UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

PROGRAMA DE PÓS–GRADUAÇÃO EM INFORMÁTICA APLICADA

CLEYTON FERREIRA GONÇALVES

MODELOS ESTOCÁSTICOS PARA O PLANEJAMENTO DE AMBIENTES AVA MOODLE BASEADOS EM CONTÊINERES E MÁQUINAS VIRTUAIS

RECIFE – PE

2022

CLEYTON FERREIRA GONÇALVES

MODELOS ESTOCÁSTICOS PARA O PLANEJAMENTO DE AMBIENTES AVA MOODLE BASEADOS EM CONTÊINERES E MÁQUINAS VIRTUAIS

Dissertação submetida à Coordenação do Programa de Pós-Graduação em Informática Aplicada da Universidade Federal Rural de Pernambuco, como parte dos requisitos necessários para obtenção do grau de Mestre.

ORIENTADOR: Prof. Dr. Ermeson Carneiro de Andrade

RECIFE – PE

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

G635m Ferreira Gonçalves, Cleyton
MODELOS ESTOCÁSTICOS PARA O PLANEJAMENTO DE AMBIENTES AVA MOODLE BASEADOS EM CONTÊINERES E MÁQUINAS VIRTUAIS / Cleyton Ferreira Gonçalves. - 2022.
105 f. : il.

Orientador: Ermeson Carneiro de Andrade.
Inclui referências.

Dissertação (Mestrado) - Universidade Federal Rural de Pernambuco, Programa de Pós-Graduação em Informática Aplicada, Recife, 2022.

1. Contêiner. 2. Máquina Virtual. 3. Avaliação de Desempenho. 4. Consumo de Energia. 5. Redes de Petri Estocásticas. I. Andrade, Ermeson Carneiro de, orient. II. Título

CDD 004

CLEYTON FERREIRA GONÇALVES

MODELOS ESTOCÁSTICOS PARA O PLANEJAMENTO DE AMBIENTES AVA MOODLE BASEADOS EM CONTÊINERES E MÁQUINAS VIRTUAIS

Dissertação submetida à Coordenação do Programa de Pós-Graduação em Informática Aplicada do Departamento de Estatística e Informática - DEINFO - Universidade Federal Rural de Pernambuco, como parte dos requisitos necessários para obtenção do grau de Mestre.

Aprovada em: 20 de abril de 2022.

BANCA EXAMINADORA

Prof. Dr. Ermeson Carneiro de Andrade (Orientador)
Universidade Federal Rural de Pernambuco - UFRPE
Departamento de Computação

Prof. Dr. Gustavo Rau de Almeida Callou
Universidade Federal Rural de Pernambuco - UFRPE
Departamento de Computação

Prof. Dr. Bruno Costa e Silva Nogueira
Universidade Federal de Alagoas - UFAL
Instituto de Computação

Dedico esta dissertação de mestrado à minha esposa Lianne que esteve sempre me incentivando a ser alguém melhor e ao meu filho Guilherme que teve momentos de nossa convivência renunciados em prol desta pesquisa.

Agradecimentos

Agradeço a minha família por ter tido paciência suficiente neste percurso que me fez abdicar de vários momentos importantes de convivência em detrimento da pesquisa e do estudos. Agradeço especialmente a minha esposa, Lianne, pela paciência, colaboração e cumplicidade em toda trajetória deste trabalho. Também externo os meus agradecimentos à UFRPE pela infraestrutura fornecida durante a pesquisa e os professores que colaboraram com o meu aprendizado, como Prof. Dr. Júlio Mendonça, Prof. Dr. Gustavo Callou, Prof. Dr. Bruno Nogueira, Prof. Dr. Tiago Ferreira e Prof. Dr. Gilberto Cysneiros.

Por fim, declaro os meus sinceros agradecimentos ao Prof. Dr. Ermeson Andrade pelos conhecimentos proferidos em sala de aula e a orientação fornecida durante a vigência da pesquisa.

Se eu tivesse apenas uma hora para cortar uma árvore, eu usaria os primeiros quarenta e cinco minutos afiando meu machado.

(Abraham Lincoln)

Resumo

Os Ambientes Virtuais de Aprendizagens (AVA) Moodle representam ferramentas de dimensão pedagógica, onde o professor utiliza vários recursos para estimular a aprendizagem dos alunos. Os conteúdos apresentados nessas plataformas em formatos de hipertextos, áudios ou vídeos podem ser adotados como meios para facilitar a aprendizagem. No entanto, elas tendem a produzir altas taxas de processamento nos servidores, grandes volumes de dados na rede e, consequentemente, degradam o desempenho, aumentam o consumo de energia e os custos. Assim, para fornecer o compartilhamento eficiente de recursos computacionais e ao mesmo tempo minimizar os custos financeiros, os AVAs normalmente são executados em infraestruturas virtualizadas provisionadas em Máquinas Virtuais (VM) ou contêineres, as quais possuem vantagens e desvantagens. Os modelos estocásticos, como as Redes de Petri Estocásticas (SPNs), podem ser usados na modelagem e avaliação de tais ambientes. Dessa forma, este trabalho utiliza as SPNs para avaliar o desempenho, o consumo de energia e o custo de ambientes baseados em contêineres e VMs. Métricas como vazão, tempo de resposta, consumo de energia e custo são coletadas e analisadas. Para demonstrar a aplicabilidade do trabalho proposto, dois estudos de caso são apresentados: o primeiro analisa os ambientes AVA Moodle baseados em contêineres e VMs através de *benchmarks*, enquanto o segundo adota as SPNs para modelar e analisar tais ambientes. Dentre os resultados obtidos, foi possível observar que, por exemplo, um agrupamento com 10 réplicas, na sua vazão máxima, pode gerar uma redução de 46,54% na demanda de energia elétrica caso os contêineres sejam usados. Adicionalmente, validamos a acurácia dos modelos analíticos comparando os resultados gerados nas análises com os experimentos obtidos em uma infraestrutura real.

