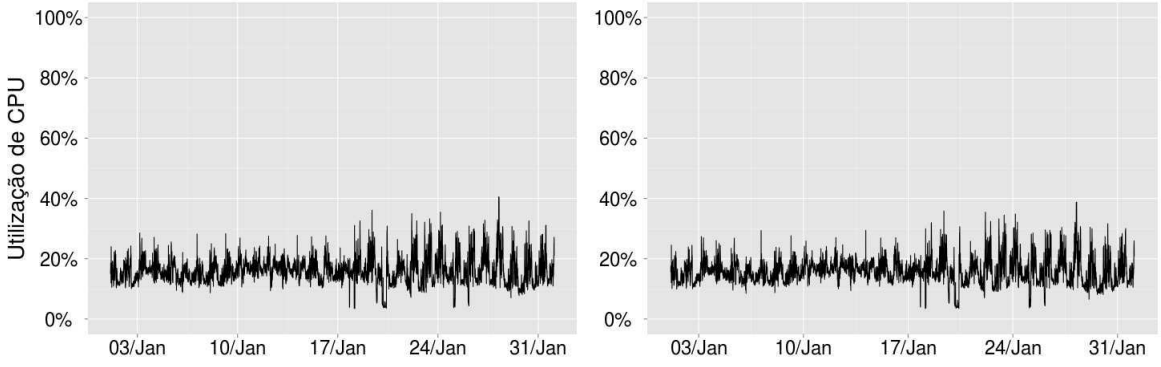


Figura A.55: Rastros 109 e 110 de curta duração.

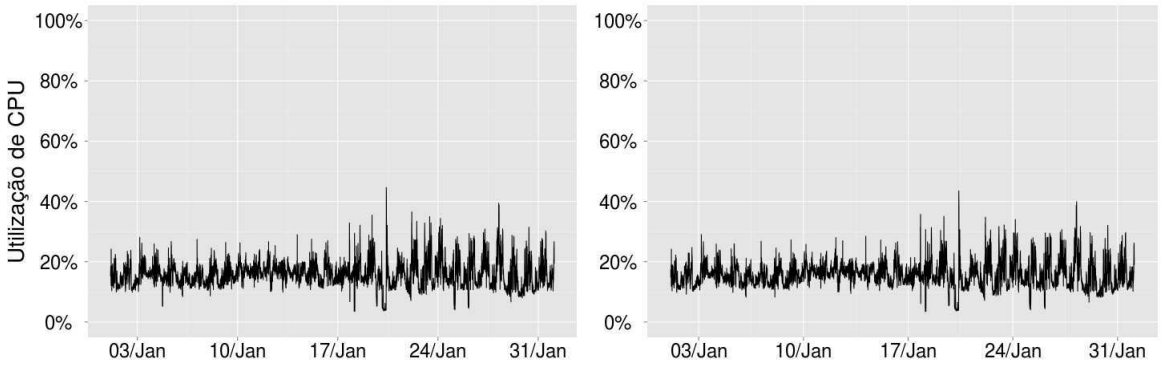


Figura A.56: Rastros 111 e 112 de curta duração.

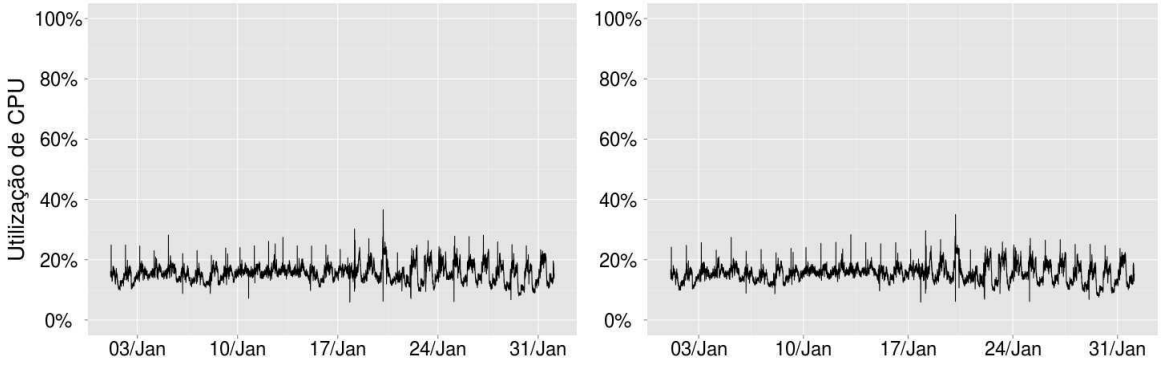


Figura A.57: Rastros 113 e 114 de curta duração.

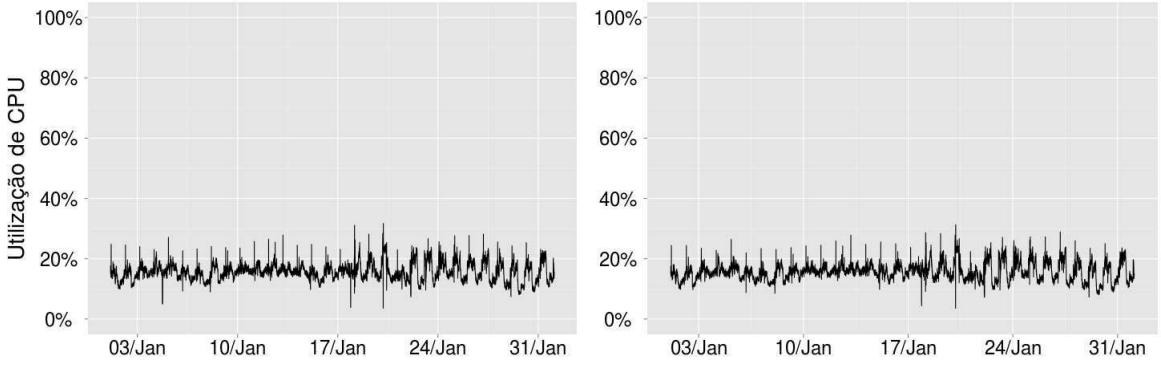


Figura A.58: Rastros 115 e 116 de curta duração.

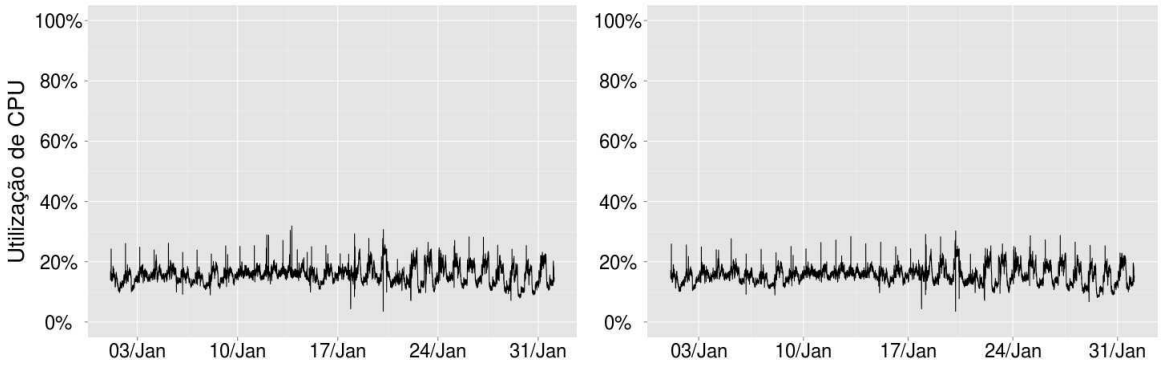


Figura A.59: Rastros 117 e 118 de curta duração.

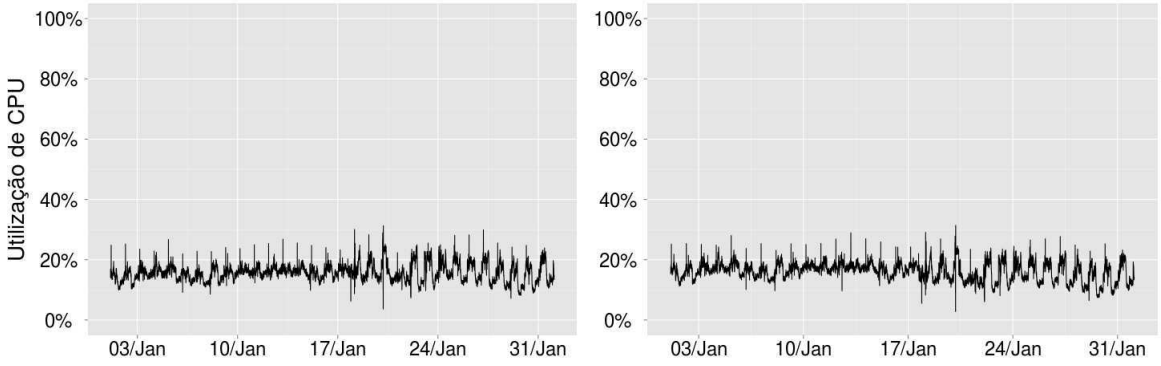


Figura A.60: Rastros 119 e 120 de curta duração.

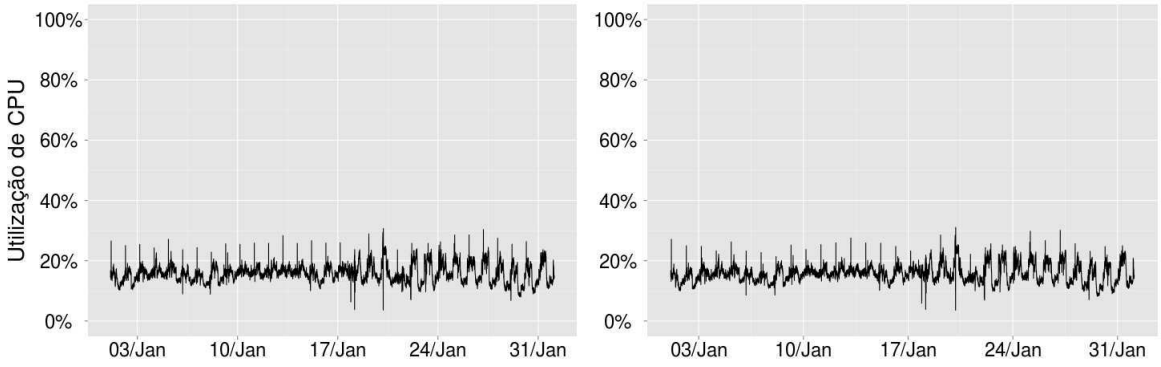


Figura A.61: Rastros 121 e 122 de curta duração.

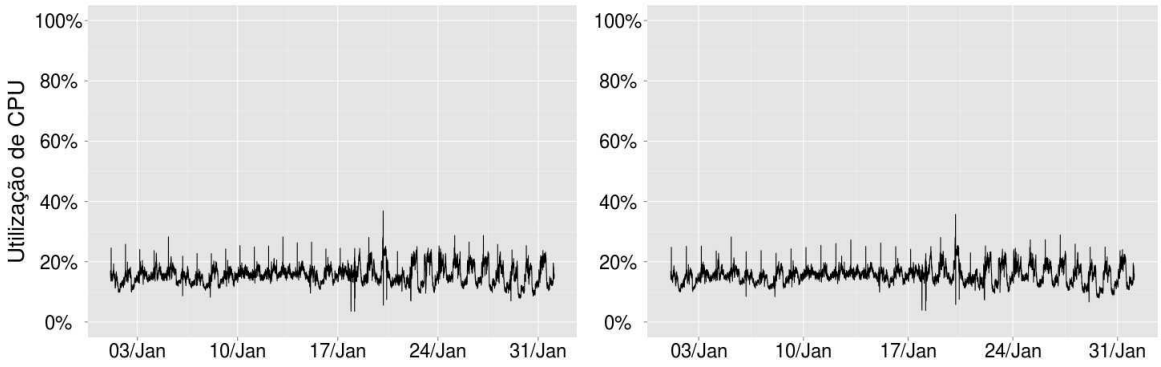


Figura A.62: Rastros 123 e 124 de curta duração.

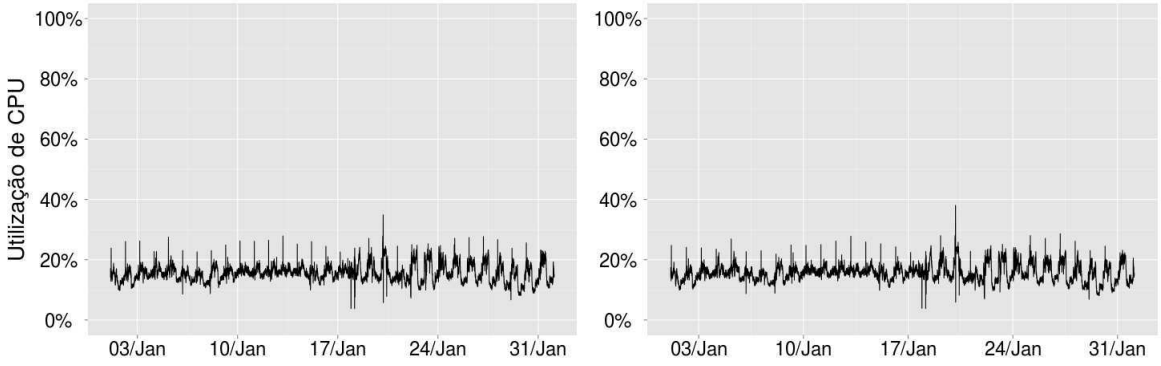


Figura A.63: Rastros 125 e 126 de curta duração.

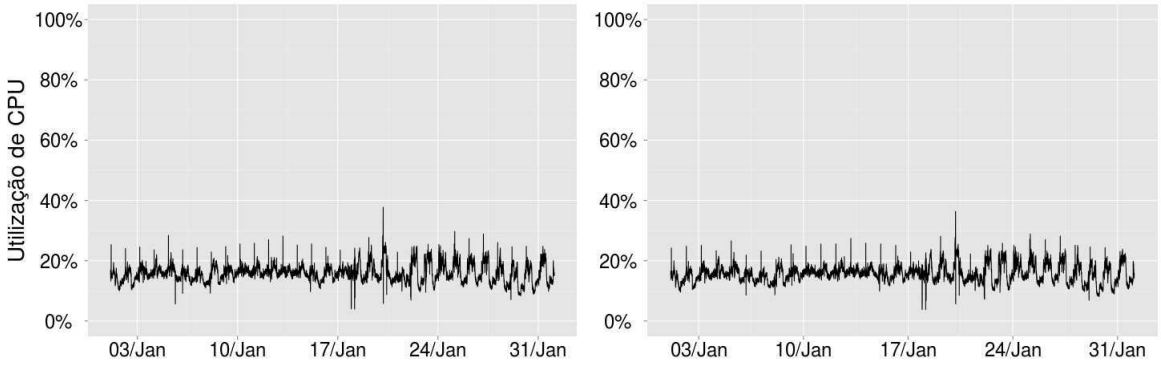


Figura A.64: Rastros 127 e 128 de curta duração.

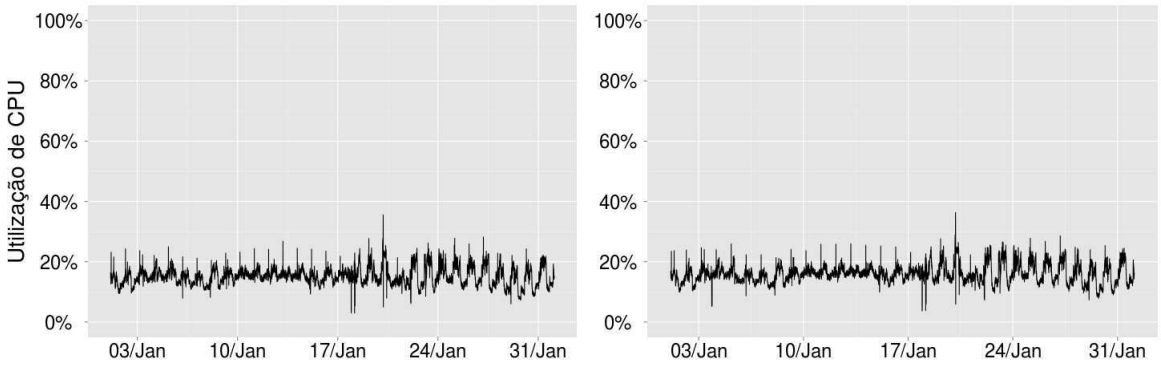


Figura A.65: Rastros 129 e 130 de curta duração.

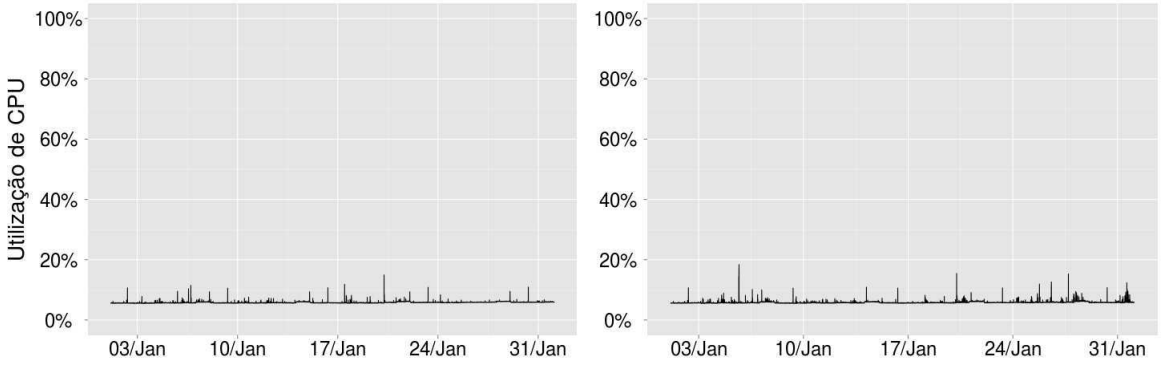


Figura A.66: Rastros 131 e 132 de curta duração.

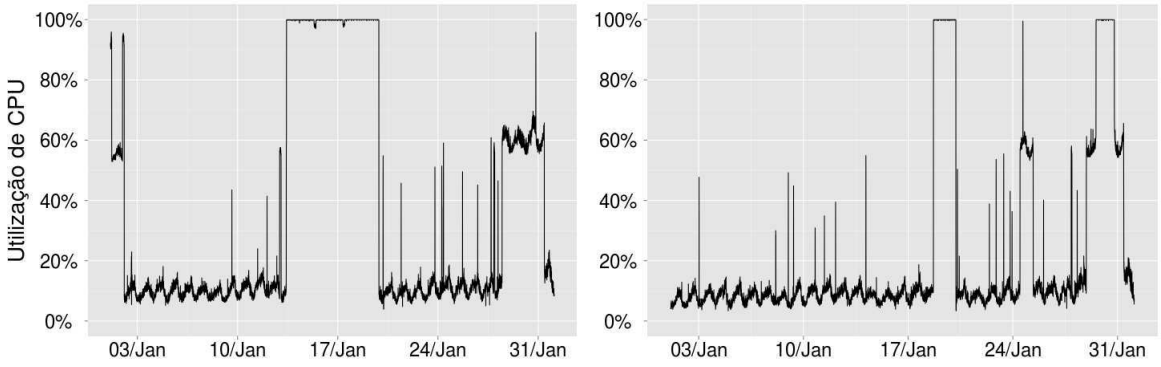


Figura A.67: Rastros 133 e 134 de curta duração.

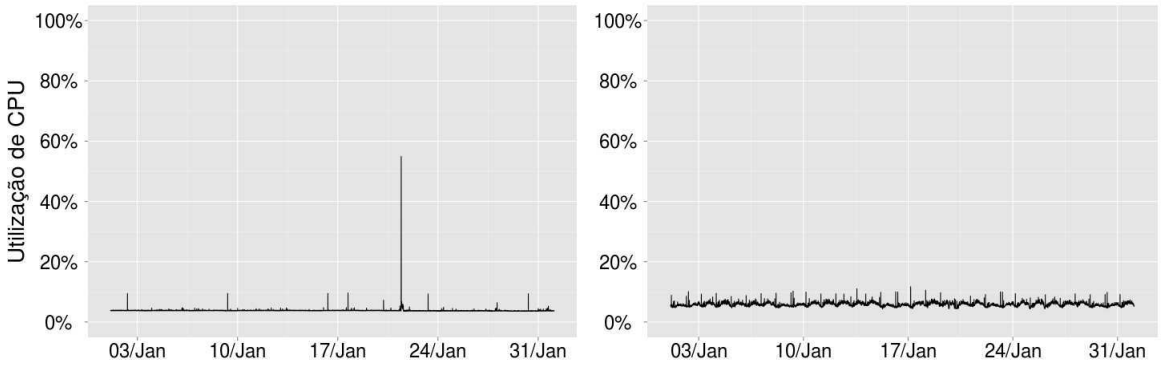


Figura A.68: Rastros 135 e 136 de curta duração.