Palavras-chave: Contêiner. Máquina Virtual. Avaliação de Desempenho. Consumo de Energia. Custo. Redes de Petri Estocásticas

Abstract

Moodle Virtual Learning Environments (VLEs) represent tools of a pedagogical dimension, where a teacher uses various resources to stimulate student learning. Content presented on these platforms as hypertext, audio, or video formats can be adopted to facilitate learning. Nevertheless, these platforms tend to produce high processing rates on servers, large volumes of data on the network, degrade performance, and increase energy consumption and costs. However, to provide efficient sharing of computing resources and at the same time minimize financial costs, VLE platforms typically run on virtualized infrastructures provisioned in Virtual Machines (VM) or containers, which have advantages and disadvantages. Stochastic models, such as Stochastic Petri Nets (SPNs), can be used to model and evaluate such environments. Therefore, this work aims to use analytical modeling through SPNs to assess the performance, energy consumption, and cost of environments based on containers and VMs. Metrics such as throughput, response time, energy consumption, and cost are collected and analyzed. Two case studies are presented to demonstrate the applicability of the proposed work: the first analyzes the AVA Moodle environments based on containers and VMs through benchmarks, while the second adopts the SPNs to model and analyze such environments. Among the results obtained, it was possible to observe that, for example, a cluster with ten replicas, occupied at its maximum capacity, can generate a 46.54% reduction in energy consumption if containers are used. Additionally, we validate the accuracy of the analytical models by comparing their results with the results obtained in real infrastructure.

Keywords: Container. Virtual Machine. Performance Evaluation. Energy Consumption. Cost Evaluation. Stochastic Petri Nets

Lista de Figuras

Lista de tabelas

Lista de Siglas

EAD	*Ensino à distância*
AVA	*Ambiente Virtual de Aprendizagem*
Moodle	*Modular Object-Oriented Dynamic Learning Environment*
VM	*Virtual Machine*
KVM	*Kernel-based Virtual Machine*
SPN	*Stochastic Petri Net*
SLA	*Service Level Agreement*
TI	*Tecnologia da Informação*
SO	*Sistema Operacional*
SDN	*Software Defined Networking*
API	*Application Programming Interface*
CLI	*Command-Line Interface*
HD	*Hard Disk*
RAM	*Random Access Memory*
TCP	*Transmission Control Protocol*

Sumário

1 Introdução

Os *data centers* de alta performance possuem infraestruturas para suportar altas taxas de utilização dos recursos em virtude da demanda em larga escala. Devido ao aumento da utilização dos serviços de TIC (Tecnologia da Informação e Comunicação), o poder computacional requerido no processamento e na transmissão de dados tem tornado as infraestruturas dos *data centers* como grandes instalações compostas por várias salas ou edificações. Esses ambientes são compostos por diferentes componentes computacionais, como servidores convencionais, equipamentos de processamento (ex.: *clusters* com *blades*) e armazenamento de dados (ex.: *storages*), dispositivos de rede (ex.: roteadores e *switches*), alimentadores de energia e sistema de refrigeração. Todos esses componentes funcionam para entregar serviços com aplicações atreladas a grandes volumes de dados com características diversas, elevando a utilização dos recursos, consumo de energia e custo para manter a operação.

O crescente consumo de serviços de TICs, em grande parte decorrente do aumento nas aquisições de dispositivos móveis, tem impactado no consumo de energia e na emissão de gases de efeito estufa por parte dos *data centers* (BELKHIR; ELMELIGI, 2018). A (CISCO, 2021) estimou que, até o ano de 2023, os serviços atrelados às plataforma de *streaming* de vídeos representarão 82% parcela do tráfego da Internet. Essa projeção confirma uma maior participação dos serviços de *streaming* em relação aos percentuais de utilização dos recursos e ao consumo de energia atribuído aos *data centers*. Segundo uma pesquisa realizada por Andrae e Edler (ANDRAE; EDLER, 2015), até 2030 a indústria de TIC poderá ser responsável por valores entre 8% e 21% das emissões de carbono global. A Nature (JONES, 2018) pontua que as emissões de carbono geradas pela indústria global de TIC representa mais de 2% das emissões globais de carbono, o que são comparáveis às emissões geradas pela indústria de aviação. Nos EUA, no ano de 2018, por exemplo, o consumo de energia anual dos *data centers* representou cerca de 200 terawatt-hora (tWh). Esse valor correspondeu a 1% da demanda de energia elétrica global, o que supera o consumo de energia de alguns países (JONES, 2018). Durante a pandemia do (covid-19), foi necessário submeter a população à medidas de quarentena e distanciamento social para conter a proliferação do vírus. Em virtude da necessidade de permanecer mais tempo nas residências, as pessoas passaram a usar Internet com mais frequência

para diversas atividades. Dessa forma, a pandemia do coronavírus acelerou o processo de transformação digital, devido ao crescimento das demandas de serviços online, como, por exemplo, teletrabalho, telemedicina, jogos online, serviços de *streaming* aplicados a entretenimento e educação à distância. Dessa forma, o volume de tráfego na Internet foi impulsionada rapidamente em curto espaço de tempo, aumentando consideravelmente a taxa de utilização dos recursos e a demanda de energia elétrica dos *data centers* (AMDOCS, 2021).

Os conteúdos relacionados a *streaming* de áudio e vídeo tendem a produzir cargas de trabalhos superiores em relação a outros formatos de mídias. Sobretudo, quando as mídias de áudio e vídeo estão atrelados às aplicações voltadas para os serviços de *streaming* de entretenimento e ensino à distância (educação), visto que tais nichos normalmente possuem fortes taxas de adesão, devido à diversidade de plataformas de ensino à distância (EAD) que promovem excelentes experiências aos usuários, produzindo resultados satisfatórios na aprendizagem. Considerando o nicho da educação, os AVAs representam plataformas de ensino via softwares que têm contribuído de forma promissora na difusão do conhecimento com a colaboração de recursos digitais, independentemente da modalidade de aprendizagem. Os AVAs realizam o gerenciamento de todo o processo de aprendizagem através de práticas pedagógicas e métodos de ensino implementados em aplicações Web ou mobile. Um tipo de AVA amplamente adotado no âmbito acadêmico, é o *Modular Object Oriented Dynamic Learning Environment*, ou simplesmente Moodle, o qual oferece bastantes recursos e ferramentas que auxiliam todo o ciclo de aprendizagem presencial, semipresencial e à distância (SALEKHOVA et al., 2019).

Devido à consolidação da virtualização conciliada com as nuvens computacionais, sejam públicas ou privadas, foi acelerado o processo de migração ou construção de negócios educacionais, viabilizando o funcionamento de grandes plataformas de ensino cujo ciclo de aprendizagem são gerenciados por aplicações baseadas em AVAs. Isso assegurou que as plataformas AVAs virtualizados contribuíssem na redução do consumo de energia e, consequentemente, mitigar da emissão de carbono, visto que a utilização dos recursos computacionais tornaram-se maneira mais eficiente (Chen et al., 2019). Com a virtualização, os hospedeiros têm os seus recursos compartilhados entre uma maior quantidade de máquinas virtuais (VMs), reduzindo drasticamente a ociosidade dos recursos dentro dos *data centers* (Cuadrado-Cordero et al., 2017). A VM representa uma réplica idêntica

de um computador composto pelos seus os dispositivos virtuais e sistema operacional (SO) (Tadesse et al., 2017). O *hypervisor* refere-se à camada de software intermediária que redireciona as instruções dos SOs convidados (*guest OS*) das VMs para o hardware do computador hospedeiro (Tadesse et al., 2017). Outro mecanismo conhecido é o contêiner, o qual tem como característica principal compartilhar o próprio *kernel* do SO hospedeiro, dispensando o uso do *hypervisor* e do SO convidado (Tadesse et al., 2017). O descarte do SO convidado para os contêineres, asseguram tempos de inicializações significativamente inferiores aos das VMs. Esses aspectos tornam os contêineres mais velozes e flexíveis para distribuir aplicações. Além do mais, demandam menos utilização dos recursos por serem compactos e leves (LIN et al., 2018). Dessa forma, os ambientes AVA Moodle provisionados por contêineres podem ser uma alternativa viável em substituição às VM, visando economizar os recursos compartilhados e, consequentemente, reduzir o consumo de energia e o custo da operação. Todavia, o desempenho e o consumo de energia da operação pode ser afetado com subdimensionamento de recursos, bem como trazer problemas de custos elevados devido ao superdimensionamento de recursos. Dessa forma, será necessário avaliar *trade-offs* relacionadas às métrica de desempenho e de consumo de energia.

A definição da capacidade de um *cluster* não é um tarefa simples de implementar, no entanto, as técnicas de modelagens podem auxiliar na identificação de valores aproximados para todos os parâmetros de configuração dos ambientes. Isso permite que os ambientes AVA Moodle virtualizadas possam suportar cargas de trabalhos de tamanhos variados e cumprir com exatidão os requisitos de desempenho e de consumo de energia. O mecanismo de auto-escalonamento consiste em uma técnica de elasticidade de capacidade de réplicas, implementado na maioria dos sistemas de orquestração de *clusters*. No entanto, a capacidade do *cluster* está limitada à quantidade de nós definidos na sua criação. Com a adoção de modelos analíticos é possível predizer os limites mínimos e máximos das instâncias adequadas às políticas de auto-escalonamento. A modelagem analítica é uma técnica poderosa que têm sido empregada para representar e analisar o desempenho de diversos tipos de sistemas complexos, incluindo sistemas computacionais, como *data centers* (BALBO, 2007; MURATA, 1989; MARSAN et al., 1994). O uso da modelagem analítica pode ajudar a analisar diferentes aspectos referente ao desempenho, a confiabilidade e a eficiência energética no contexto dos sistemas computacionais. Dentre os vários tipos de modelagens analíticas, como cadeia de *Markov*, árvore de decisão, diagrama

de blocos de confiabilidade (*Reliability block diagram* - RDB), as redes de Petri (RdP), entre outros. Destaca-se as redes de Petri (RdP) e suas extensões, as quais correspondem a formalismos amplamente adotados para representar os estados de sistemas através de uma notação formal composta por elementos gráficos. Esses modelos podem ser criados para representar abstrações de sistemas complexos cujos comportamentos podem denotar concorrência, paralelismo, assincronicidade, distribuição, etc (MURATA, 1989; MARSAN et al., 1994). Vale destacar que as RdPs podem ser usadas para encontrar possíveis pontos de gargalos nas infraestruturas sob análise, realizar o planejamento de capacidade e prover informações para os projetistas/administradores na tomada de decisão.