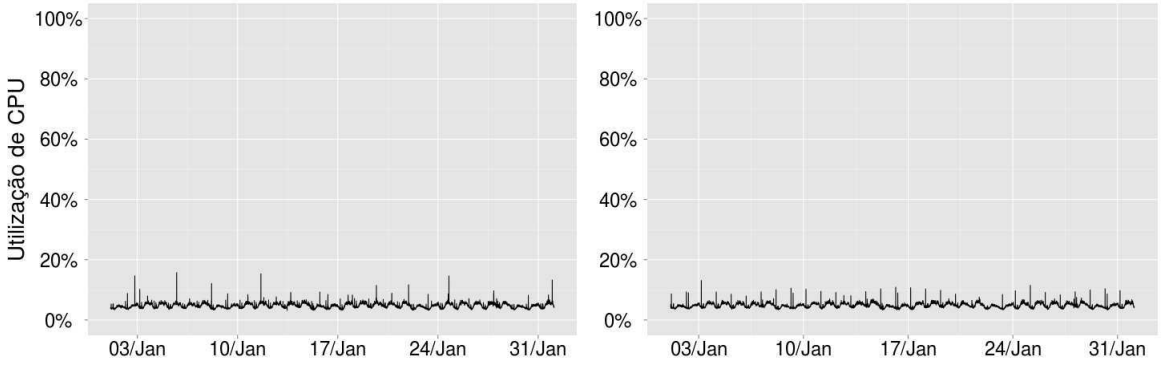


Figura A.69: Rastros 137 e 138 de curta duração.

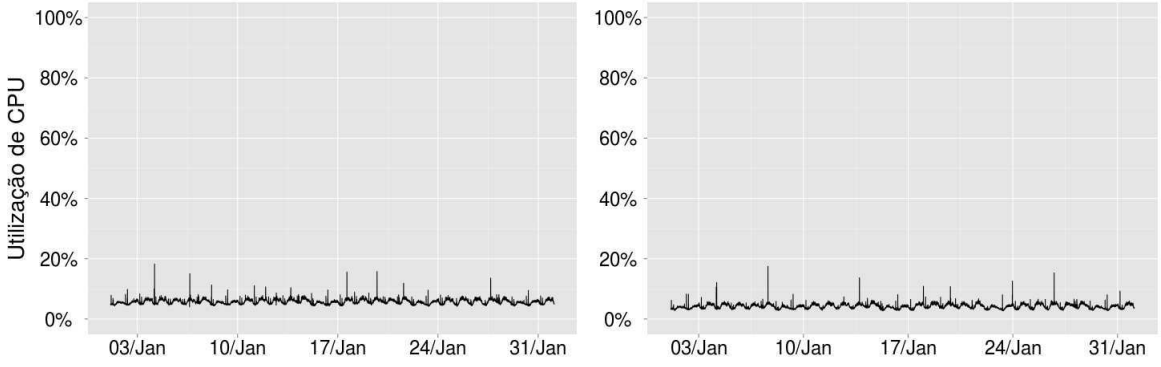


Figura A.70: Rastros 139 e 140 de curta duração.

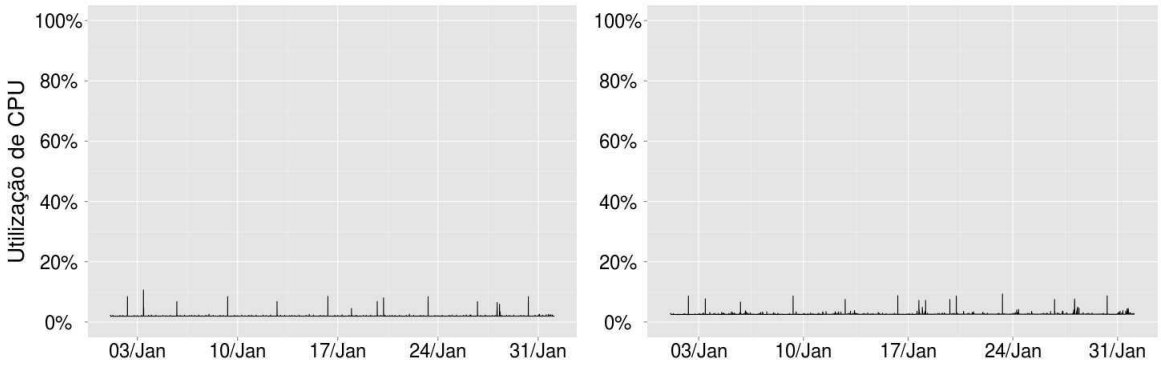


Figura A.71: Rastros 141 e 142 de curta duração.

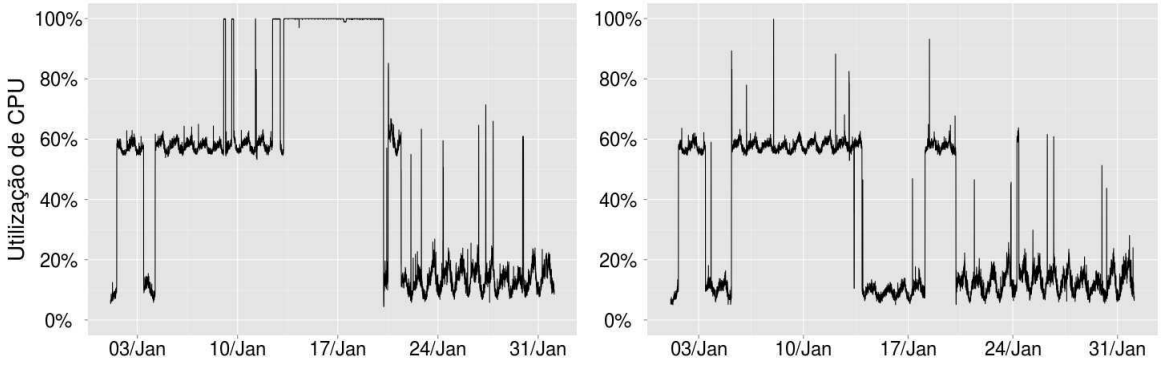


Figura A.72: Rastros 143 e 144 de curta duração.

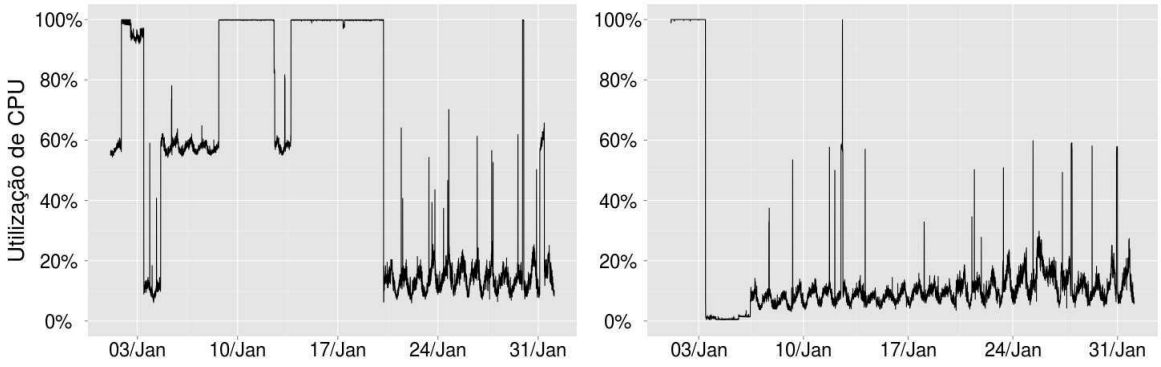


Figura A.73: Rastros 145 e 146 de curta duração.

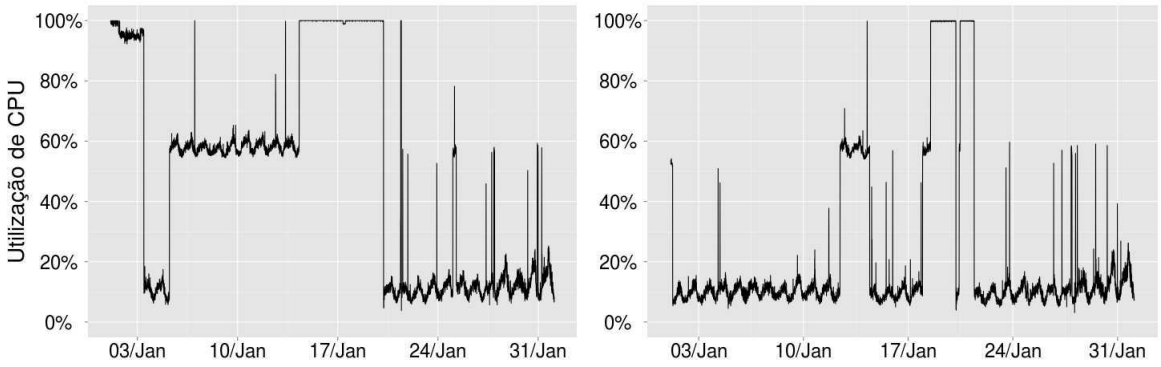


Figura A.74: Rastros 147 e 148 de curta duração.

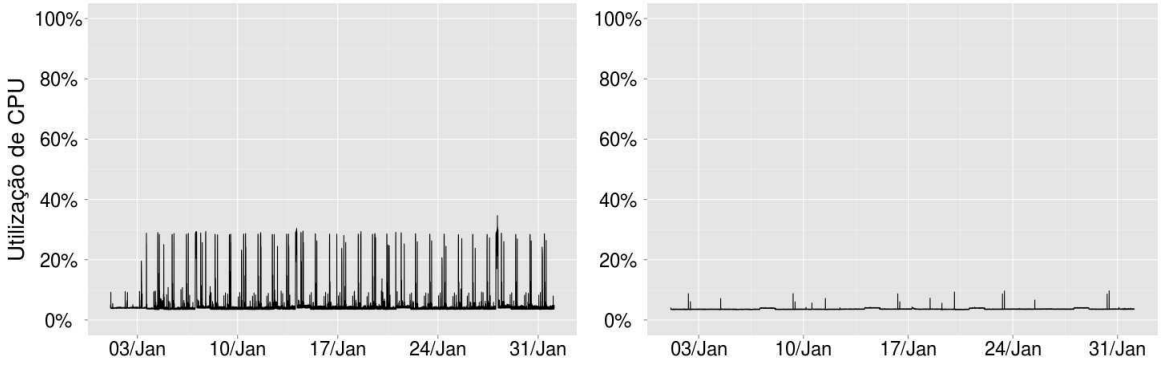


Figura A.75: Rastros 149 e 150 de curta duração.

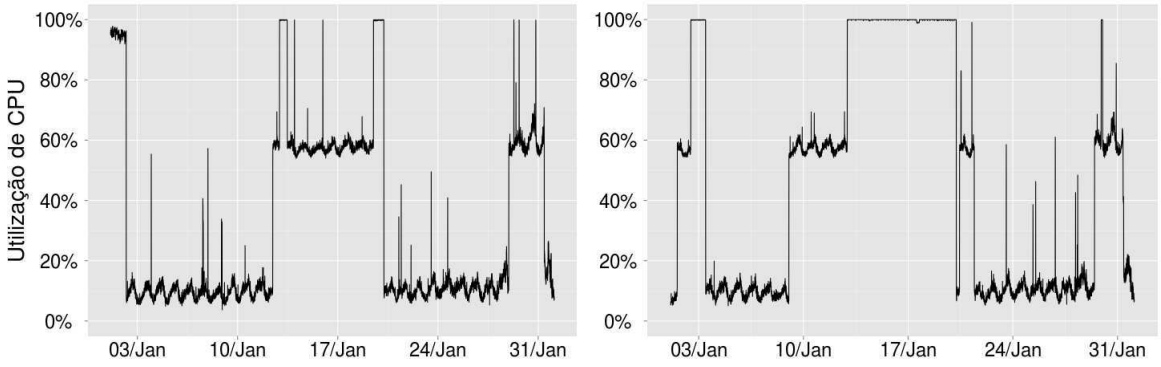


Figura A.76: Rastros 151 e 152 de curta duração.

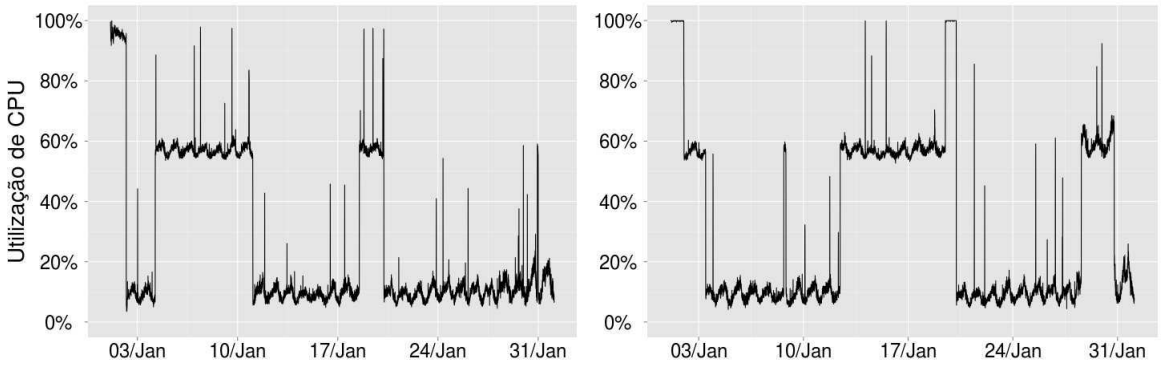


Figura A.77: Rastros 153 e 154 de curta duração.

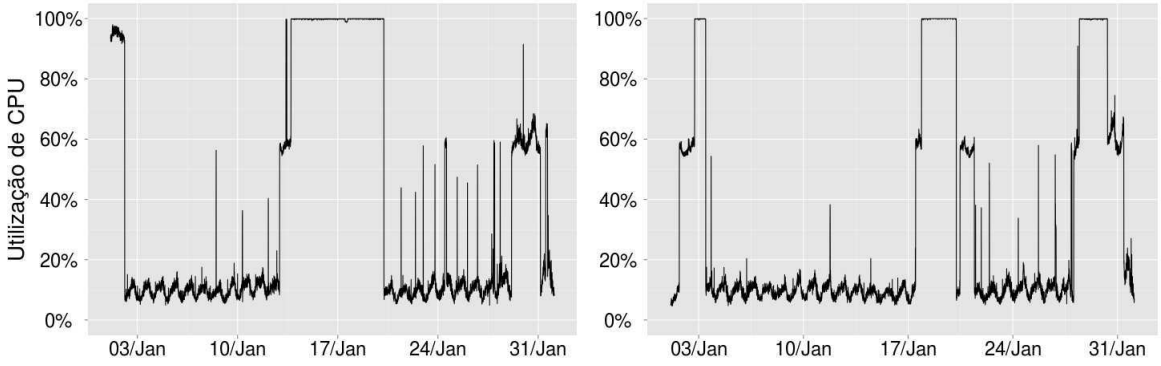


Figura A.78: Rastros 155 e 156 de curta duração.

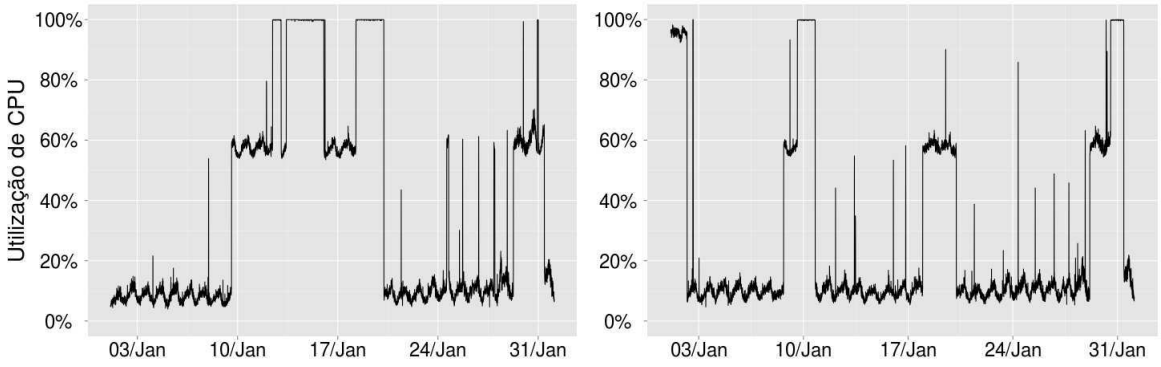


Figura A.79: Rastros 157 e 158 de curta duração.

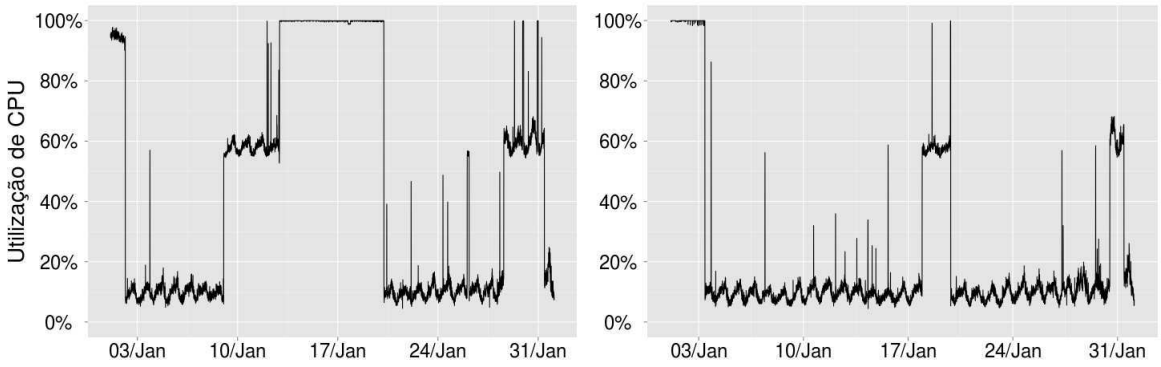


Figura A.80: Rastros 159 e 160 de curta duração.

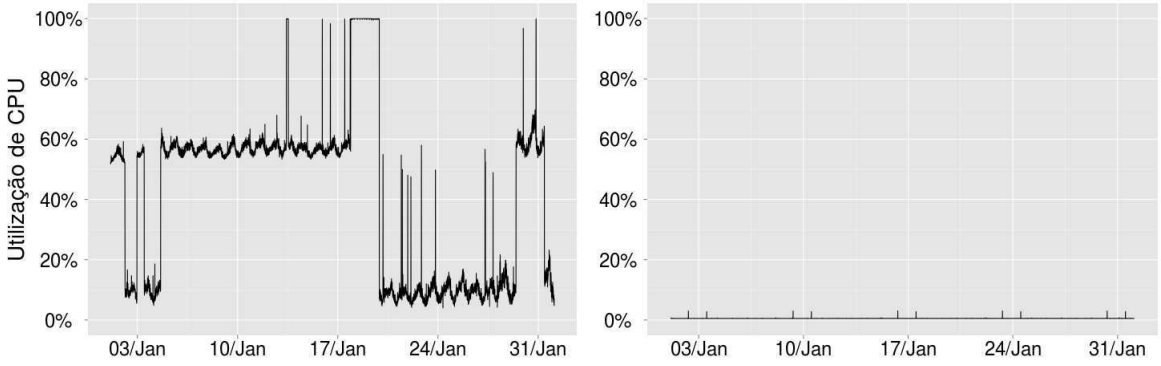


Figura A.81: Rastros 161 e 162 de curta duração.

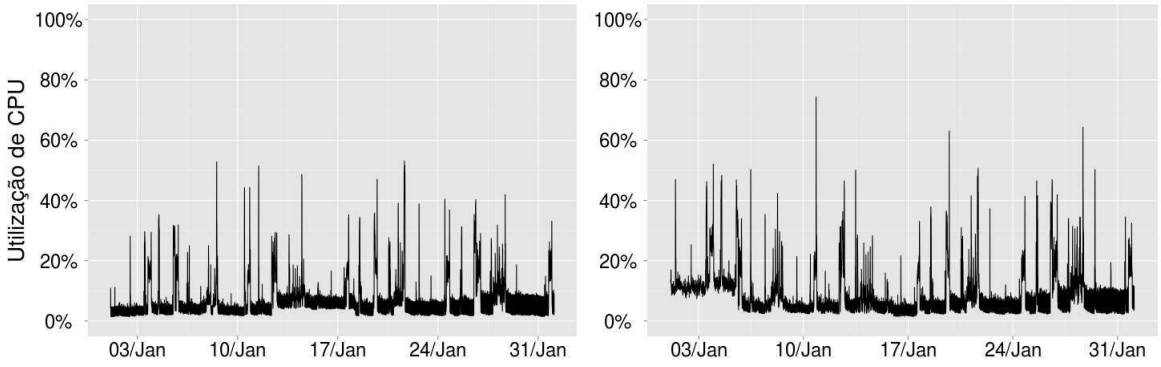


Figura A.82: Rastros 163 e 164 de curta duração.