1.1 Motivação

O cenário pandêmico trouxe várias mudanças no cotidiano do mundo em relação à mobilidade da população, devido às determinações governamentais de distanciamento social para conter a disseminação do vírus SARS-CoV-2. No auge da pandemia, os acessos aos estabelecimentos comerciais e públicos foram limitados apenas a grupos estritamente necessários. Dessa forma, as rotinas das pessoas limitaram boa parte do tempo as suas residências. Esse contexto ocasionou um crescimento na demanda dos serviços online relacionados às redes sociais, aos *e-commerces*, às aulas e cursos online, aos serviços de *streaming* para filmes e músicas, as videoconferências para as mais diversas finalidades e ao *homeoffice*. Esse último levou uma grande parcela dos negócios a migrarem as suas atividades para a modalidade de teletrabalho que representou um aumento no uso da Internet nas residências através de aplicações nas nuvens. No contexto mais amplo, o uso da Internet cresceu em escala global durante a pandemia. Nos EUA, no ano de 2021, uma pesquisa da (AMDOCS, 2021) mostrou que os jogos online tiveram um aumento de 49% em relação ao período pré-pandemia, seguidos por serviços de *streaming* de vídeo com 48% e *e-Learning* com 45%. No ano de 2020, a Anatel mostrou que o consumo da Internet no Brasil teve um aumento entre 40% a 50% (ANATEL, 2020). De acordo com a (CISCO, 2021), os serviços remotos de saúde, *streaming* de vídeo e de jogos e *homeoffice* representaram um crescimento entre 25% a 45% no tráfego da Internet em muitos pontos do terra. A (AMDOCS, 2021) afirma que os usuários na Internet tiveram um aumento do uso de aproximadamente 2 horas por dia em relação ao período de pré-pandemia.

Nesse quesito, o Brasil passou de 2,5 horas para 3 horas e 43 minutos. Essa mudança de comportamento no tráfego de rede levou os provedores de serviços a fazer ajustes na capacidade de suas infraestrutura para suportar maiores taxas de requisições. Os provedores de serviços como *Netflix*, *Amazon*, *Youtube*, *Facebook* e *Instagram* tiveram que reduzir a qualidade de resolução dos vídeos para haver um equilíbrio entre a demanda e o desempenho da operação.

Os setores públicos e privados da educação também foram afetados em todos os níveis de ensino, migrando a continuidade dos anos letivos para a modalidade de ensino à distância através da utilização de plataformas baseadas em AVAs. Isso obrigou as instituições acadêmicas a expandirem suas capacidades de operações dos seus *data centers* para atender o aumento na demanda de milhares estudantes que utilizaram os recursos digitais como meios de aprendizagens. Não somente durante a pandemia, mas ao longo dos anos, a boa aceitação dos AVAs perante à comunidade acadêmica tem crescido consideravelmente, à medida que os estudantes e docentes tornam-se mais experientes e adaptados às rotinas de estudos através de recursos digitais. O AVA Moodle dispõe de práticas pedagógicas que tornam as rotinas de estudos mais produtivas, eficazes e agradáveis, podendo resultar tanto no aumento das frequências de acessos quanto no tempo de permanência dos estudantes na plataforma. Note que os servidores de aplicações tendem a receber uma maior quantidade de requisições, produzindo grandes volumes de dados na rede e aumentando a utilização dos recursos. Isso torna a operação mais onerosa relação ao desempenho, ao consumo de energia e ao custo (LIMA et al., 2019).

É importante haver um equilíbrio razoável entre as cargas de trabalhos e os recursos alocados para variados cenários, evitando ajustes incorretos na capacidade da infraestrutura que possam causar problemas de queda no desempenho da operação, em virtude do subdimensionamento dos recursos. Por outro lado, o superdimensionamento pode gerar alocação desnecessárias de recursos que poderiam ser destinados a outras finalidades. Para fornecer o compartilhamento eficiente de recursos computacionais e, ao mesmo tempo, minimizar os custos financeiros, a plataforma Moodle normalmente é executada em infraestruturas virtualizadas como VMs ou contêineres. No entanto, a VM pode acrescentar uma sobrecarga desnecessária ao desempenho geral do sistema operacional hospedeiro, visto que cada instância baseada em VM possui seu próprio sistema operacional (SO) (Bhimani et al., 2017). Por outro lado, a tecnologia contêiner preconiza baixa utilização dos

recursos de hardware, visto que dispensa o uso do *hypervisor*, VMs e sistemas operacionais convidados (Salah et al., 2017), isto é, os contêineres são carregados no nível do sistema operacional da máquina física. Adicionalmente, a virtualização através das VMs permite o trabalho com SOs diversos no mesmo ambiente, enquanto o contêiner possui a dependência do SO que ele está rodando. Como cada um desses ambientes virtualizados possuem suas especificidades e devem atender às necessidades dos mais variados negócios, se faz necessário estudar os *trade-offs* em termo de desempenho, consumo de energia e custo dessas infraestruturas (GONCALVES et al., 2020).

Os modelos baseados em Redes de Petri Estocásticas podem contribuir para minimizar incertezas relacionadas ao planejamento de capacidade dos *data centers*, auxiliando as partes interessadas na definição de limiares de réplicas necessárias para suportar as cargas de trabalhos conforme os variados cenários. Os modelos SPNs possuem fundamentos matemáticos sólidos e uma semântica precisa que garante uma maior acurácia nas análises e simulações, reduzindo as taxas de erros nas previsões de cenários. Através dos modelos SPNs, pode-se avaliar um conjunto métricas relacionadas ao funcionamento das infraestruturas destinadas às plataformas Moodle ou as diversas aplicações distribuídas. Com as SPNs, levando em consideração um conjunto de parâmetros, os escopos das análises podem ser extrapolados para configurações de ambientes complexos e caros de implementar no mundo real. As análises permitem observar métricas, como vazão, tempo de resposta e potência elétrica. Essas métricas são importantes nas escolhas das melhores soluções (MURATA, 1989; MARSAN et al., 1994).