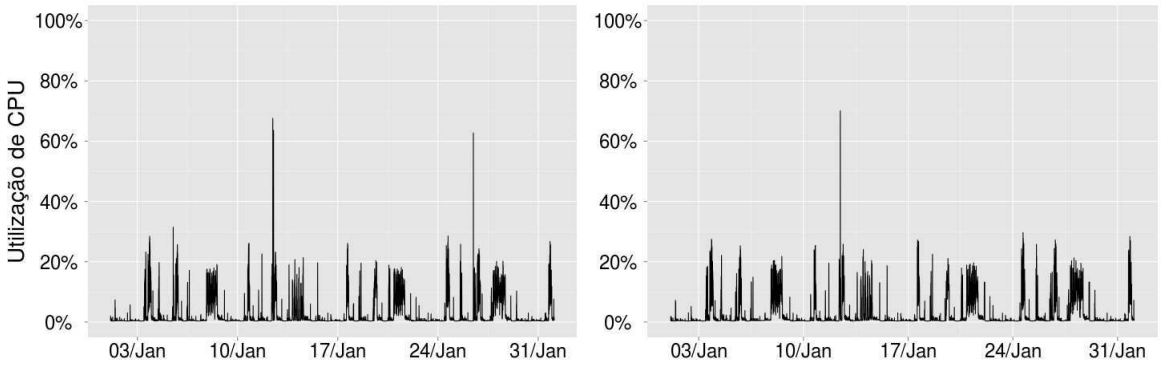


Figura A.83: Rastros 165 e 166 de curta duração.

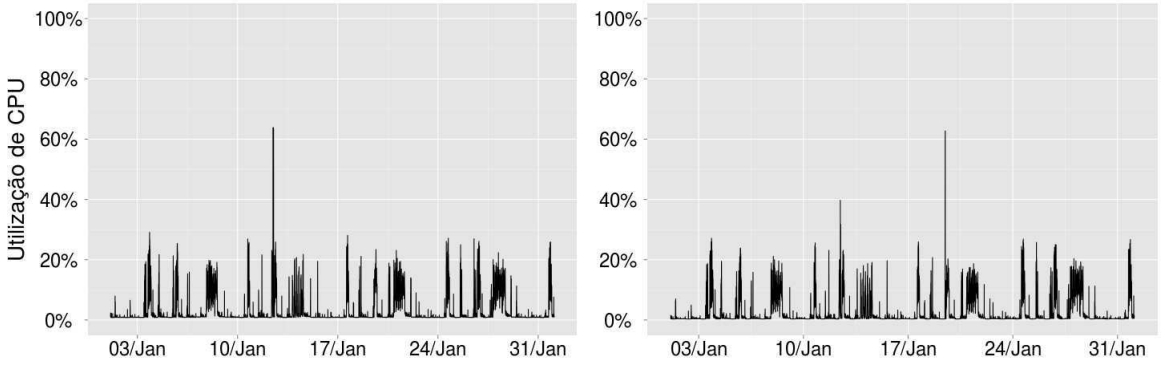


Figura A.84: Rastros 167 e 168 de curta duração.

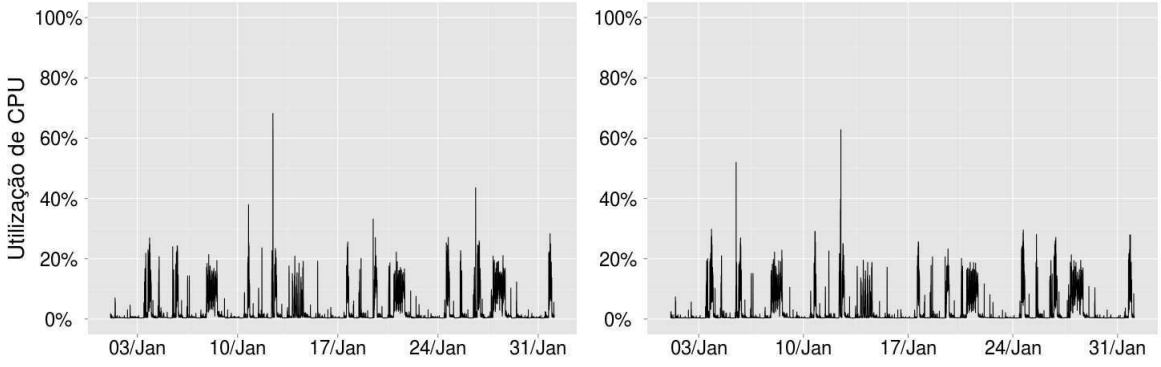


Figura A.85: Rastros 169 e 170 de curta duração.

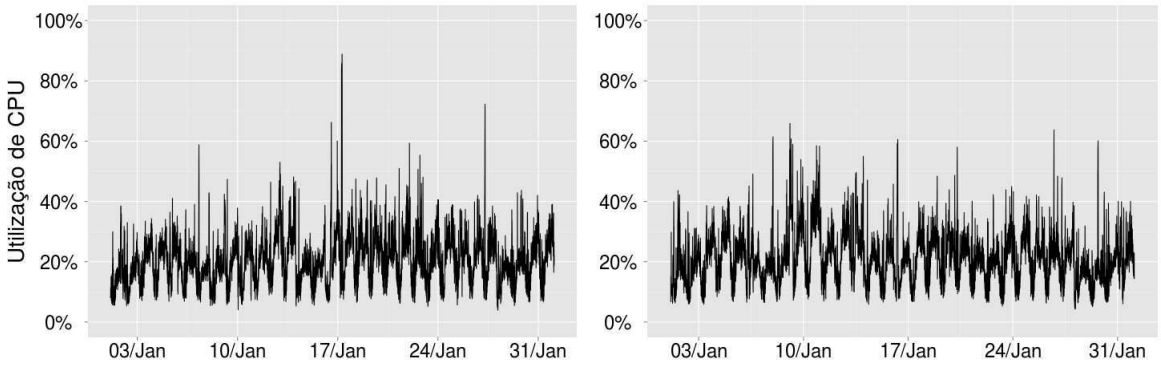


Figura A.86: Rastros 171 e 172 de curta duração.

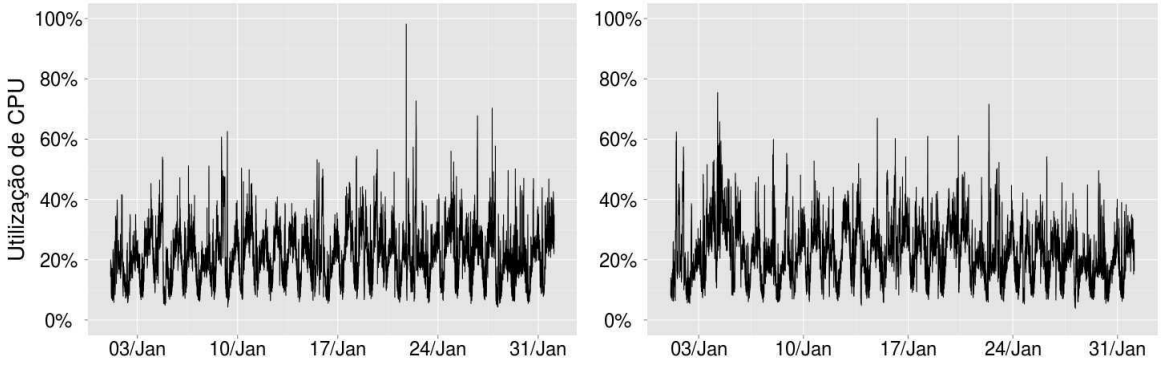


Figura A.87: Rastros 173 e 174 de curta duração.

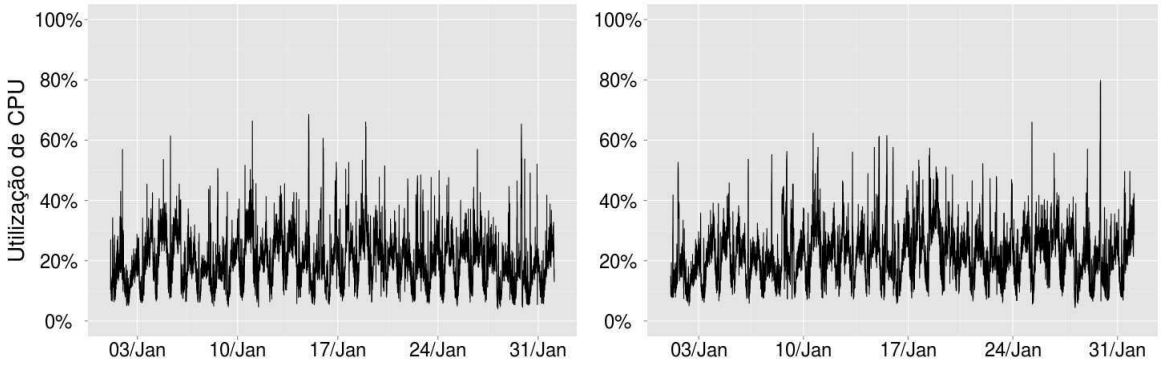


Figura A.88: Rastros 175 e 176 de curta duração.

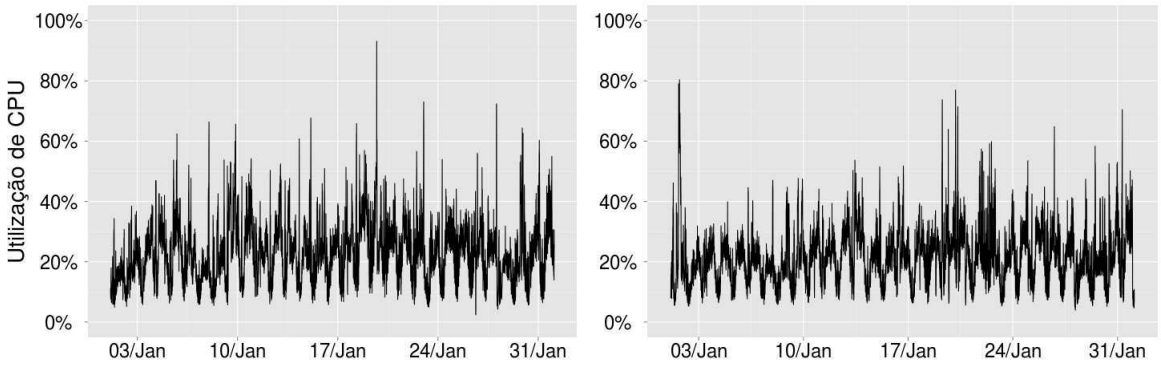


Figura A.89: Rastros 177 e 178 de curta duração.

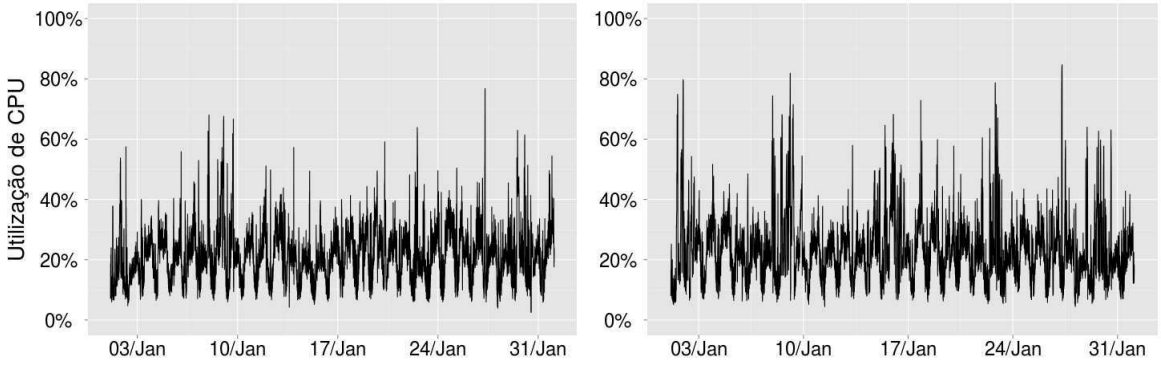


Figura A.90: Rastros 179 e 180 de curta duração.

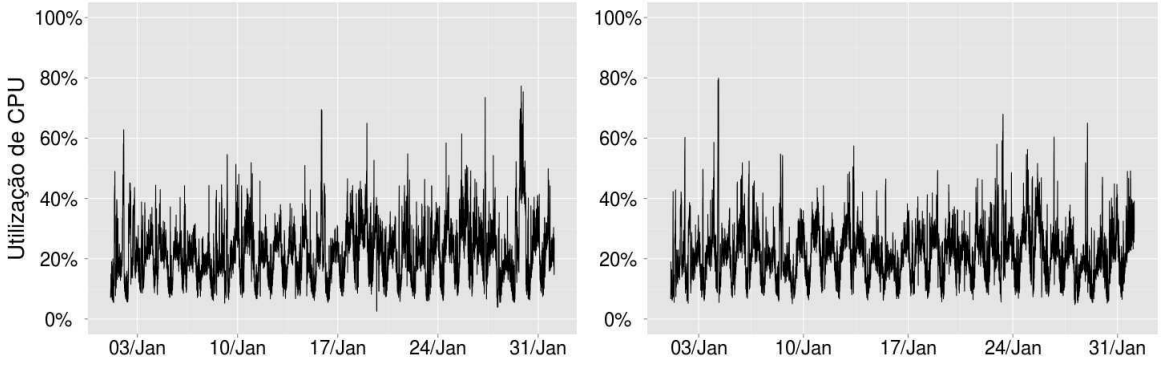


Figura A.91: Rastros 181 e 182 de curta duração.

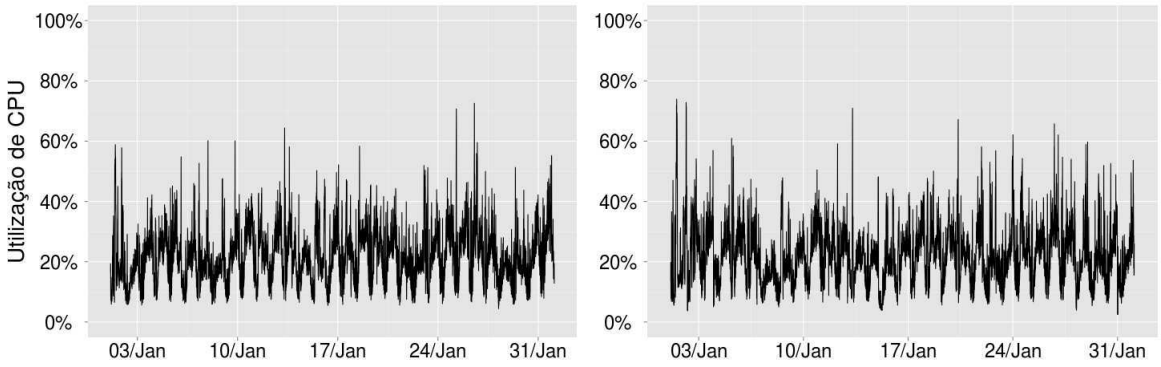


Figura A.92: Rastros 183 e 184 de curta duração.

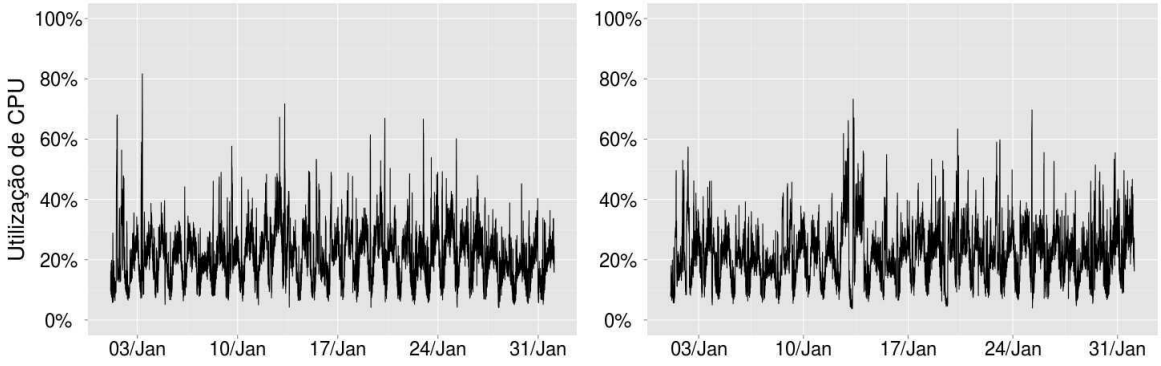


Figura A.93: Rastros 185 e 186 de curta duração.

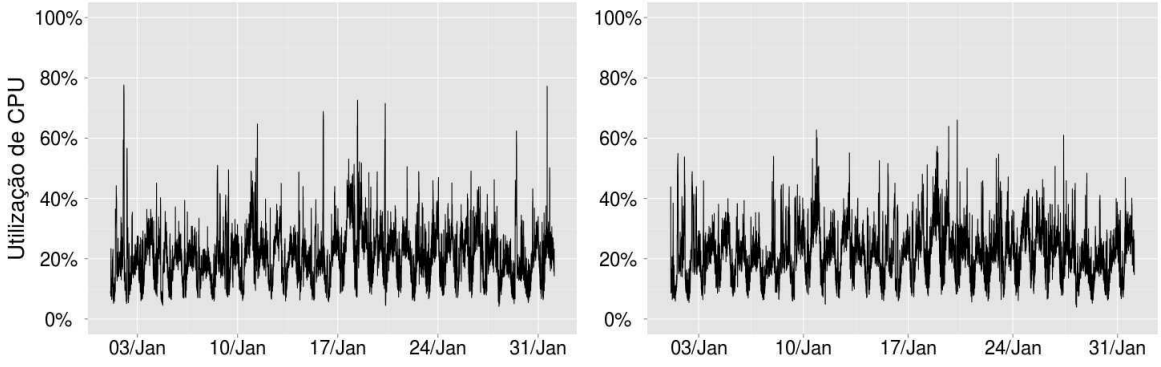


Figura A.94: Rastros 187 e 188 de curta duração.

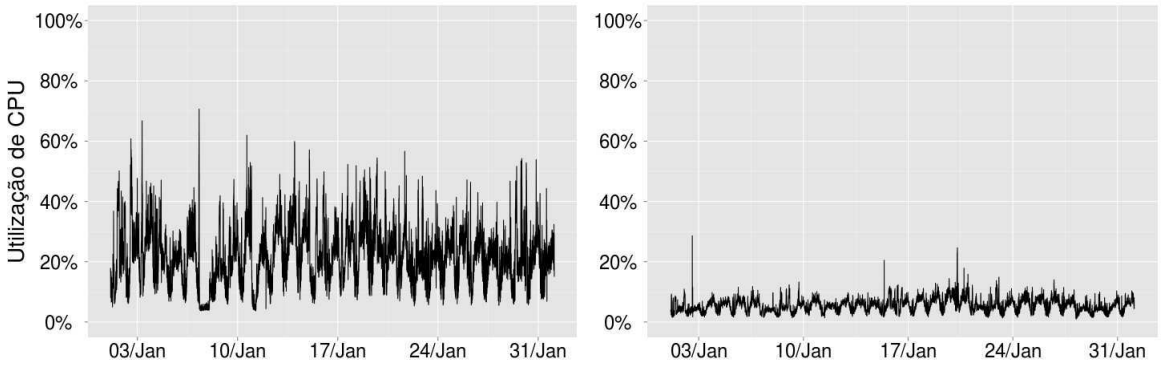


Figura A.95: Rastros 189 e 190 de curta duração.

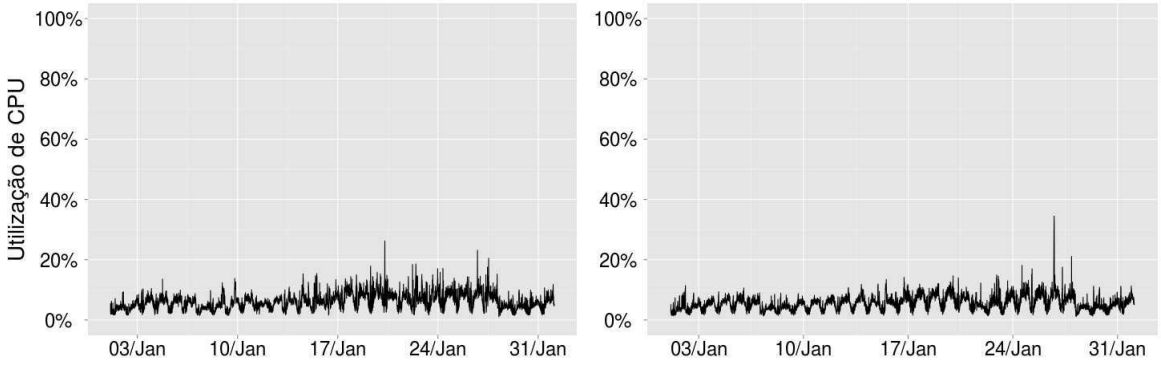


Figura A.96: Rastros 191 e 192 de curta duração.

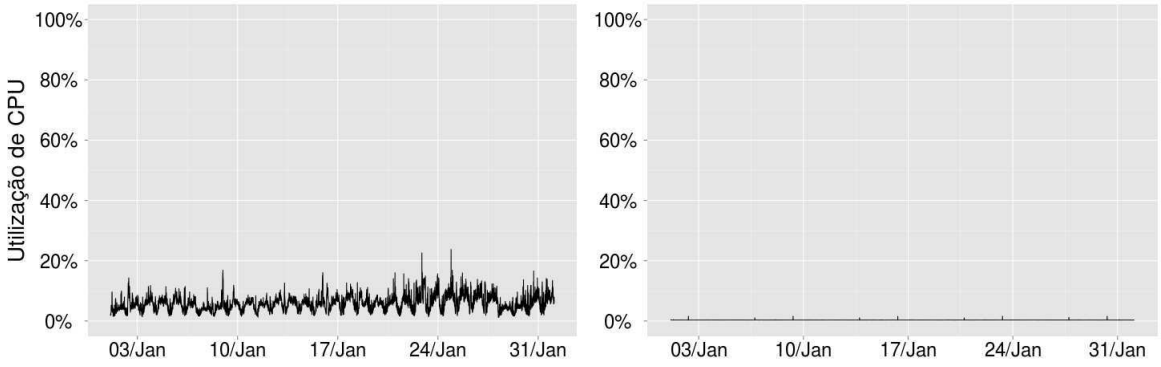


Figura A.97: Rastros 193 e 194 de curta duração.

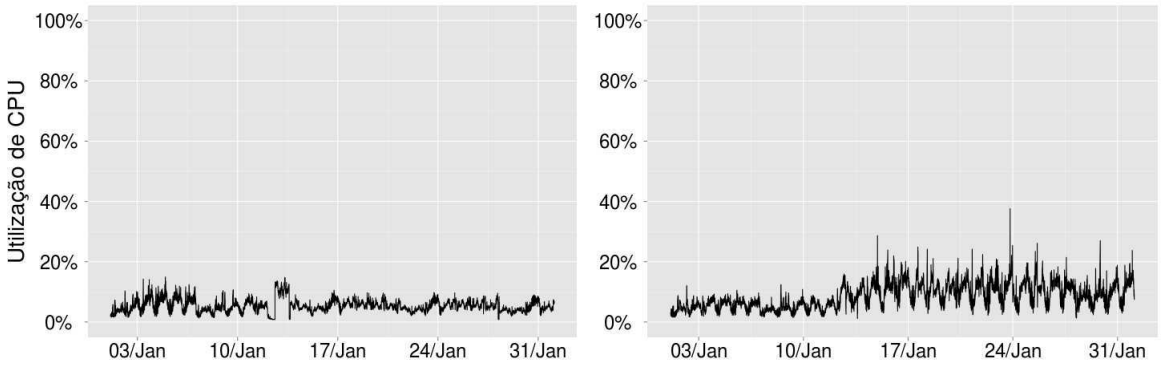


Figura A.98: Rastros 195 e 196 de curta duração.

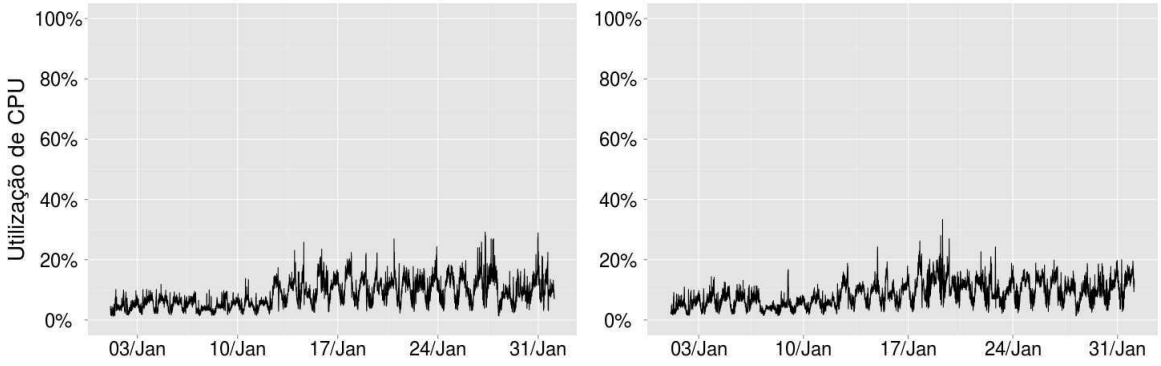


Figura A.99: Rastros 197 e 198 de curta duração.

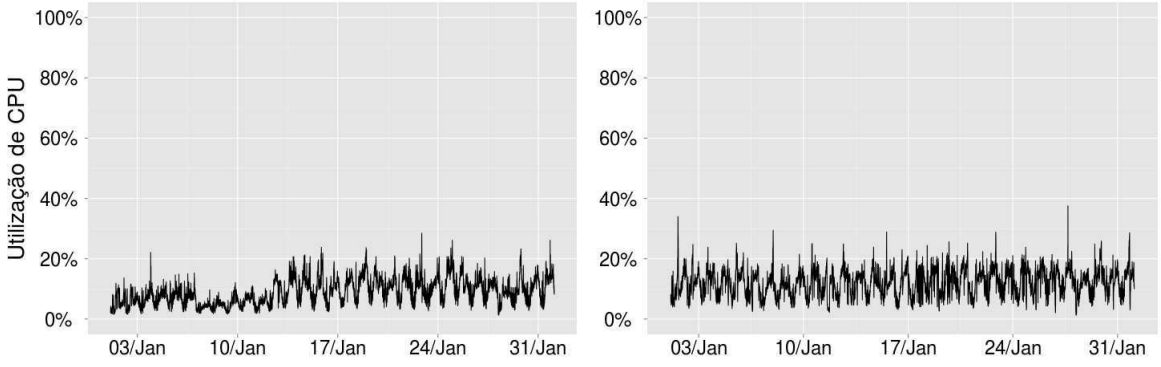


Figura A.100: Rastros 199 e 200 de curta duração.

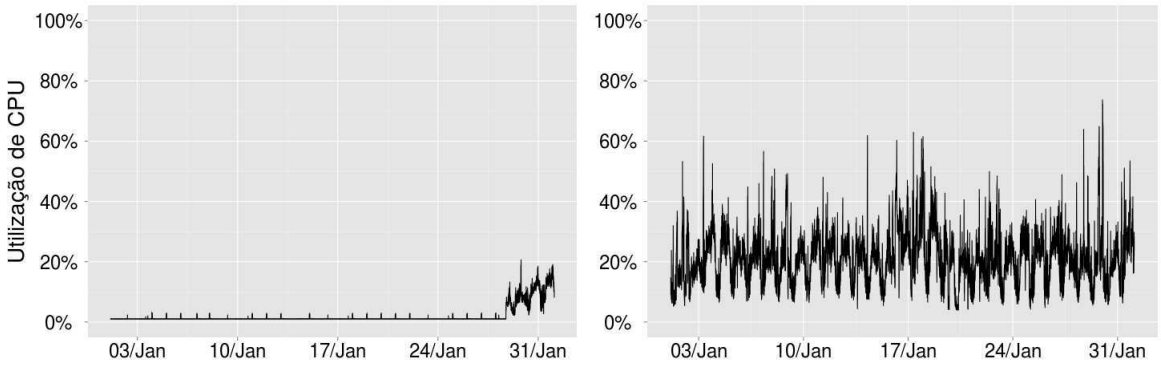


Figura A.101: Rastros 201 e 202 de curta duração.

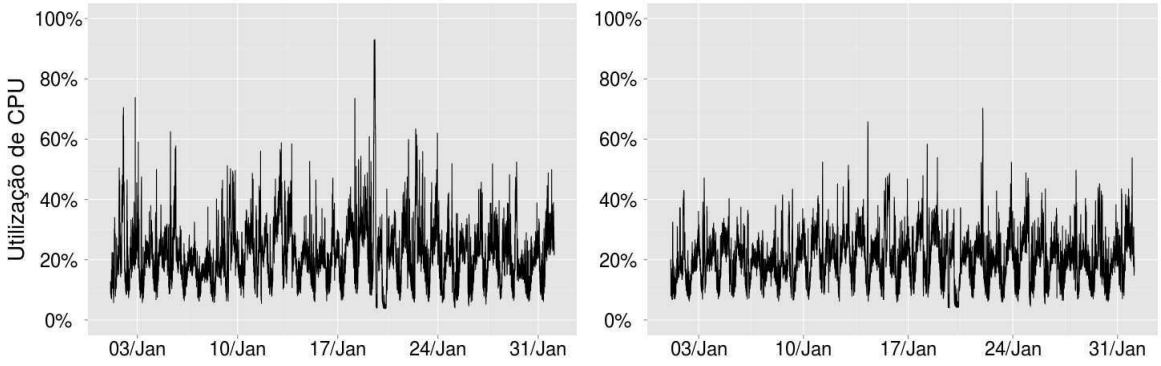


Figura A.102: Rastros 203 e 204 de curta duração.

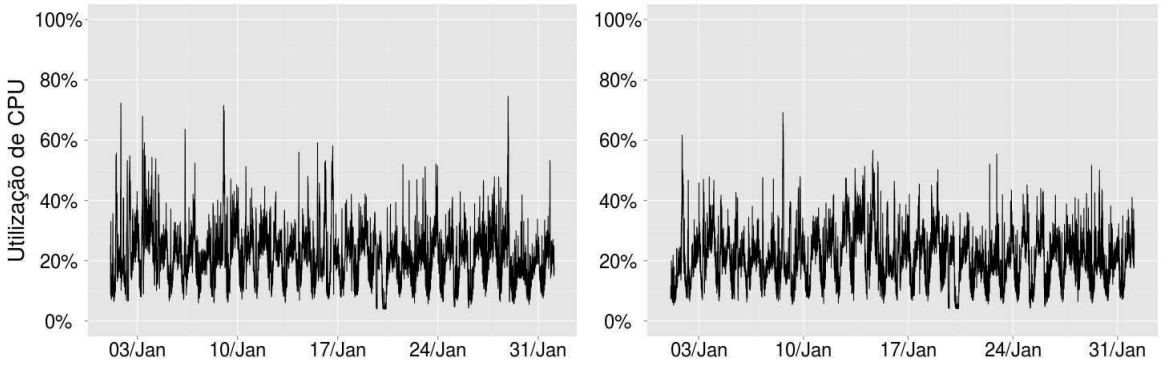


Figura A.103: Rastros 205 e 206 de curta duração.

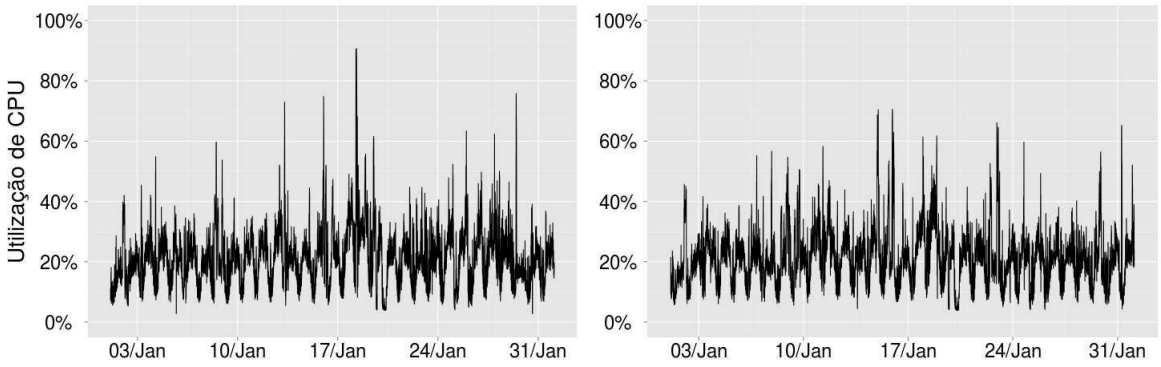


Figura A.104: Rastros 207 e 208 de curta duração.

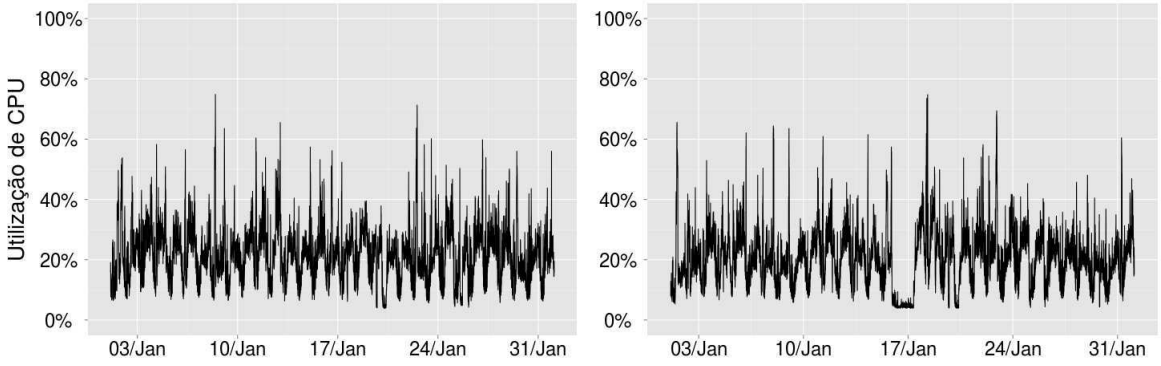


Figura A.105: Rastros 209 e 210 de curta duração.

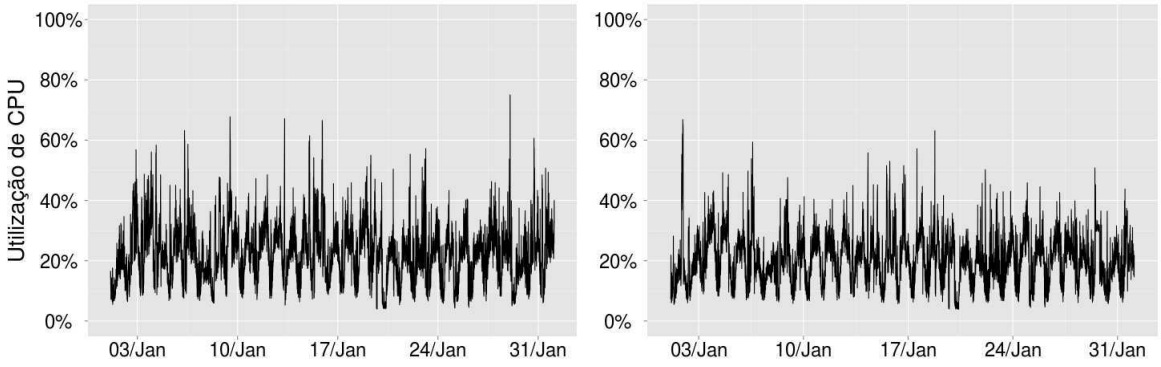


Figura A.106: Rastros 211 e 212 de curta duração.

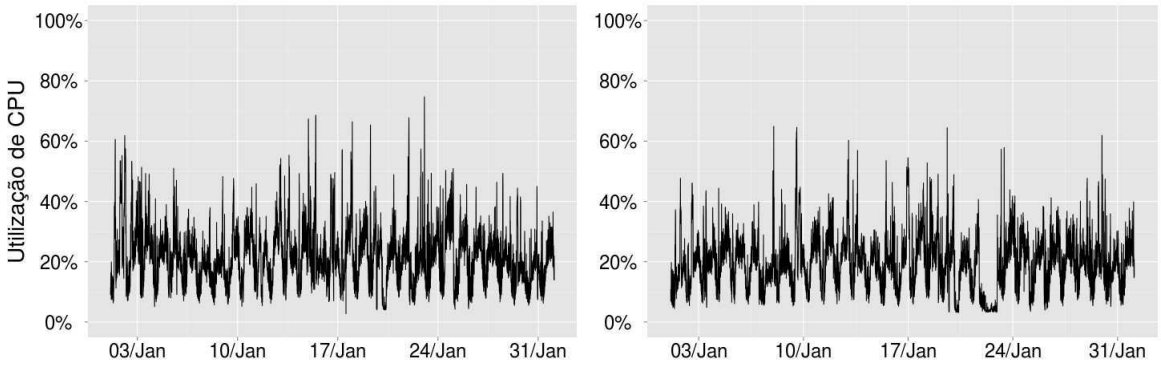


Figura A.107: Rastros 213 e 214 de curta duração.

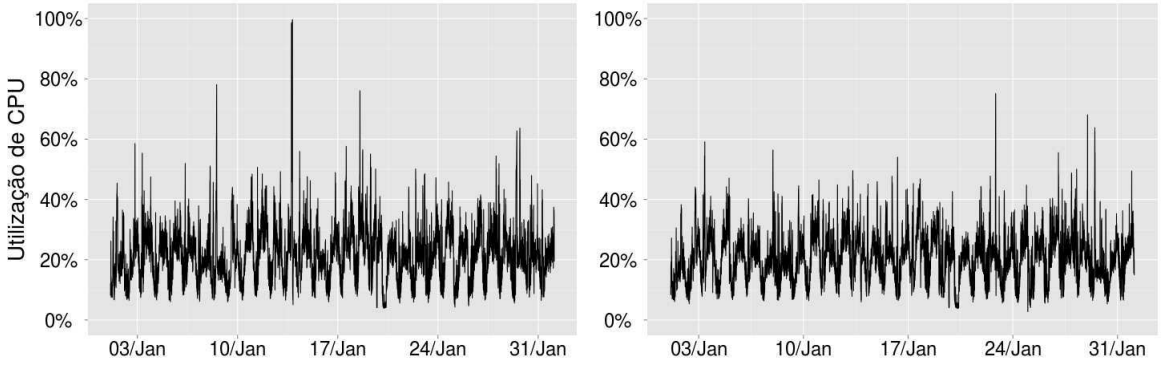


Figura A.108: Rastros 215 e 216 de curta duração.

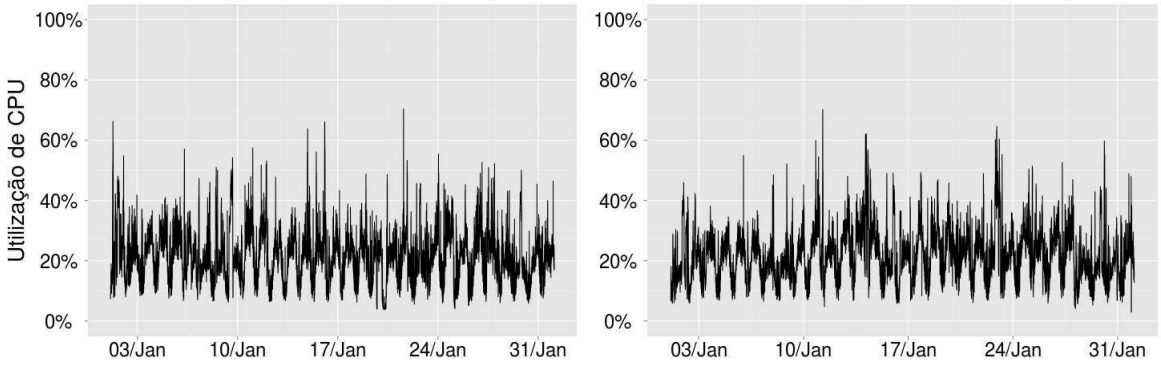


Figura A.109: Rastros 217 e 218 de curta duração.

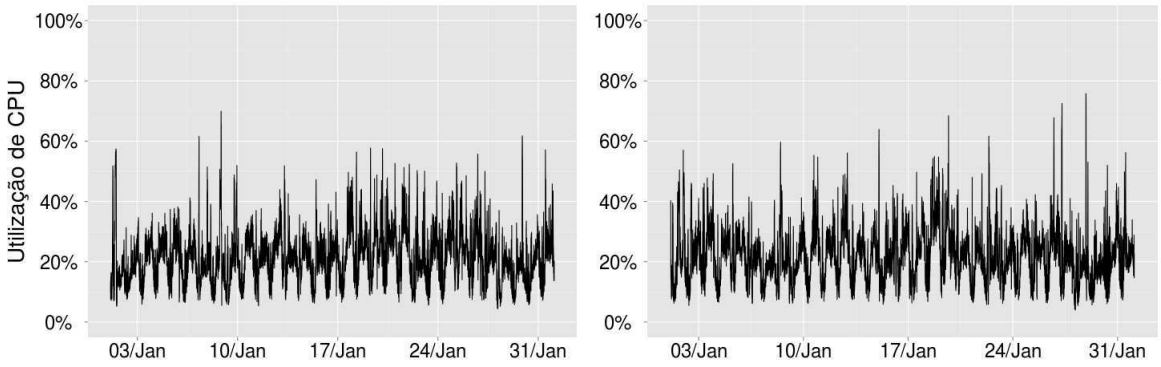


Figura A.110: Rastros 219 e 220 de curta duração.