1.2 Objetivos

Uma infraestrutura computacional com a capacidade bem ajustada à demanda pode trazer ganhos financeiros e ambientais para os gestores, desde que o *data center* possa prover operações dentro de níveis de serviços razoáveis, cumprindo os requisitos de desempenho e gerando custos satisfatórios no orçamento da operação. Isto é, uma infraestrutura de *data center* pode suportar diversas cargas de trabalhos com os limites de recursos alocados adequados para atendê-las, sem causar prejuízo a experiência do usuário final. No entanto, esses objetivos só podem ser atingidos com a adoção de práticas e tecnologias que visem gerar menos utilização de recursos e consumo de energia, tornando a operação mais leve

e sustentável para o meio ambiente. Boa parte dos *clusters* computacionais possuem funcionalidades elásticas que aumentam e diminuem as quantidades réplicas conforme variações de cargas de trabalhos. O termo comum é chamado de auto-escalonamento, o qual toma decisões baseadas em parâmetros de desempenho que disparam gatilhos para escalar as VMs ou os contêineres. Contudo, o grande desafio dos projetistas e gestores de *data centers* é definir as melhores configurações de ambientes para atender as demandas de variados cenários concretos. Decisões equivocadas podem afetar negativamente a qualidade dos serviços oferecidos.

Nesse contexto, considerando os desafios expostos acima, o objetivo geral desta dissertação será propor uma abordagem integrada baseada em experimentos e SPNs para auxiliar as partes interessadas no planejamento de capacidade de *clusters* baseados em contêineres ou VMs. Os modelos SPNs propostos têm como principal objetivo avaliar os *trade-offs* das infraestruturas adotadas com o foco no AVA Moodle, através da obtenção de métricas como a vazão, o tempo de resposta e o consumo a energia para cada cenário avaliado. Eles ainda tornam possível estimar os limites mínimos e máximos das instâncias adequadas às políticas de auto-escalonamento. Além disso, este trabalho objetiva montar e analisar arquiteturas experimentais para tanto validar os modelos SPNs quanto para avaliar as operações equivalentes à plataforma AVA provida por ambientes baseados em contêineres e VMs. Uma metodologia de avaliação que compreende a definição de parâmetros, a criação de modelos de desempenho das infraestruturas, a avaliação dos modelos e a análise dos resultados também é proposta. Através dos resultados, é possível indicar a capacidade de recursos necessários para cumprir determinadas cargas de trabalhos, conforme métricas de desempenho, consumo de energia e custo. Por fim, os modelos propostos têm como intuito fornecer informações para apoiar na tomada de decisão de quaisquer partes interessadas, como administradores, especialistas, projetistas e gestores.

Mais especificamente este trabalho possui os seguintes objetivos:

- Montar e executar experimentos em infraestruturas reais com o foco na plataforma AVA Moodle;
- Avaliar e comparar o desempenho, o consumo de energia e o custo do AVA Moodle provisionados por contêineres e VMs;
- Elaborar uma metodologia de modelagem e análise de ambientes implantados com o Moodle;

- Desenvolver e validar os modelos SPNs que representem cenários do AVA Moodle provisionados por contêineres e VMs;

1.3 Estrutura da Dissertação

O restante desta dissertação está organizado como segue. A Seção 2 descreve os principais conceitos usados neste trabalho. A Seção 3 detalha os trabalhos relacionados à avaliação de desempenho, de consumo de energia e de custo dos ambientes AVA baseados em contêineres e VMs. A Seção 4 apresenta a metodologia utilizado. A Seção 5 apresenta as arquiteturas experimentais adotadas. A Seção 6 descreve os modelo SPNs desenvolvidos. A Seção 7 apresenta uma discussão sobre os resultados. Por fim, a Seção 8 apresenta as conclusões, os trabalhos futuros e as limitações deste trabalho.

2 Fundamentação Teórica

Esta seção apresenta os conceitos fundamentais empregados neste trabalho. A seguir são apresentados os conceitos acerca do AVA, da virtualização, da conteinerização e da modelagem usando as redes de Petri estocásticas.

2.1 Ambientes Virtuais de Aprendizagem

Os AVAs definem-se como softwares desenvolvidos para auxiliar os professores na promoção do ensino e da aprendizagem de forma virtual, tendo como principais características a interatividade, a hipertextualidade e a conectividade (DEHON et al., 2018). As instituições acadêmicas podem adotá-los de acordo com a necessidade do seu contexto educacional para promover cursos nas modalidades presenciais, semipresenciais ou à distância (JR, 2019). Esses ambientes permitem integrar inúmeras metodologias e recursos, organizar e apresentar os conteúdos por mídias digitais e disseminar o conhecimento utilizando recursos de tecnologia da informação e comunicação. Os AVAs são considerados como sistemas de ensino e aprendizagem integrados, sendo capazes de promover o interesse do aluno na aprendizagem e complementar a interação em sala de aula com professor (JR, 2019). Os AVAs representam ferramentas de dimensão pedagógica onde o professor utiliza vários recursos para estimular a aprendizagem dos alunos. Os conteúdos apresentados em formatos de hipertextos, áudios ou vídeos podem ser adotados como meios para facilitar a aprendizagem, bem como incentivar o engajamento dos alunos na realização das atividades e questionários dinâmicos compostos por diversos recursos (LIMA et al., 2019). Vale destacar que os conteúdos disponibilizados pelas plataformas AVAs podem ser acessados de forma ampla e flexível em qualquer lugar (LIMA et al., 2019).