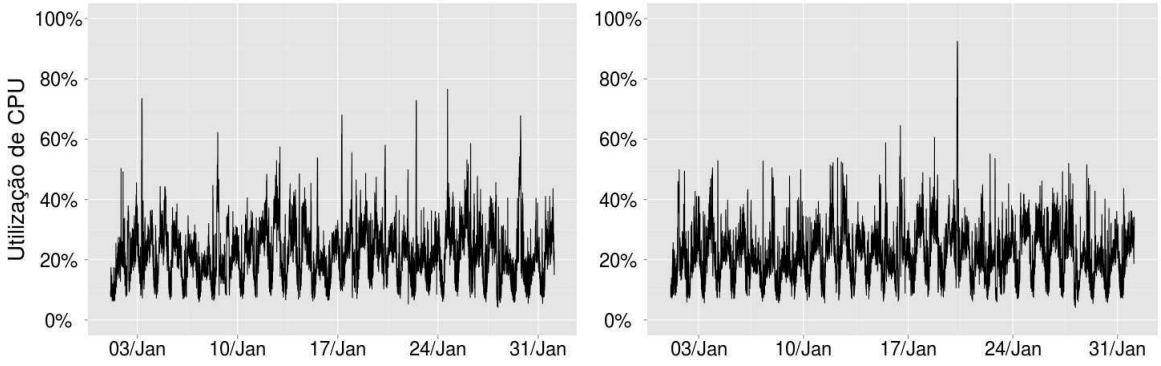


Figura A.111: Rastros 221 e 222 de curta duração.

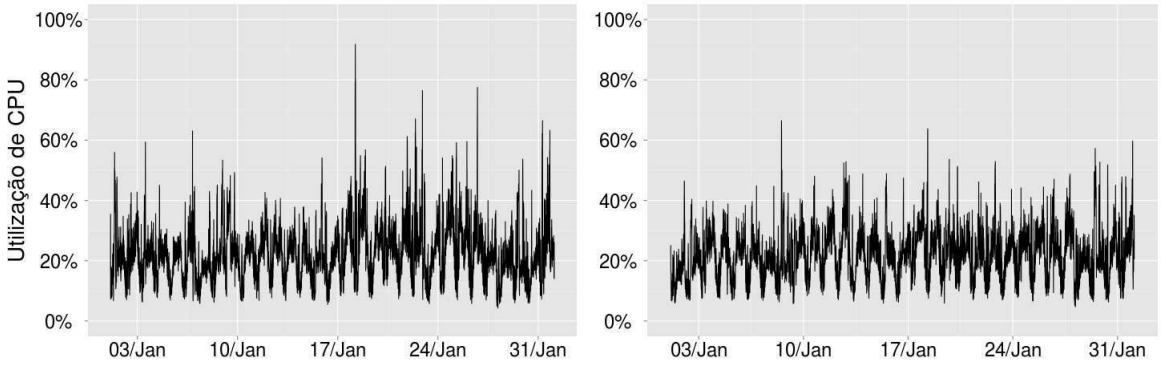


Figura A.112: Rastros 223 e 224 de curta duração.

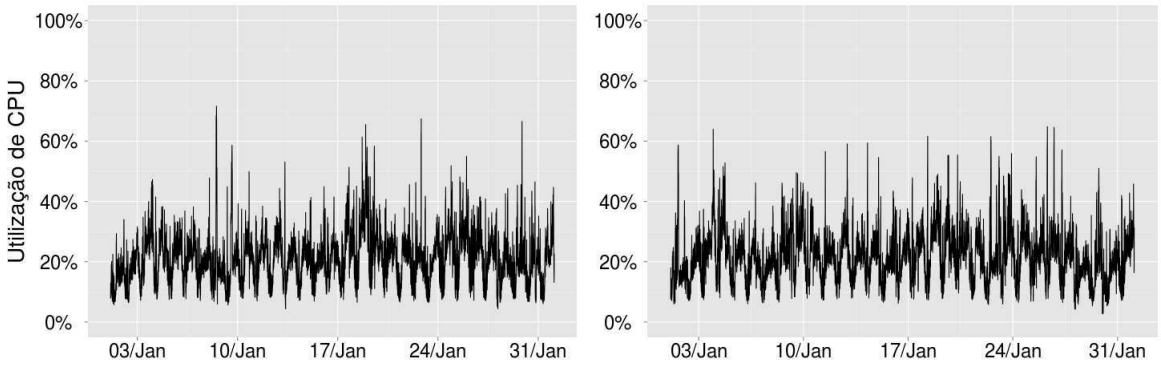


Figura A.113: Rastros 225 e 226 de curta duração.

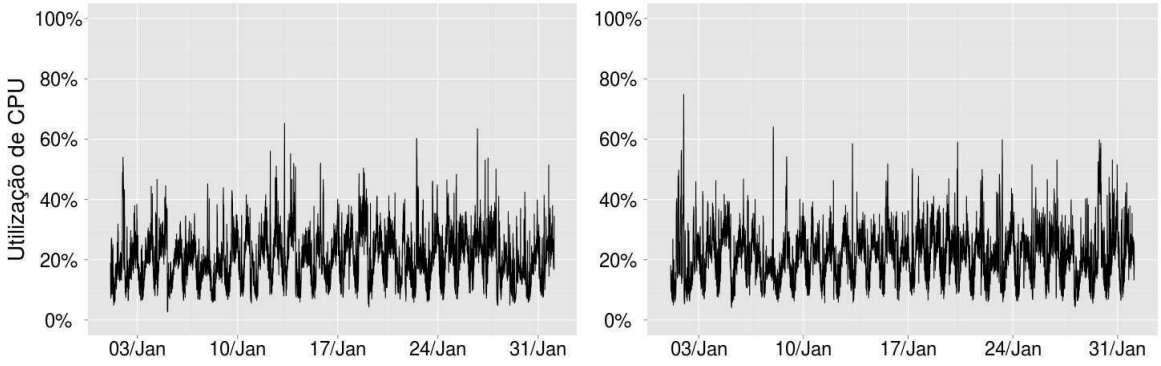


Figura A.114: Rastros 227 e 228 de curta duração.

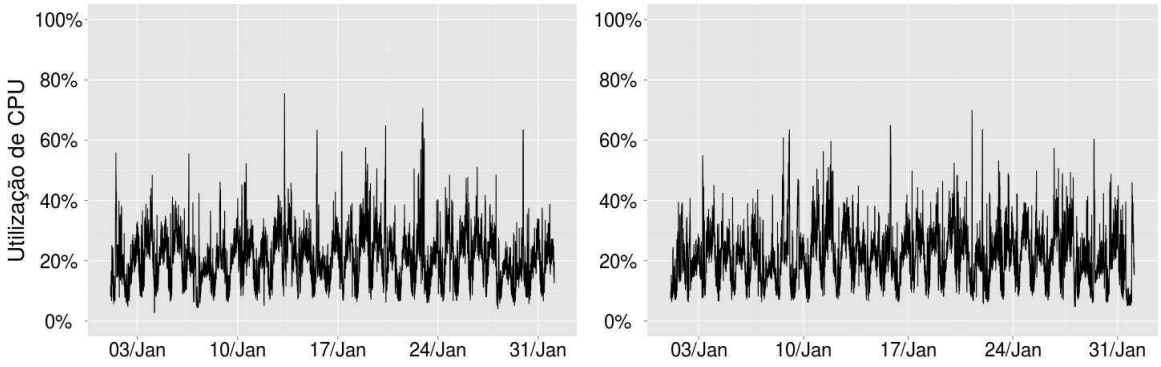


Figura A.115: Rastros 229 e 230 de curta duração.

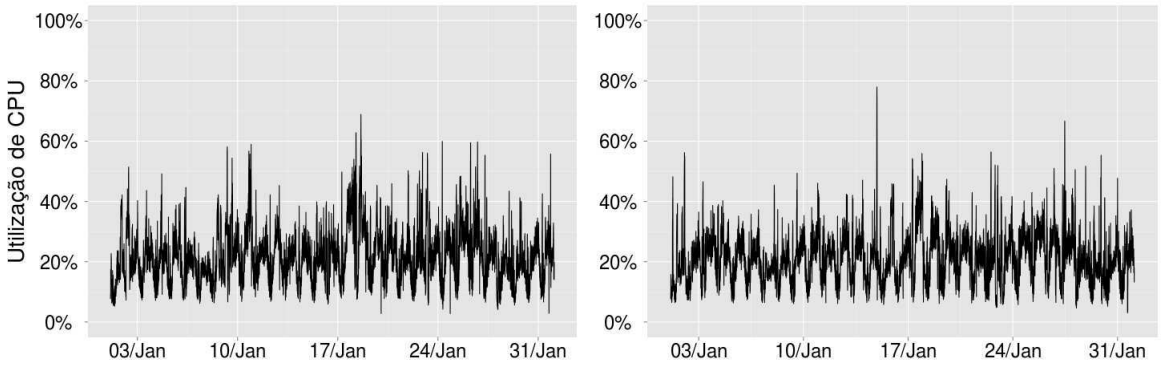


Figura A.116: Rastros 231 e 232 de curta duração.

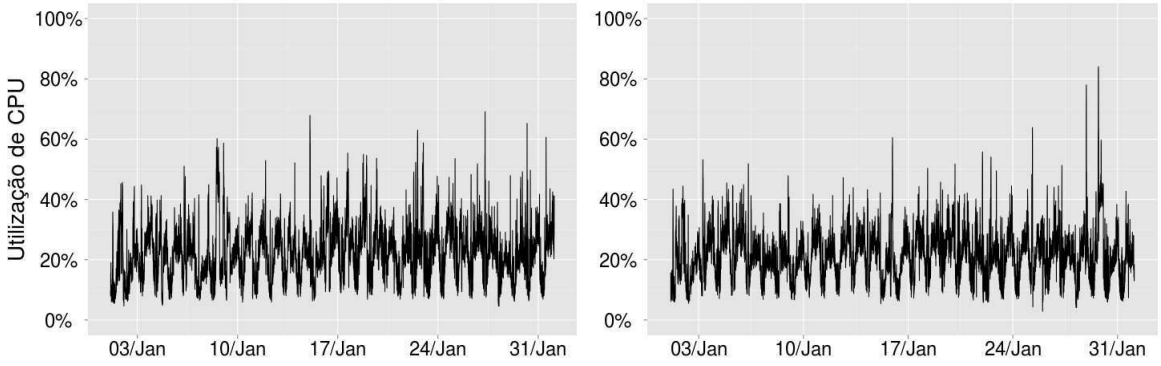


Figura A.117: Rastros 233 e 234 de curta duração.

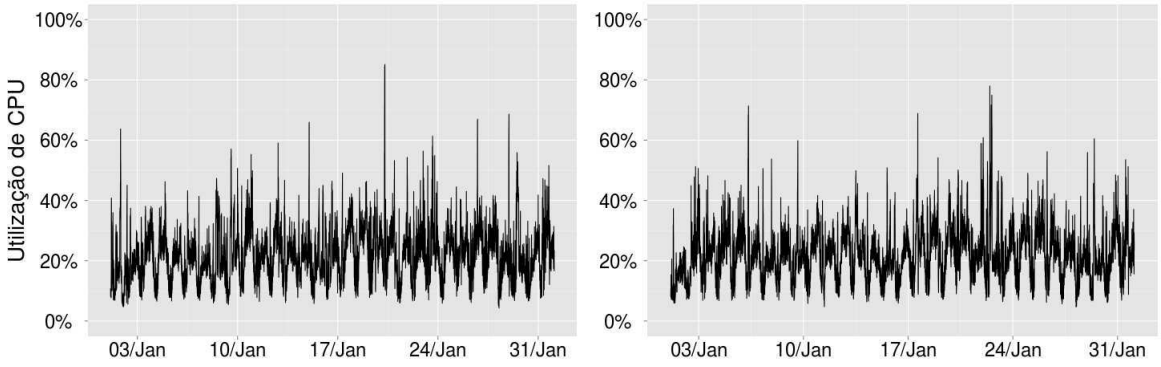


Figura A.118: Rastros 235 e 236 de curta duração.

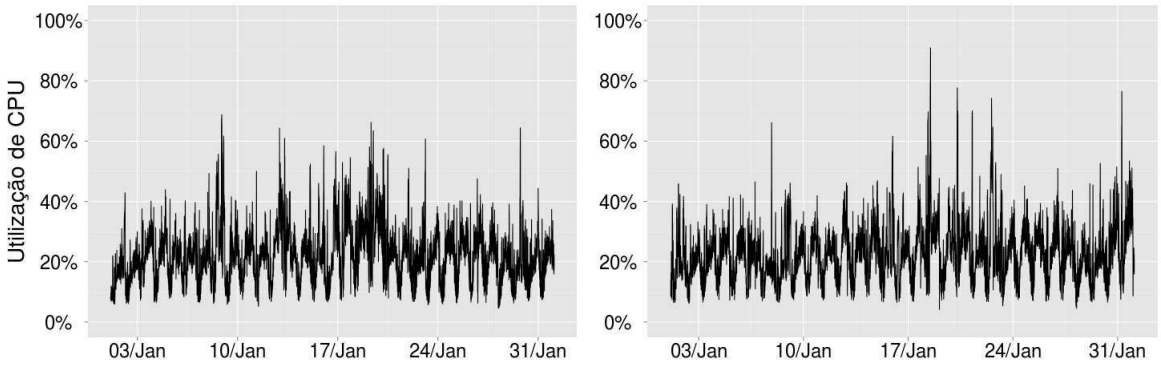


Figura A.119: Rastros 237 e 238 de curta duração.

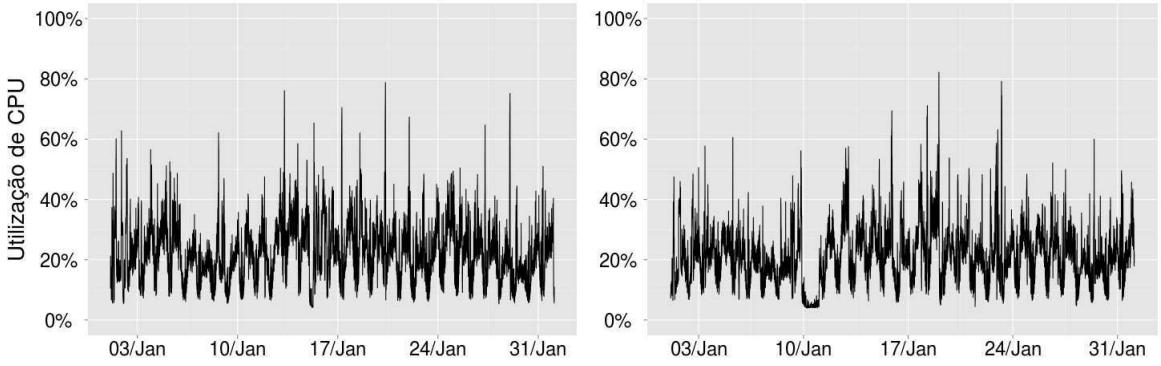


Figura A.120: Rastros 239 e 240 de curta duração.

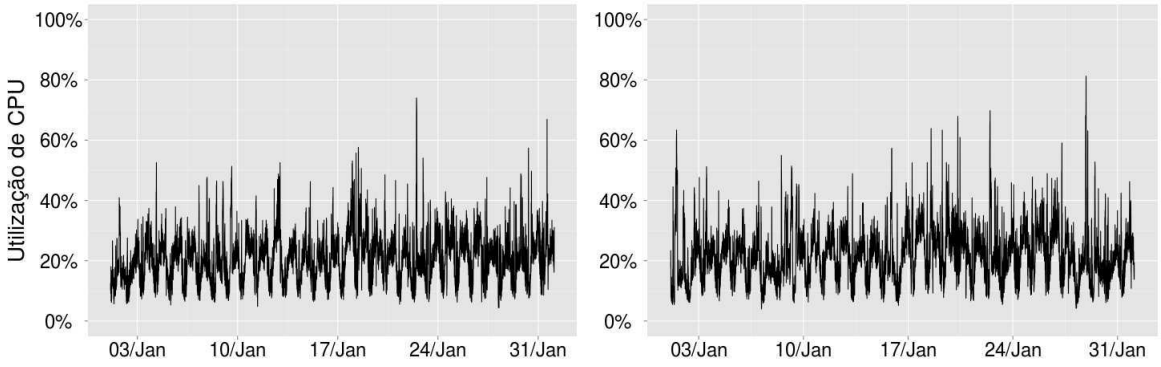


Figura A.121: Rastros 241 e 242 de curta duração.

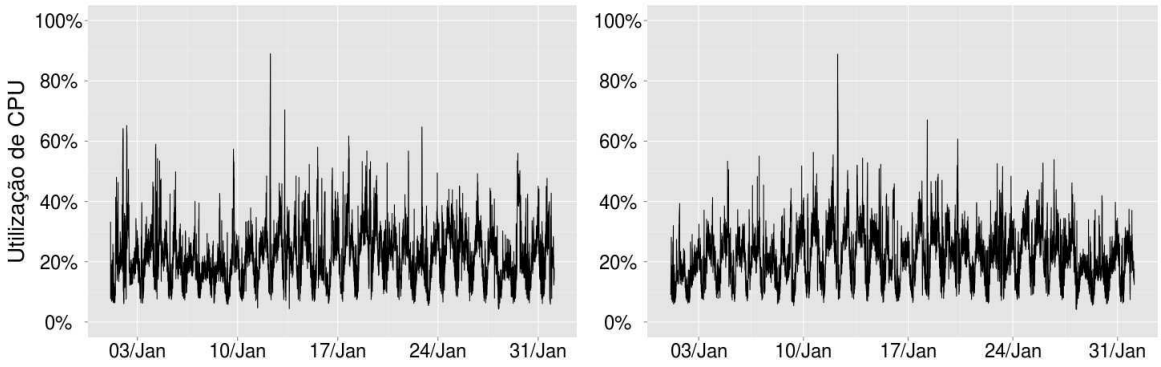


Figura A.122: Rastros 243 e 244 de curta duração.

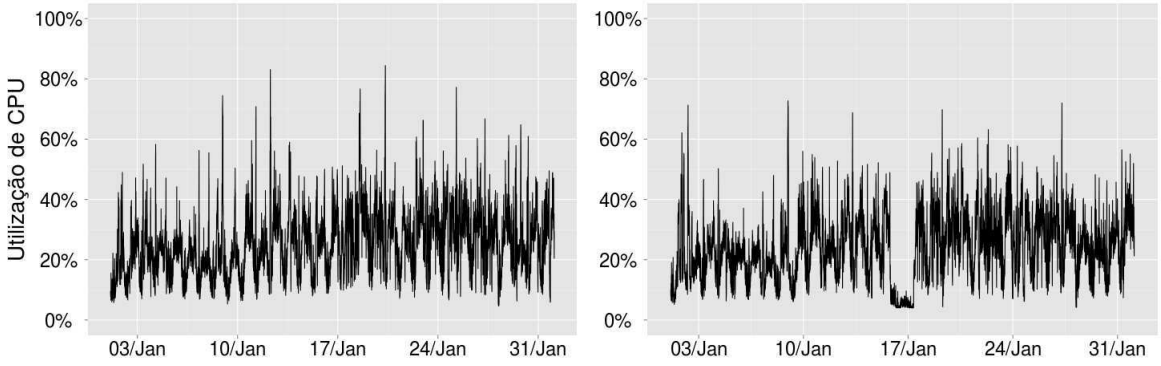


Figura A.123: Rastros 245 e 246 de curta duração.

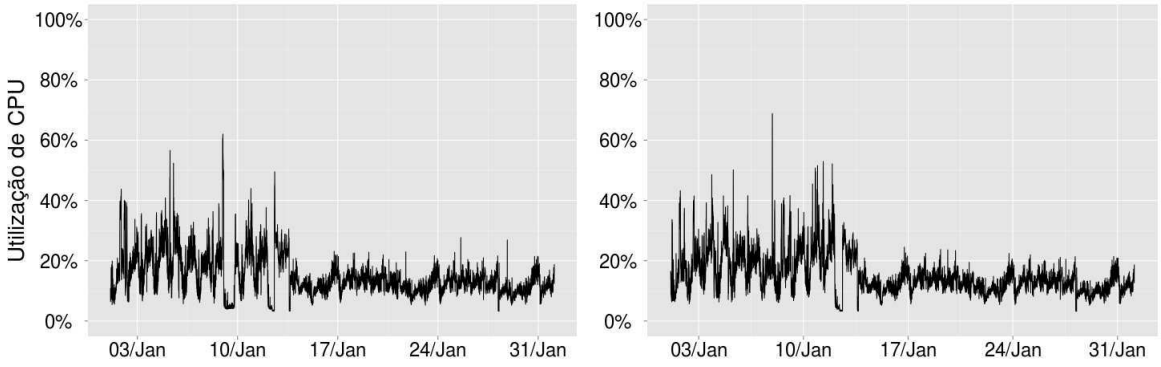


Figura A.124: Rastros 247 e 248 de curta duração.

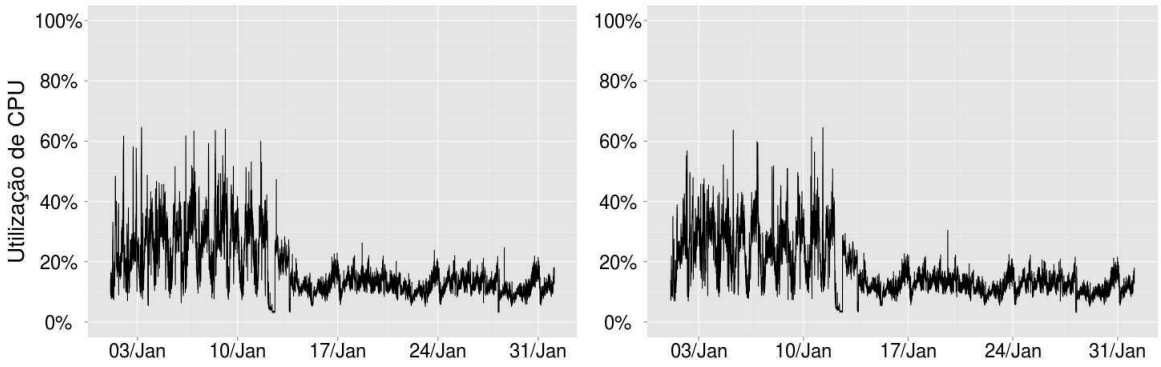


Figura A.125: Rastros 249 e 250 de curta duração.

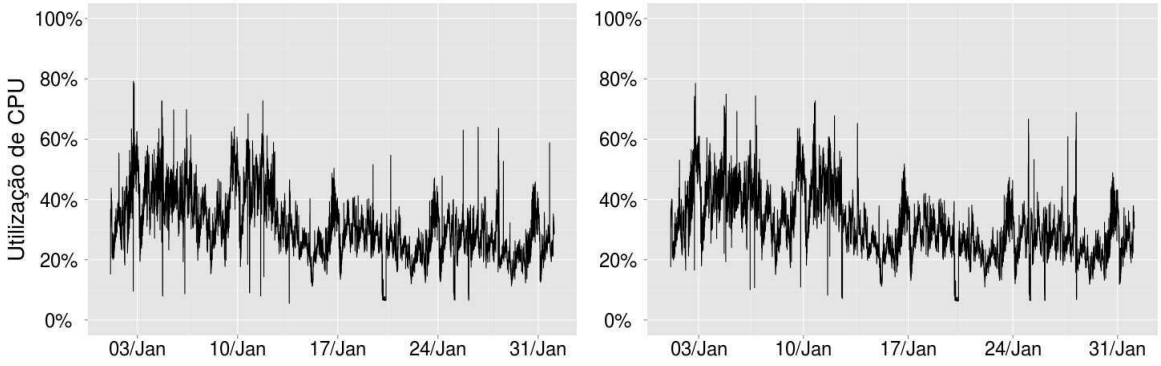


Figura A.126: Rastros 251 e 252 de curta duração.

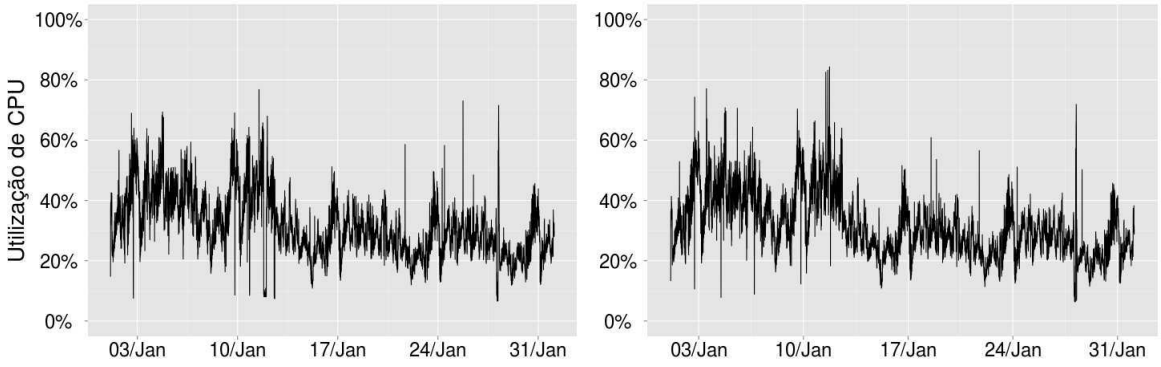


Figura A.127: Rastros 253 e 254 de curta duração.

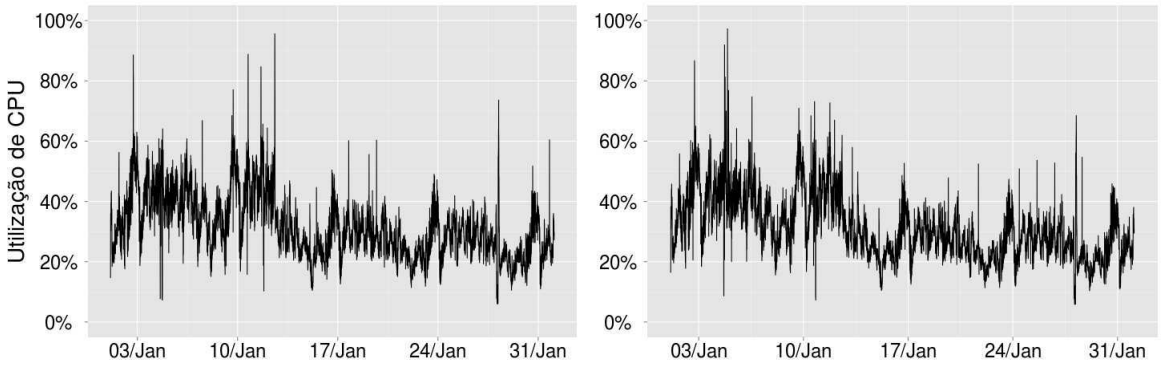


Figura A.128: Rastros 255 e 256 de curta duração.

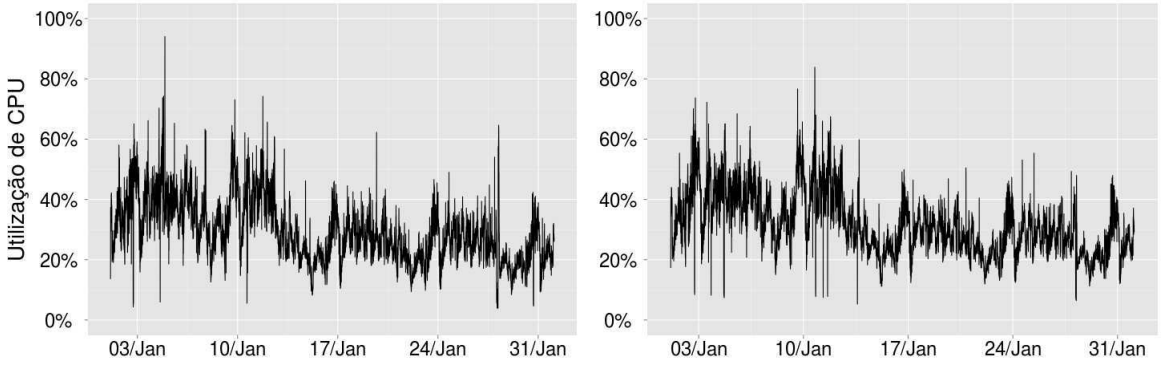


Figura A.129: Rastros 257 e 258 de curta duração.

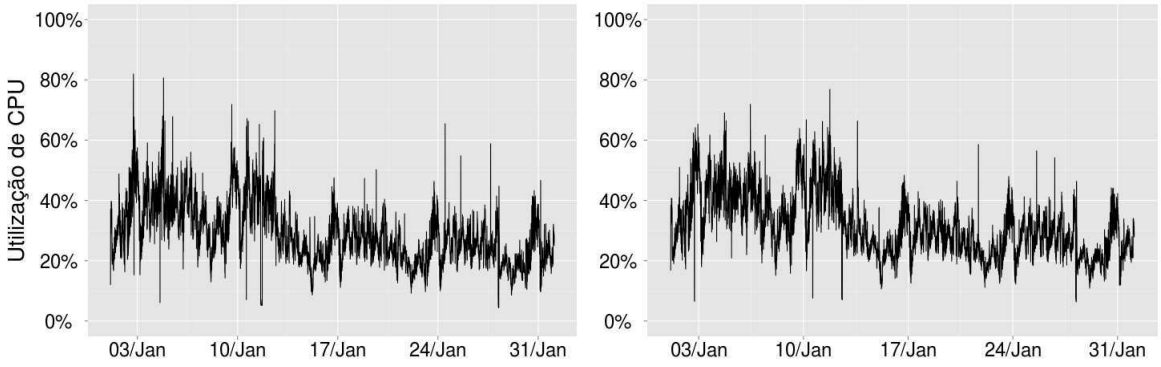


Figura A.130: Rastros 259 e 260 de curta duração.

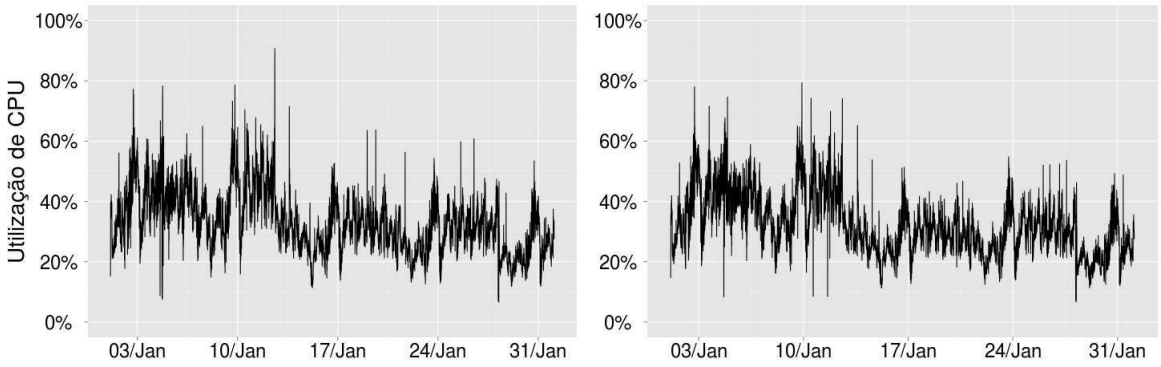


Figura A.131: Rastros 261 e 262 de curta duração.

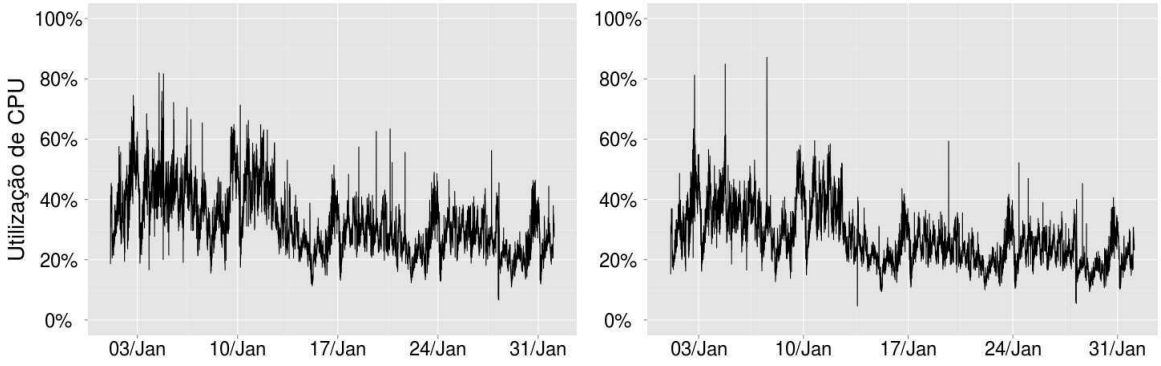


Figura A.132: Rastros 263 e 264 de curta duração.

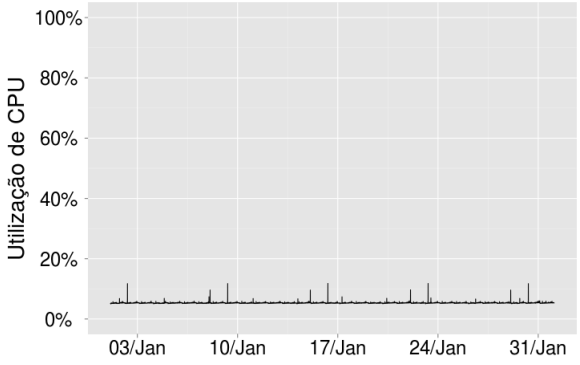


Figura A.133: Rastro 265 de curta duração.

A.2 Cargas de Trabalho de Longa Duração

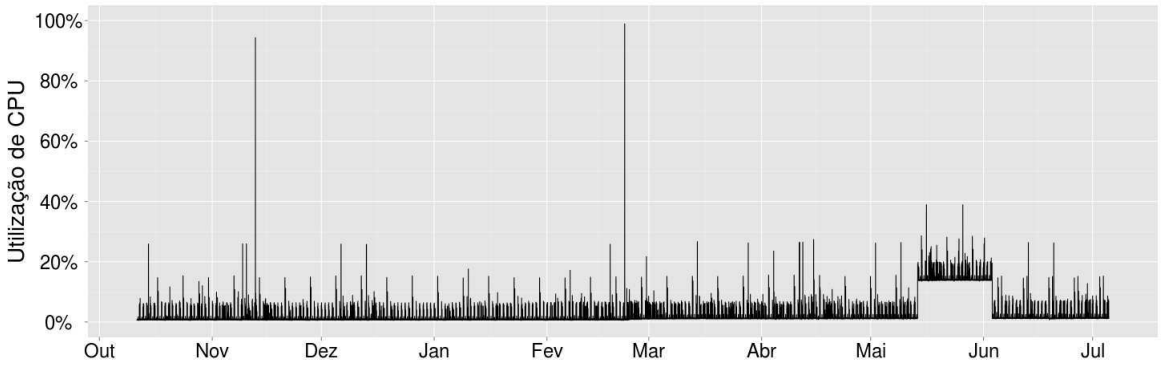


Figura A.134: Rastro 1 de longa duração.

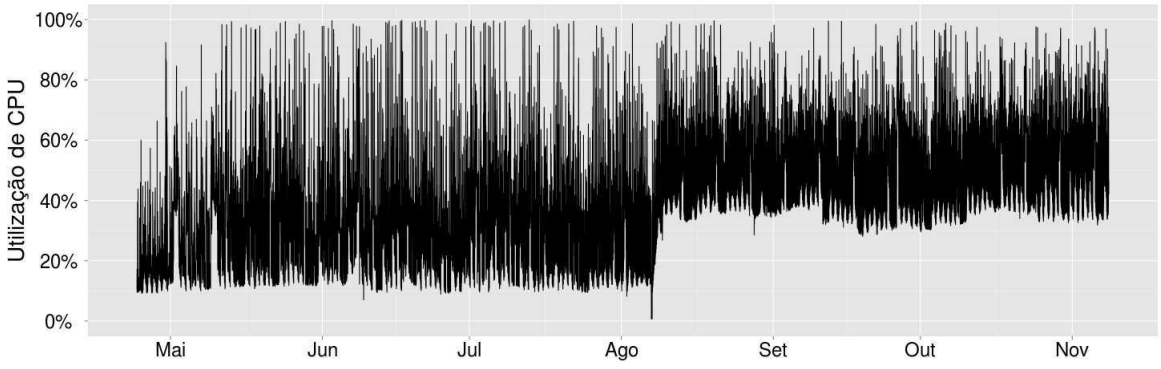


Figura A.135: Rastro 2 de longa duração.

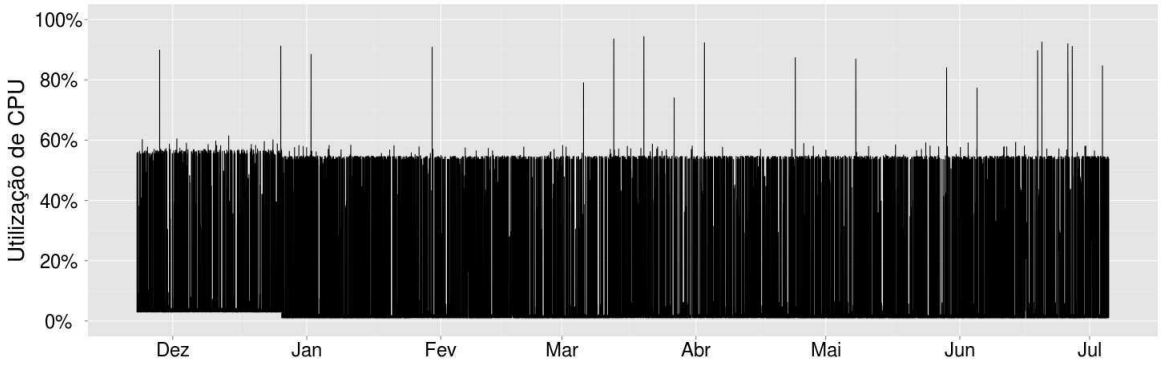


Figura A.136: Rastro 3 de longa duração.

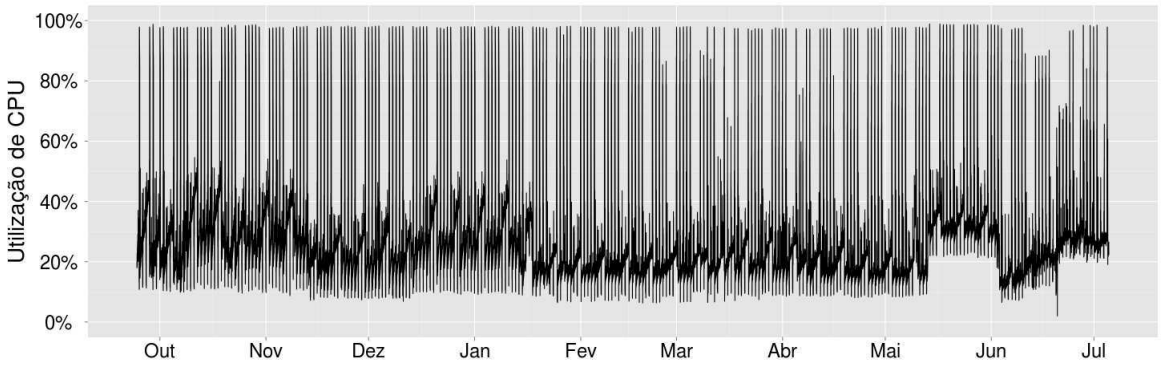


Figura A.137: Rastro 4 de longa duração.

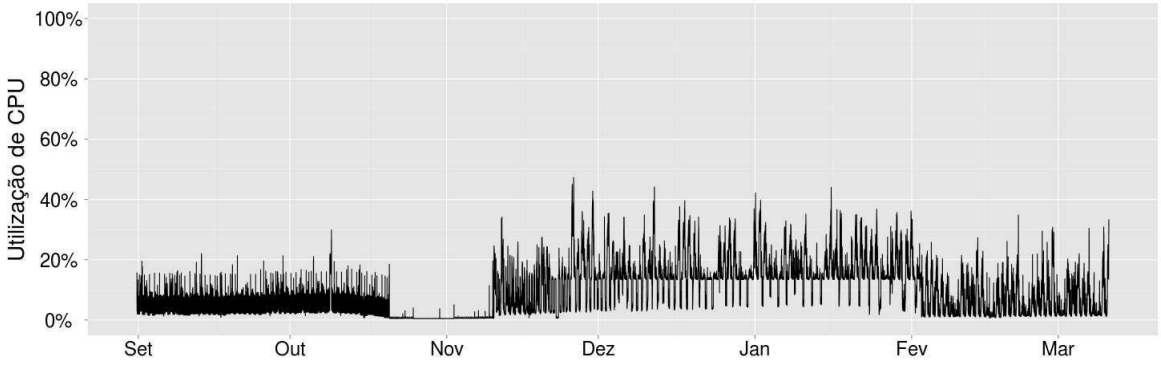


Figura A.138: Rastro 5 de longa duração.