Os AVAs mais conhecidos são o Moodle, o TelEduc, o AulaNet, o Claroline, o BlackBoard, o E-Proinfo e o Amadeus. No entanto, entre eles, apenas o Moodle, o Claroline e o Amadeus possuem licenças de código aberto (DEHON et al., 2018). Destaca-se o Moodle com cerca de 181 mil instâncias (sites), distribuídas entre 234 países, 35 milhões de cursos e 262 milhões de usuários inscritos. O Moodle está entre as principais plataformas de aprendizagens online, desenvolvida sob licença de software livre e de código aberto. Essa característica viabilizou uma ampla adesão de colaboradores dos ramos de tecnologia e de

educação que contribuem para o desenvolvimento contínuo do produto. Com a criação da comunidade virtual, os desenvolvedores de softwares, administradores de sistemas, *designers*, educadores e usuários de diversos países conseguem facilmente trocar experiências acerca de melhorias no produto, acessar documentações técnicas, tirar dúvidas de implantações, reportar problemas ou *bugs*, obter orientações de uso, entre outras (MOODLE, 2021). De acordo com Dehon et al. (DEHON et al., 2018), o Moodle agrega mais funcionalidades e recursos pedagógicos em relação aos demais AVAs, tais como: permitir o uso de repositórios externos, contemplar uma vasta opções de *plug-ins* para agregar novos recursos ao ambiente, permitir a customização de *layout* disponibilizados gratuitamente, possuir uma ampla comunidade de desenvolvedores, entre outras. Essa variedade de recursos e funcionalidades, torna o Moodle uma plataforma versátil que pode ser utilizada em atividades diversas que envolvem a formação de grupos de estudo, treinamentos de pessoal e até desenvolvimento de projetos. Além do mais, outros setores, não relacionados à educação, tem adotado o Moodle como ferramenta colaborativa, como empresas privadas, ONGs, fundações, institutos de pesquisas e comunidades online (MOODLE, 2021). Em decorrência dessas vantagens, este trabalho adotará o AVA Moodle para modelagem e análise.

2.2 Virtualização

O termo virtualizar está relacionado ao conceito de simular algo real para o mundo virtual. A virtualização refere-se a um conjunto de tecnologias que permite um único computador físico hospedar múltiplas máquinas virtuais, onde cada instância representa um SO diferente em execução sobre o hardware subjacente compartilhado (BOS; TANENBAUM, 2015). A virtualização funciona no sentido de otimizar a utilização do hardware e minimizar a ociosidade dos recursos computacionais (REDHAT, 2020). Ela torna possível a utilização da capacidade total de uma máquina física, distribuindo recursos para múltiplos ambientes, como servidores de aplicação, serviços de armazenamento e serviços de rede de acordo com a necessidade (VMWARE, 2020). Dessa forma, com o uso da virtualização, há uma tendência de economia na aquisição de hardware, uma vez que os recursos computacionais podem ser alocados de forma mais aprimorada e econômica (BOS; TANENBAUM, 2015).

O ambiente virtualizado é caracterizado pela agilidade e a flexibilidade nas entregas.

Tais características aumentam a eficiência no provisionamento dos serviços ou aplicações. Além do disso, o gerenciamento do *data center* é simplificado através do aumento na automação das tarefas. Esse aspecto oferece mais mobilidade nas cargas de trabalho e, consequentemente, pode aumentar o desempenho e a disponibilidade dos serviços ou aplicações (Mukhedkar; Vettathu, 2016). De acordo com a (VMWARE, 2020), a virtualização oferece os seguintes benefícios: (a) redução dos custos operacionais e capitais; (b) redução ou eliminação do tempo de inatividade; (c) aumento de produtividade, eficiência, agilidade e capacidade de resposta da TI; (d) rapidez no provisionamento de recursos e aplicações; (e) melhor continuidade dos negócios; (f) gerenciamento simplificado das operações dos *data centers*; e (g) aumento na disponibilidade dos serviços e aplicações dos *data centers*.

A parte superior da Figura 1 apresenta um cenário cuja infraestrutura é composta por três servidores físicos com as mesmas características, mas cada um deles com finalidades específicas. O primeiro é um servidor de *e-mail*, o segundo é um servidor *web* e o terceiro executa diferentes aplicações. Em um cenário hipotético, considere que a utilização de tais serviços consome cerca de 30% da capacidade dos recursos de cada máquina física onde esses serviços estão instalados. Assim, cerca de 70% dos recursos de cada máquina física fica ociosa. Os valores elevados de ociosidade resultam em custos desnecessários que podem prejudicar a manutenção da operação do provedor de serviço. Considerando que os serviços de *e-mail* e *web* juntos utilizam cerca de 60% da capacidade dos recursos, uma única máquina física poderia suportar duas instâncias virtualizadas para tais serviços. Dessa forma, ainda restariam cerca 40% de recursos computacionais da máquina física para eventuais aumentos de demanda das aplicações executando nas instâncias virtualizadas. Levando em conta que o computador físico da Figura 1 adota um esquema de virtualização que permite criar instâncias de computadores denominados VMs, tanto o servidor de *e-mail* quanto o servidor *web* podem carregar o seu próprio SO de forma independente e isolada (ver parte inferior da figura 1). Sendo assim, as duas máquinas virtuais podem executar simultaneamente diferentes SOs compartilhando o hardware de um único computador físico. Dessa forma, a criação de múltiplas VMs, dentro de um único computador físico, permite que a operação seja otimizada através de características como a escalabilidade, a elasticidade de serviços e o balanceamento de carga (Mukhedkar; Vettathu, 2016).

A virtualização completa e a para-virtualização são as técnicas de virtualização

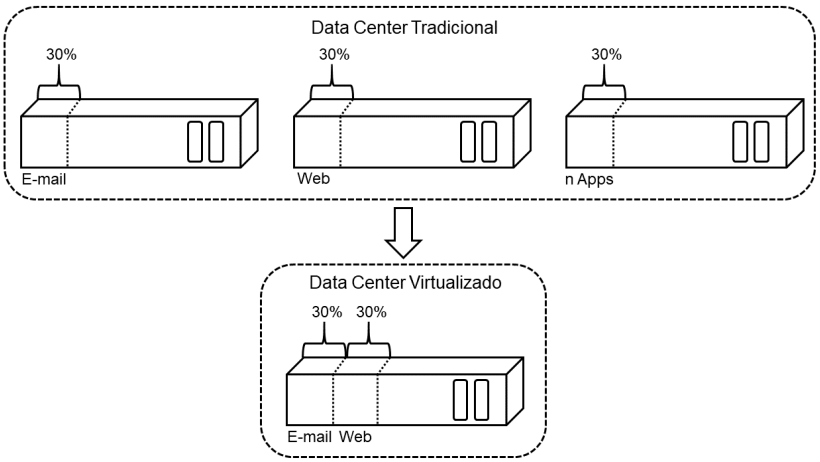

Figura 1 – Exemplo de três servidores tradicionais sendo virtualizados para um computador hospedeiro.

amplamente adotadas para provisionar VMs. A virtualização completa (ou total) utiliza o hardware subjacente da máquina física para criar replicas completas que emulam dispositivos de hardware virtuais cujo comportamento se assemelha a um computador convencional que pode executar um SO convidado. Nessa técnica, os OSs, os softwares, os serviços e as aplicações são executados diretamente no hardware emulado por um software de virtualização denominado monitor de máquina virtual (ou *virtual machine monitor*) que cria dispositivos virtuais nos SOs convidados. Sendo assim, o sistema hospedeiro ignora a existência da VM e opera como se funcionasse diretamente sobre o SO para o qual foi projetado. O resultado é um novo sistema virtual, no qual o SO convidado não sofre nenhuma modificação (Ivanov, 2017; Mukhedkar; Vettathu, 2016).

A para-virtualização realiza algumas modificações nos SOs convidados para que ocorram as interações com o *hypervisor* e as instruções sejam convertidas por ele ou interpretadas diretamente no hardware do computador hospedeiro. Nessa técnica, enquanto o sistema hospedeiro envia algumas instruções de baixo nível diretamente ao hardware hospedeiro, o *hypervisor* realiza a conversão das instruções de alto nível, intermediando a comunicação dos sistemas convidados e as camadas subjacentes. Os sistemas convidados da para-virtualização produzem menos sobrecargas no desempenho geral do sistema hospedeiro em relação aos sistemas convidados da virtualização completa (Ivanov, 2017; Mukhedkar; Vettathu, 2016). A implantação de algumas dessas técnicas varia conforme a necessidade da operação do negócio e a capacidade dos recursos.

Uma máquina virtual pode ser definida como uma réplica eficiente e isolada de um computador real com SO, aplicativos e serviços (BOS; TANENBAUM, 2015). Elas

podem simular vários computadores ou servidores virtuais hospedados em um determinado computador físico, proporcionando aos usuários as mesmas experiencias de um ou mais computadores reais. As VMs apresentam características como particionamento dos recursos do sistema entre múltiplas máquinas virtuais e a execução de diversos SOs na máquina física, incluindo as características de isolamento de falhas, segurança no nível de hardware, preservação no desempenho do sistema, otimização na alocação e gerenciamento dos recursos. Os SOs e as aplicações instanciados nas VMs são encapsulados em arquivos que podem ser movimentados ou replicados entre vários ambientes. Adicionalmente, a independência do hardware garante um rápido provisionamento ou migração das aplicações contidas nas VMs para qualquer nó ou ambiente físico. Para criar uma VM, é necessária a instalação de um software chamado de *hypervisor*, o qual representa uns dos principais componentes de um sistema virtualizado. Esses softwares gerenciam o hardware físico, distribuindo os recursos de processamento, de armazenamento, de conexões de rede entre diversas VMs.

Os *hypervisors* representam uma camada de software que dissocia as VMs da máquina física hospedeira e aloca os recursos de hardware dinamicamente ou estaticamente para cada VM criada. Eles possuem implementações baseadas em softwares e hardwares que permitem criar, executar e gerenciar as VMs (Ivanov, 2017). Sendo assim, os *hypervisors* gerenciam os recursos de hardware de uma máquina física entre várias VMs, seus SOs convidados e aplicações. Vale ser ressaltado que os recursos de uma máquina física como CPUs, memórias, discos rígidos, interfaces de redes e periféricos são compartilhados para vários SOs em forma de dispositivos virtuais. Essa flexibilidade de gerenciamento permite que as VMs sejam migradas com agilidade entre diferentes computadores hospedeiros (Barik et al., 2016). As soluções mais conhecidas de *hypervisors* incluem o Xen, o QEMU, o *VirtualBox*, o *VMware vSphere*, o *VMware Workstation*, o *Hyper-V* e o *Kernel-based Virtual Machine* (KVM) (RAHO et al., 2015). Esta dissertação fez a opção por adotar o KVM dada a sua disponibilidade nativa para o *Linux*, o qual integra facilmente as funções de virtualização assistida por hardware ao *kernel*, tornando o SO um *hypervisor* do tipo 1. Essa característica torna o KVM mais leve e performático em comparação com as demais tecnologias de virtualização (COSTA, 2021). Além do mais, o KVM é uma tecnologia de código aberto (*open-source*). Ao longo dos anos, desde a sua criação em 2006, o KVM vem ganhando uma atenção destacada na comunidade acadêmica e no mercado, conquistando

um espaço consolidado em relação aos concorrentes (COSTA, 2021).