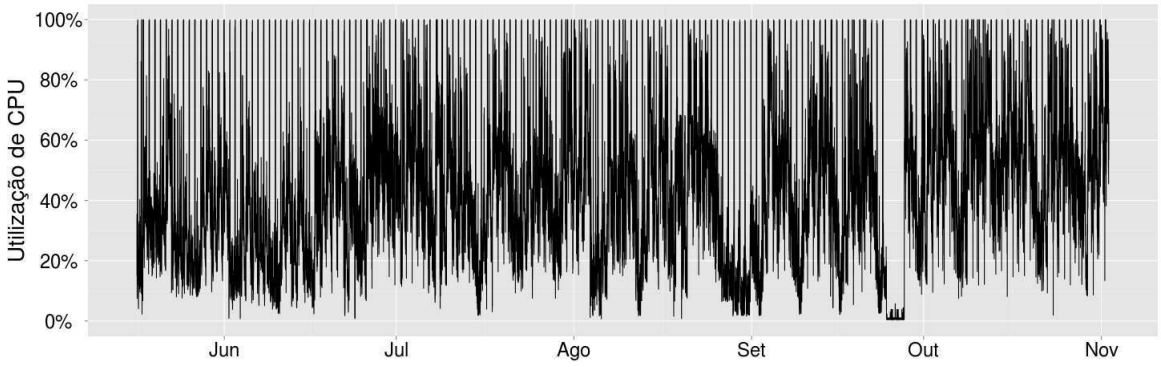


Figura A.139: Rastro 6 de longa duração.

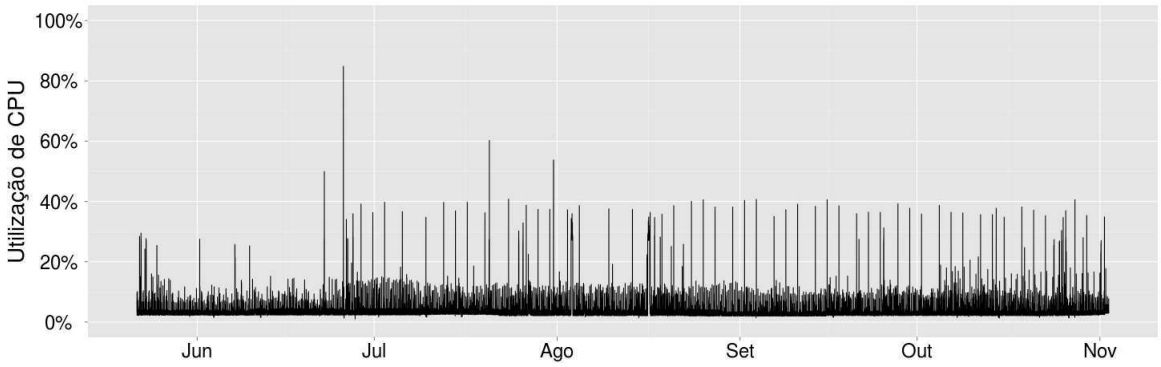


Figura A.140: Rastro 7 de longa duração.

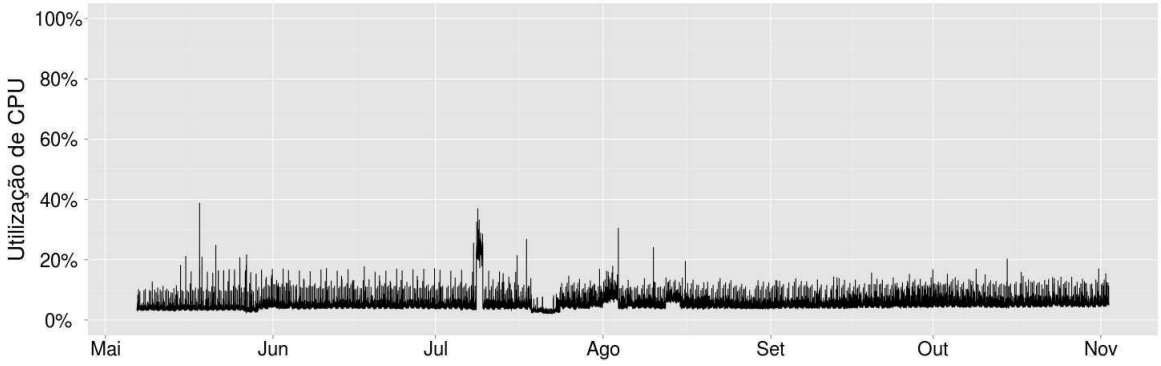


Figura A.141: Rastro 8 de longa duração.

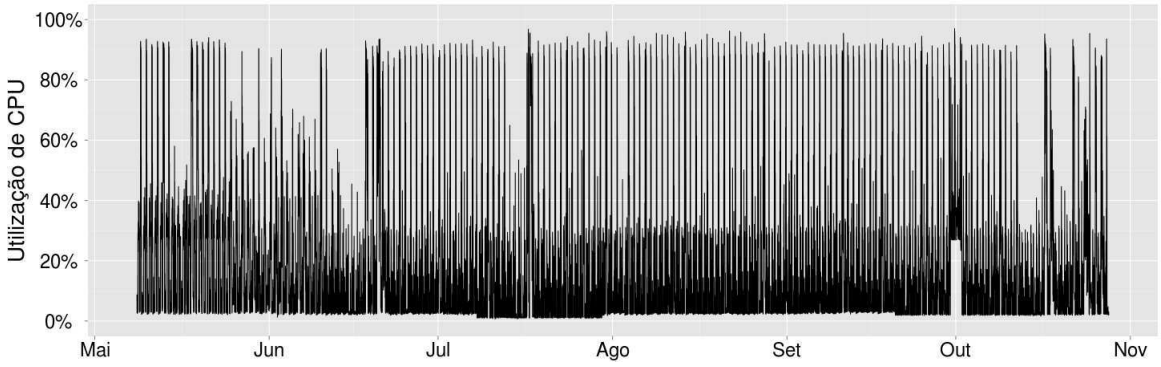


Figura A.142: Rastro 9 de longa duração.

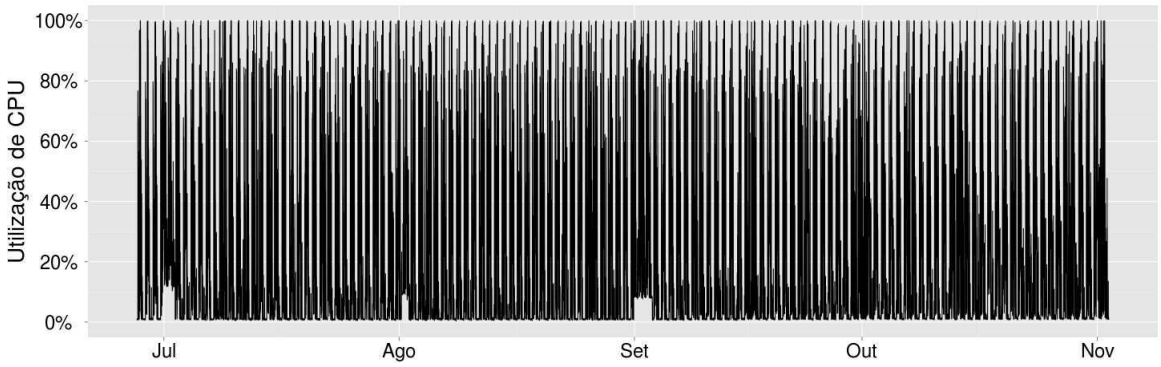


Figura A.143: Rastro 10 de longa duração.

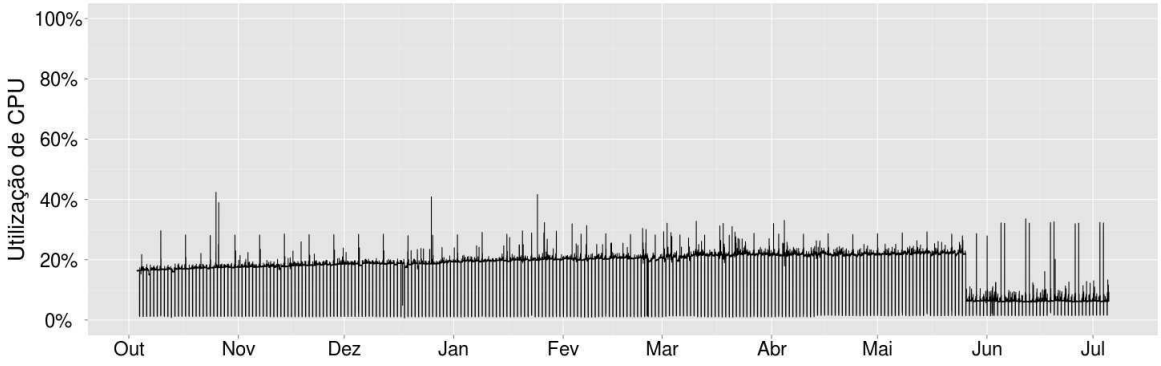


Figura A.144: Rastro 11 de longa duração.

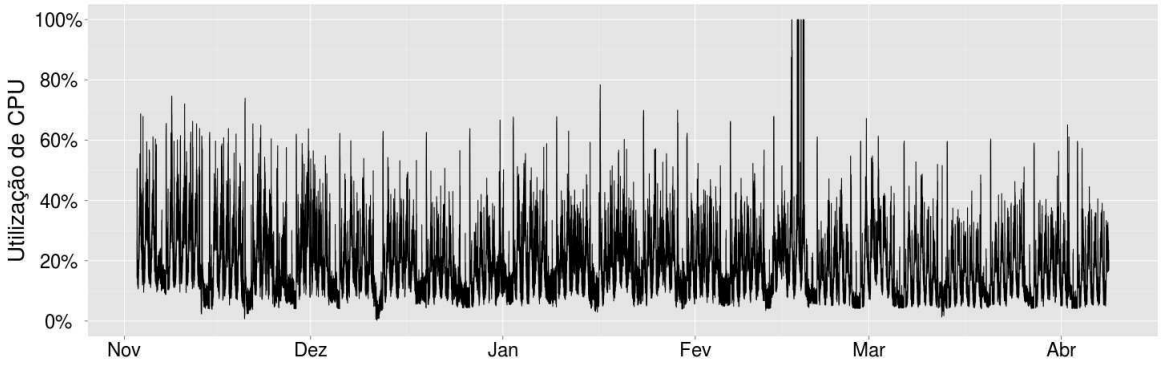


Figura A.145: Rastro 12 de longa duração.

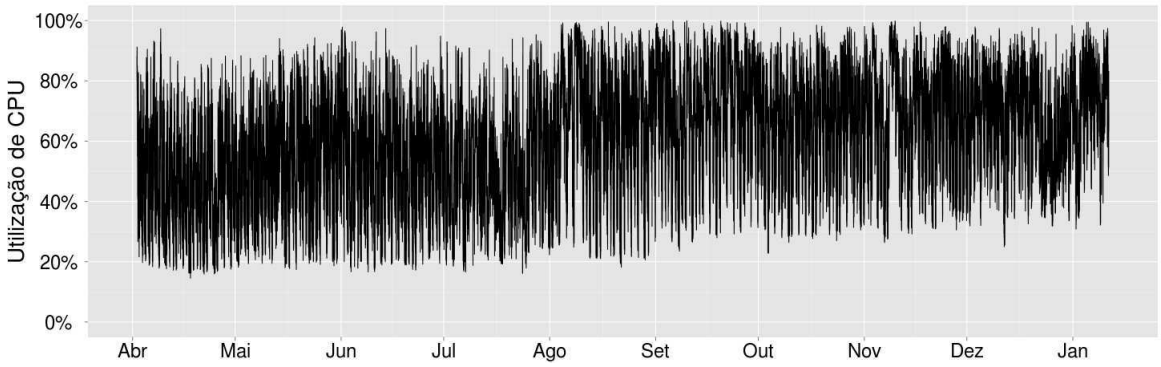


Figura A.146: Rastro 13 de longa duração.

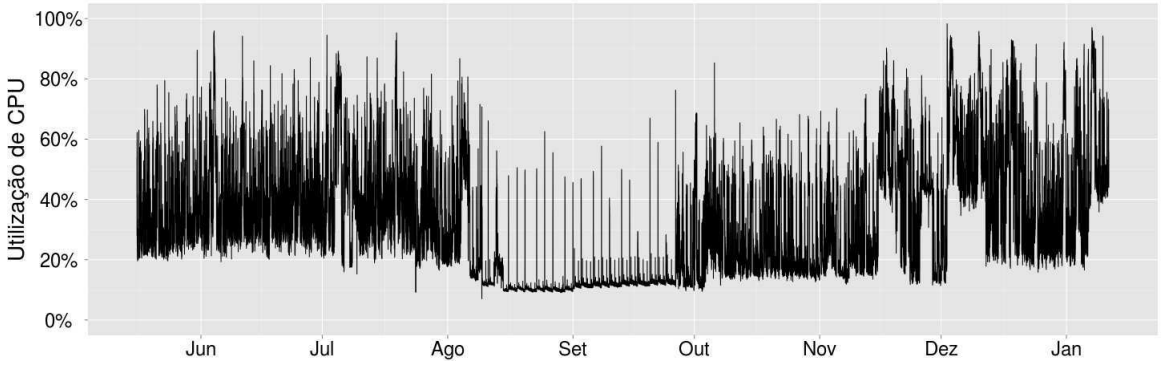


Figura A.147: Rastro 14 de longa duração.

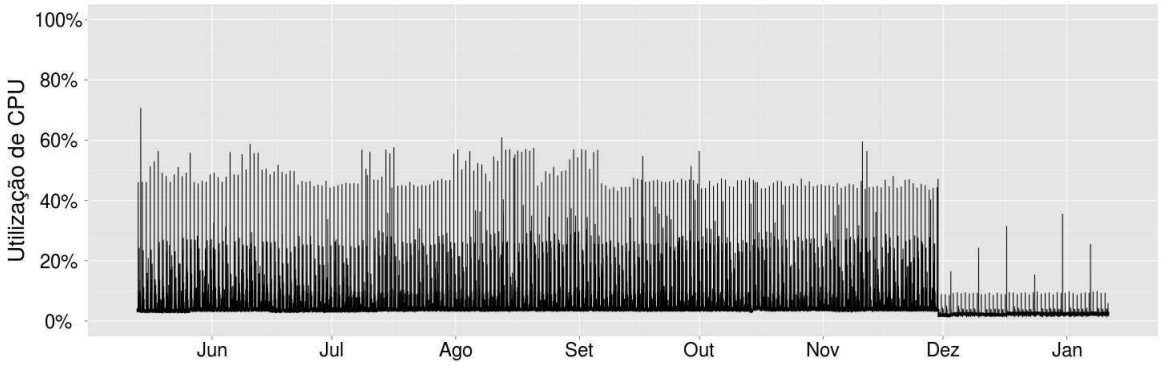


Figura A.148: Rastro 15 de longa duração.

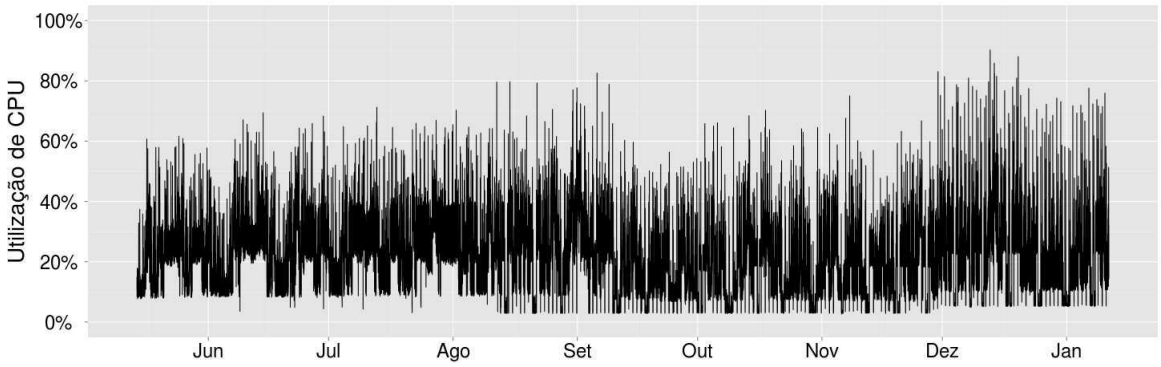


Figura A.149: Rastro 16 de longa duração.

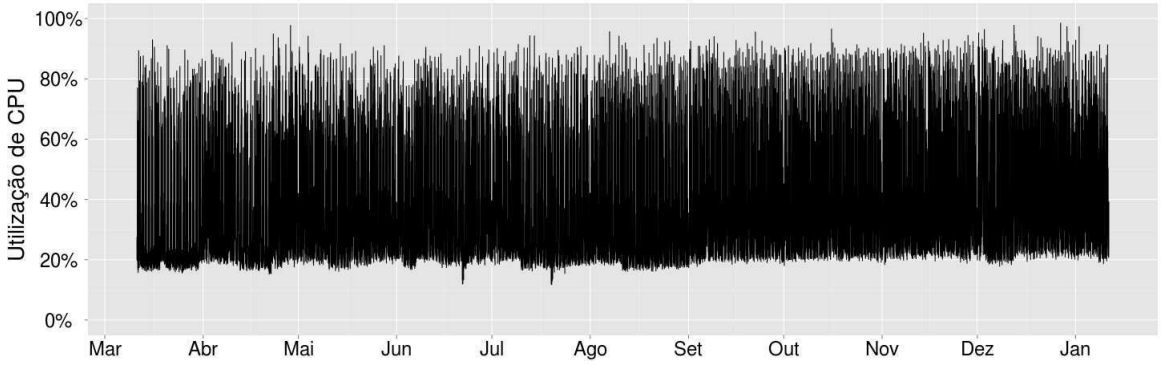


Figura A.150: Rastro 17 de longa duração.

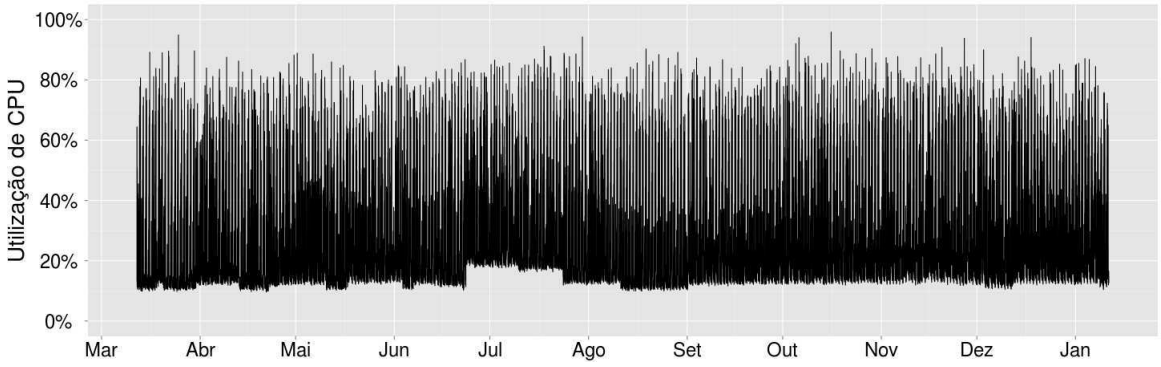


Figura A.151: Rastro 18 de longa duração.

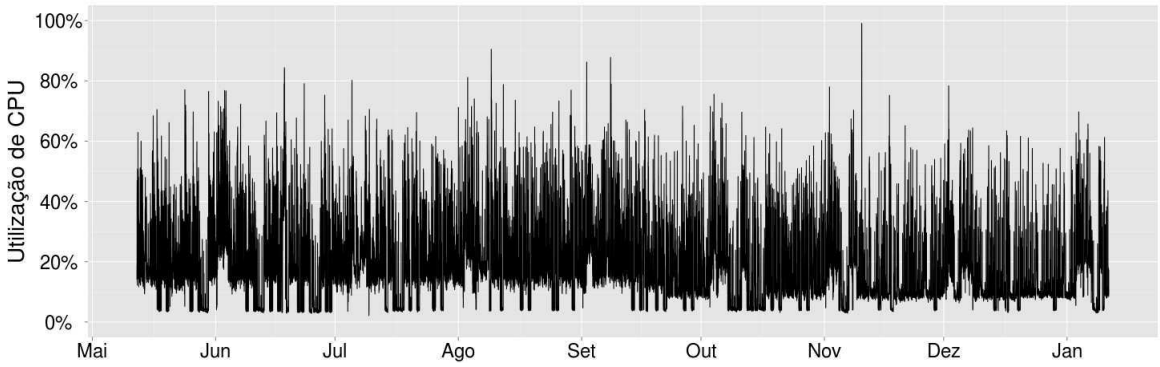


Figura A.152: Rastro 19 de longa duração.

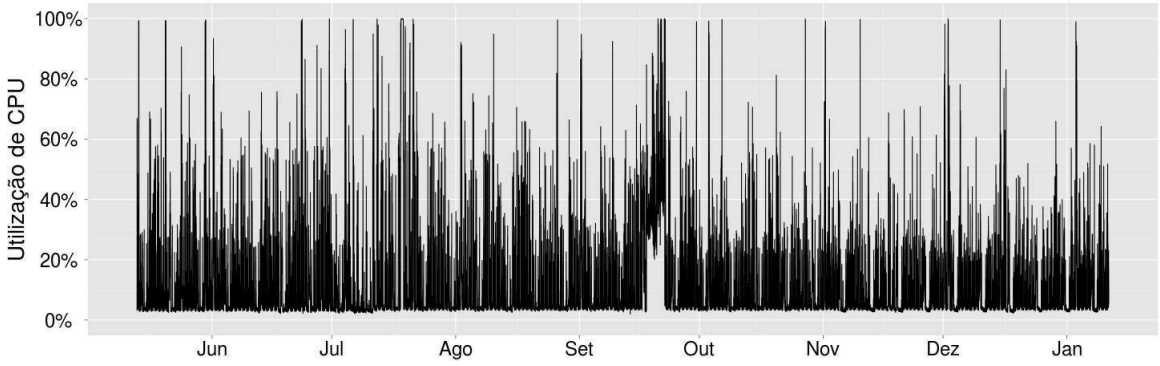


Figura A.153: Rastro 20 de longa duração.

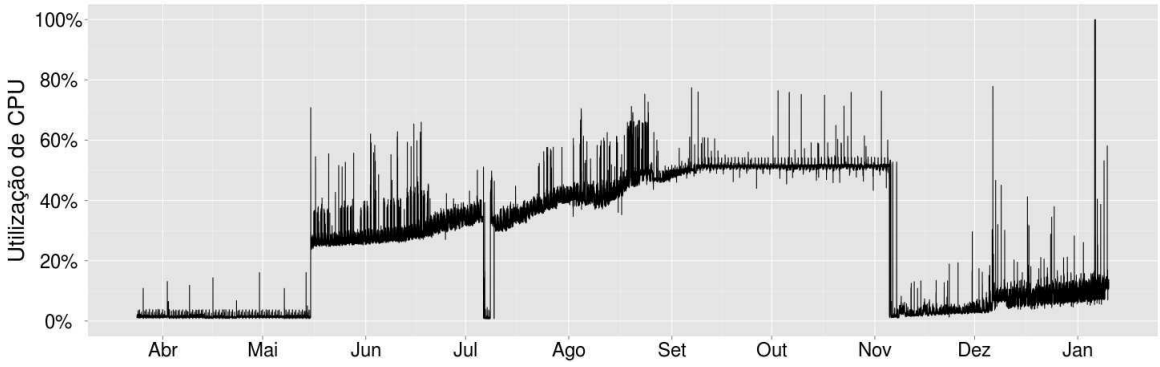


Figura A.154: Rastro 21 de longa duração.

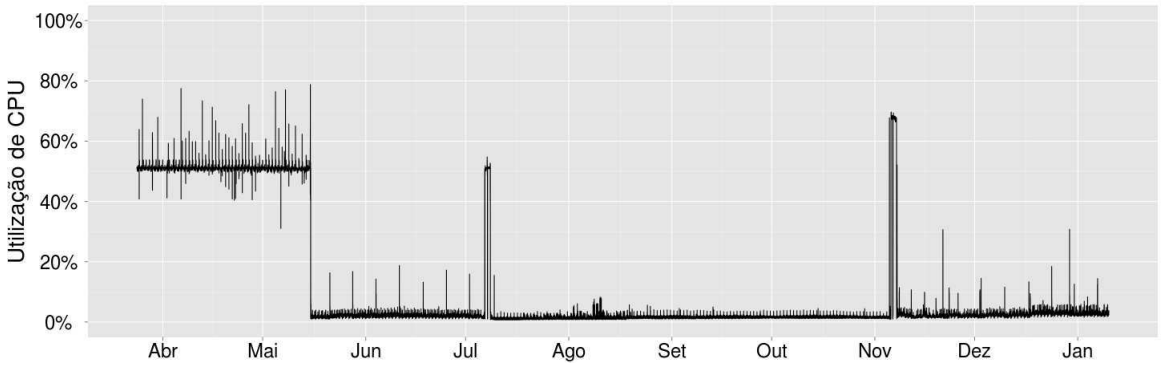


Figura A.155: Rastro 22 de longa duração.

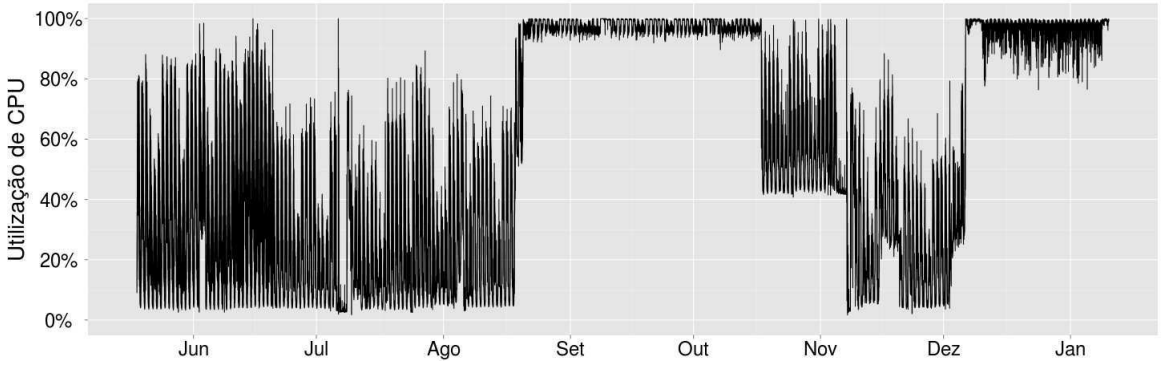


Figura A.156: Rastro 23 de longa duração.

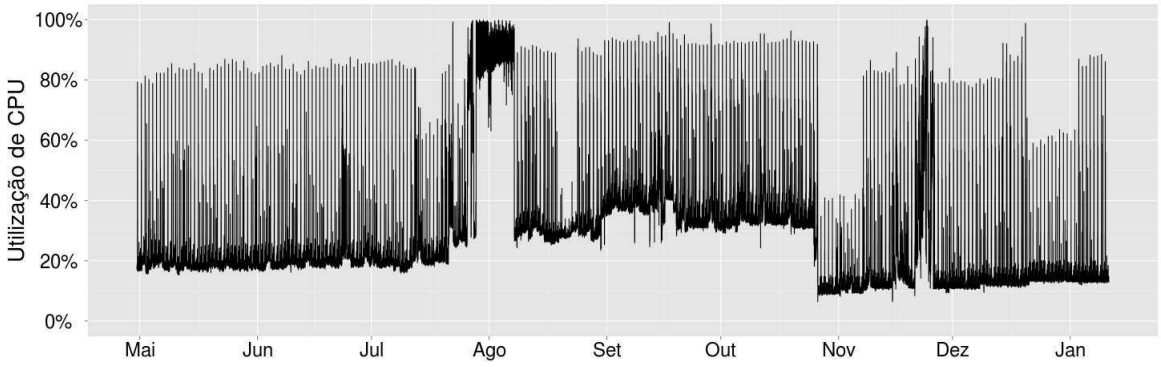


Figura A.157: Rastro 24 de longa duração.

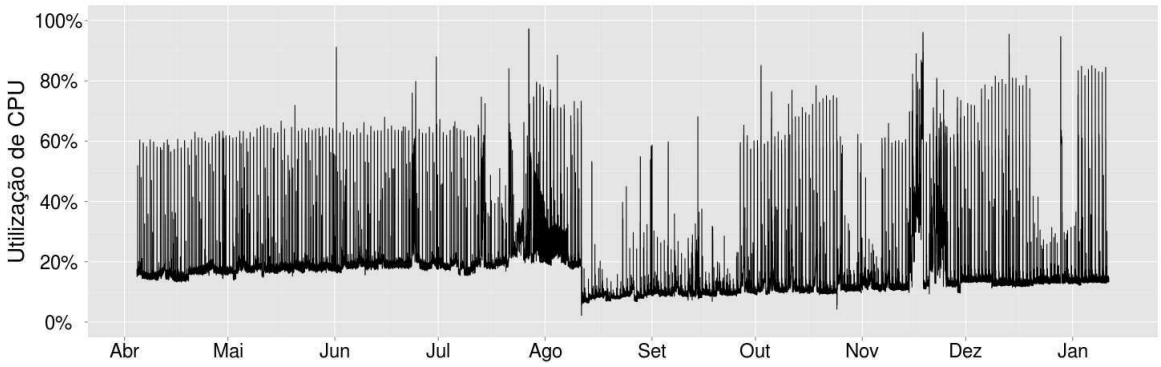


Figura A.158: Rastro 25 de longa duração.

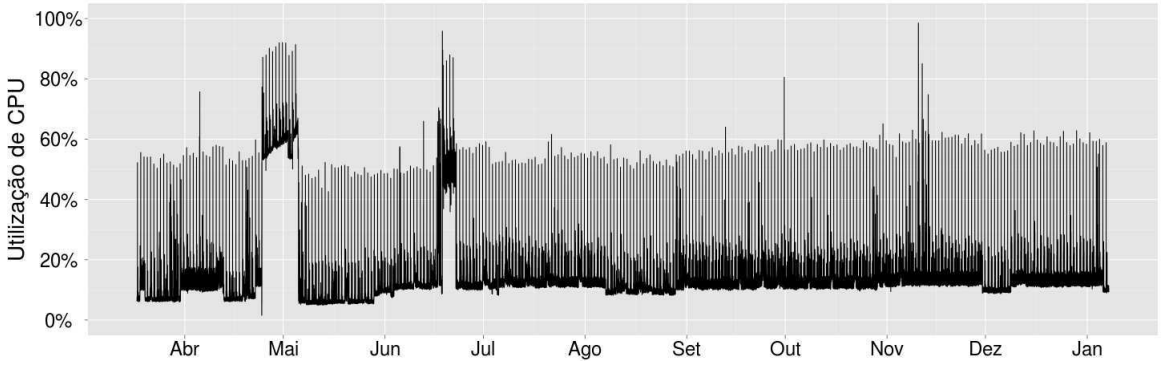


Figura A.159: Rastro 26 de longa duração.

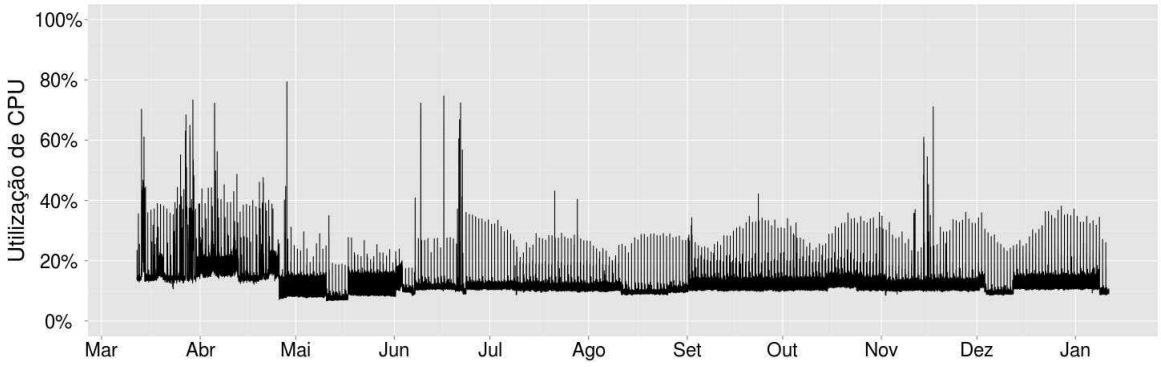


Figura A.160: Rastro 27 de longa duração.

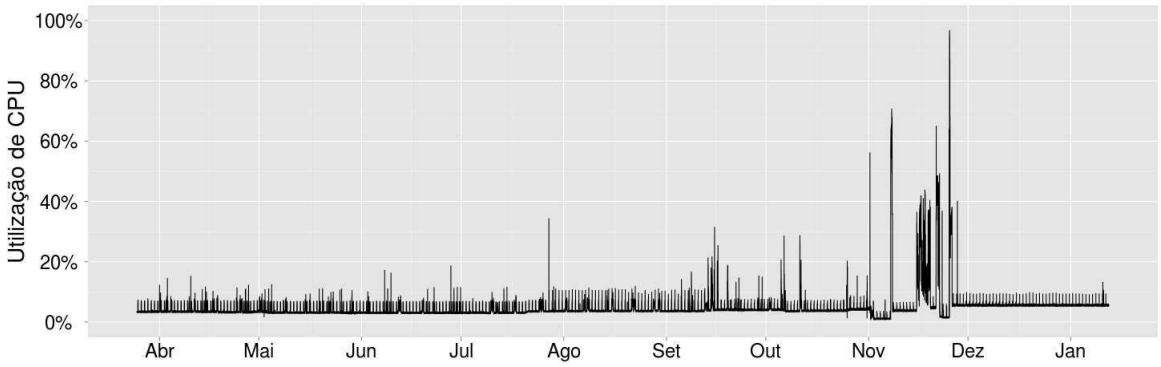


Figura A.161: Rastro 28 de longa duração.

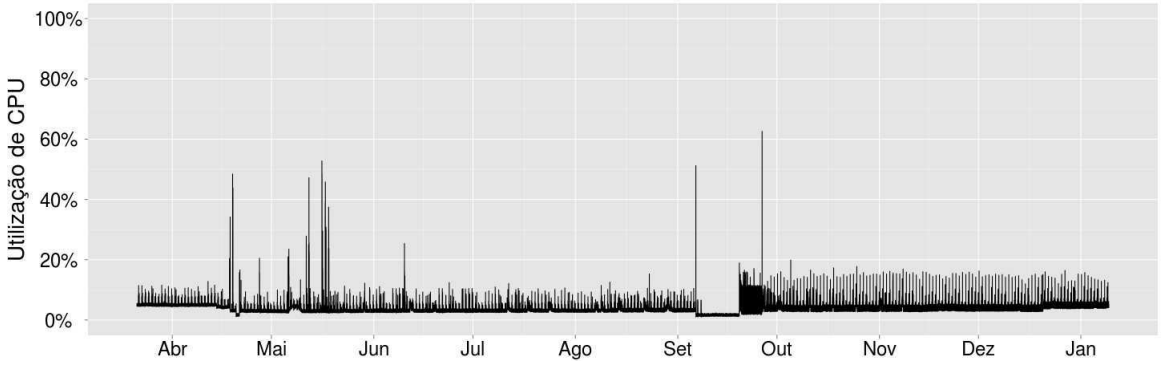


Figura A.162: Rastro 29 de longa duração.

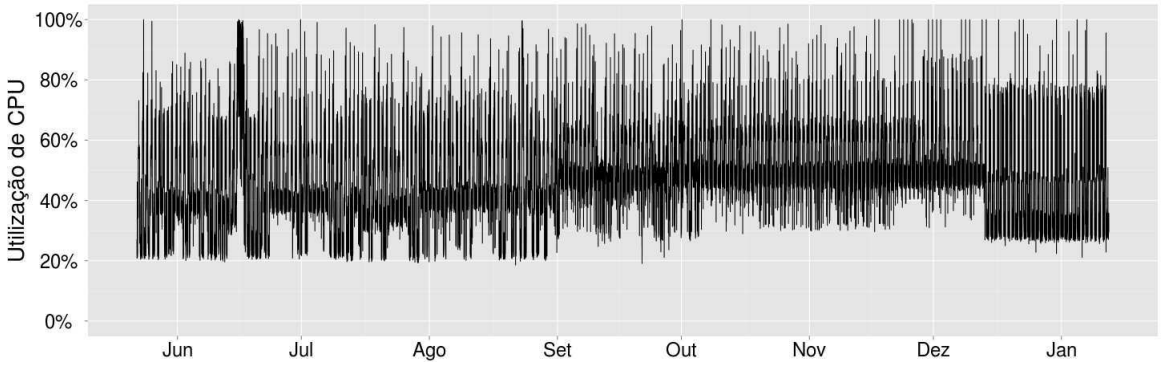


Figura A.163: Rastro 30 de longa duração.

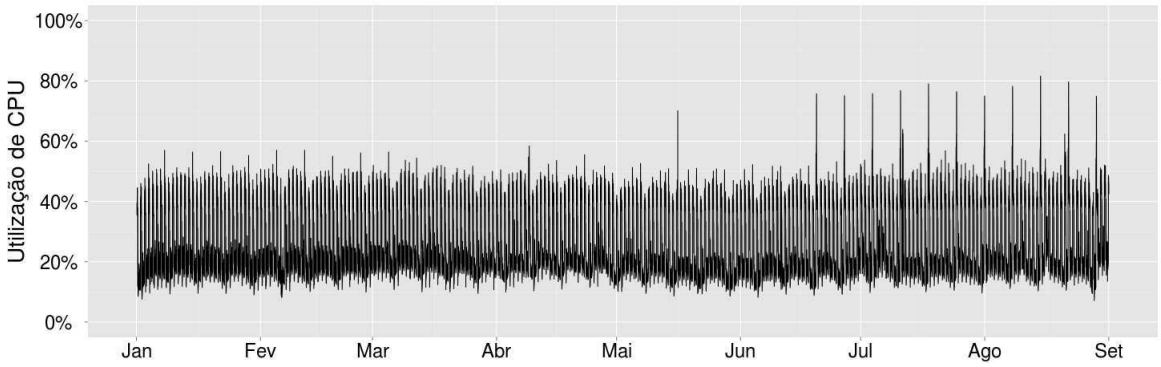


Figura A.164: Rastro 31 de longa duração.

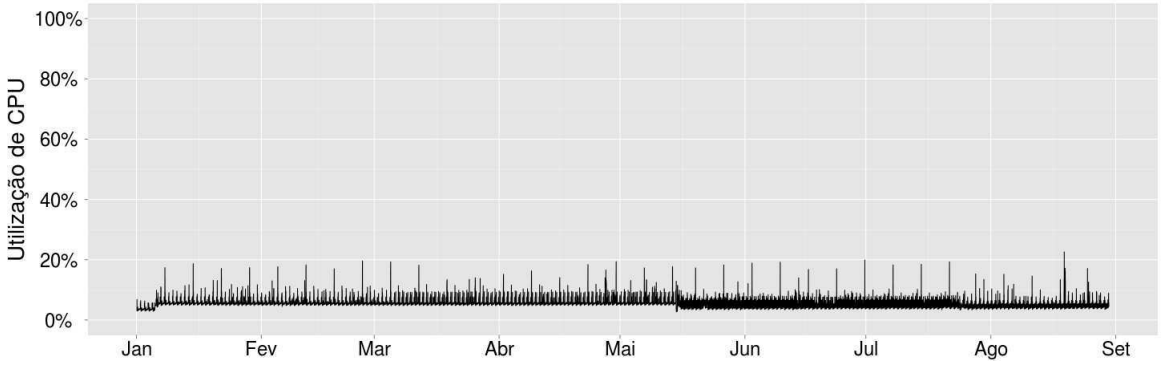


Figura A.165: Rastro 32 de longa duração.

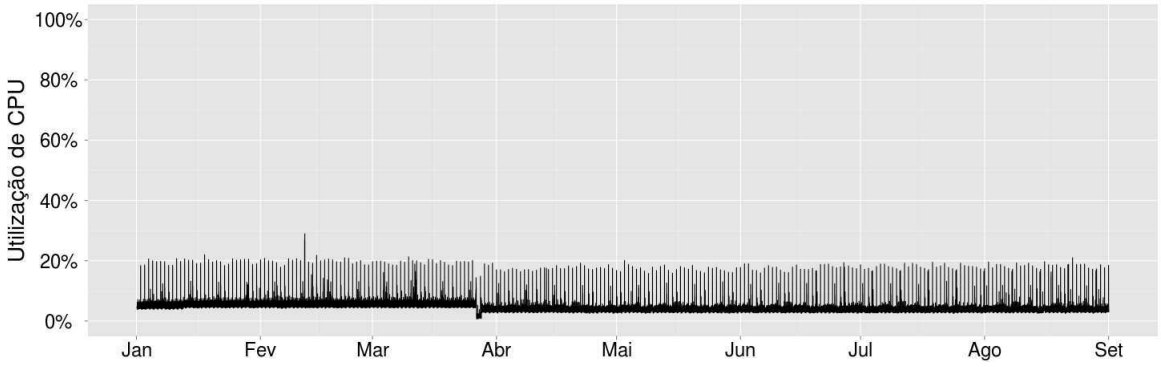


Figura A.166: Rastro 33 de longa duração.