2.3 Conteinerização

A conteinerização é definida como um mecanismo que empacota arquivos, bibliotecas e dependências de aplicações por uma tecnologia denominada de contêineres, a qual abrange características de portabilidade, escalabilidade, configurabilidade, isolamento e economicidade na utilização dos recursos. Isso permite que as distribuições dos contêineres entre os ambientes de desenvolvimento e produção sejam conduzidas com máxima agilidade nas entregas, sem causar prejuízo à operação (Nickoloff; Fisher, 2019). Diferentemente das tecnologias de virtualização tradicionais, os contêineres não exigem uma camada de emulação ou um *hypervisor* para serem executados. Em vez disso, eles utilizam uma interface no nível do SO. Essa característica os torna uma tecnologia enxuta que demanda menos sobrecarga para executá-los. Isso permite que um número maior de instâncias sejam executadas na máquina física (Turnbull, 2016). Além disso, os contêineres são carregados no espaço do usuário sobre o *kernel* do SO. Tal característica garante uma redução considerável na utilização dos recursos e aumenta o desempenho das tarefas executadas nos ambientes baseados em contêineres, visto que as instâncias são executadas diretamente no SO hospedeiro sem redirecionar as instruções para o *hypervisor* (Cuadrado-Cordero et al., 2017).

Os contêineres armazenam apenas os arquivos necessários à execução de um ou mais processos empacotados por uma imagem inicializada isoladamente no SO hospedeiro. A imagem inicializada no contêiner difere de um SO inicializado na VM, visto que uma imagem possui apenas as dependências e os binários necessários para carregar os serviços e os artefatos da aplicação, construído a partir dos códigos fontes. Ao contrário das VMs que dependem do *hypervisor* para realizar as tarefas de gerenciamento, desde alocação dos recursos, criação, inicialização, desligamento até a destruição delas (Nickoloff; Fisher, 2019). Os contêineres conseguem implantar ambientes praticamente em tempo de execução. Isso garante mais agilidade em todos os estágios no desenvolvimento da aplicações, desde a construção de ambientes de testes ou homologação até a entrega para o ambiente de produção. Os contêineres oferecerem características perfeitamente compatíveis na composição de ambientes baseados na arquitetura de microsserviços, a qual preconiza a

separação das regras de negócios da aplicação em diversos componentes (processos) menores que realizam as tarefas em conjunto. Essa abordagem valoriza que os processos possuam baixa granularidade, agilidade de implantação e comunicação assíncronas entre processos semelhantes de várias aplicações (Turnbull, 2016). A Figura 2 mostra um comparativo entre as VMs (a) e os contêineres (b). Considerando que um hardware subjacente é utilizado para ambos os cenários, as camadas de softwares para um ambiente que adota a virtualização baseada em VMs depende de seus respectivos SOs convidados e um *hypervisor* para redirecionar as instruções entre o hardware subjacente e as VMs. Enquanto, o ambiente que adota a conteinerização é composto pelo *kernel* do SO hospedeiro, o *Docker Engine* e as imagens base que empacotam os processos, as bibliotecas, as dependências e os binários da aplicação. O *Docker* é responsável pela operação dos ambientes baseados em contêineres oriundos das imagens.

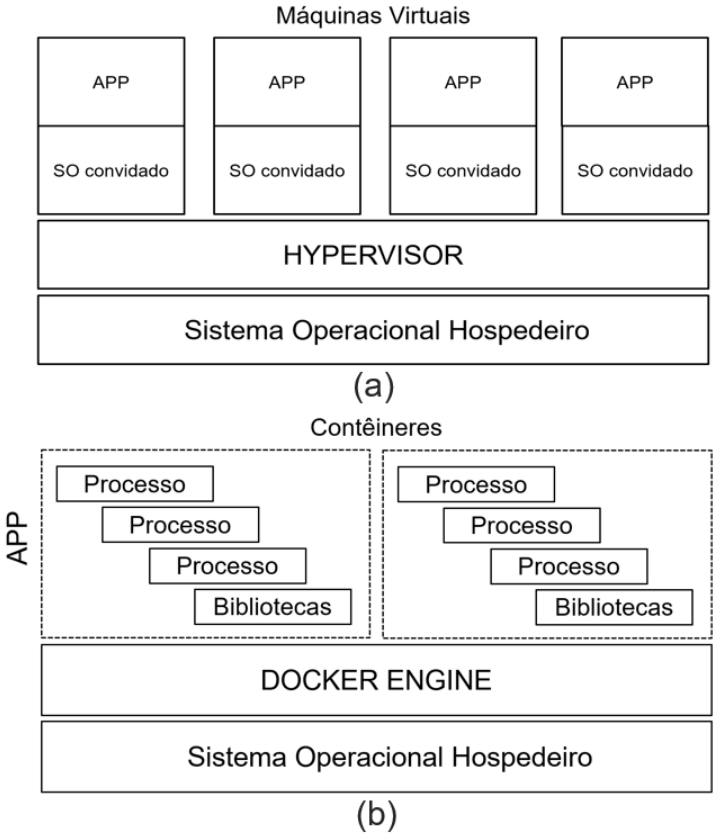

Figura 2 – Exemplos de ambientes utilizando a máquinas virtuais (a) e contêineres (b).

O *Docker Engine* é uma ferramenta capaz de realizar virtualização a nível de sistema operacional, popularmente conhecida como conteinerização. O *Docker* possibilita o empacotamento de aplicações ou ambientes completos dentro um ou mais contêineres,

tornando-os portáveis para qualquer outro ambiente baseado na tecnologia. Essa característica permite a criação, o teste e a implantação de aplicações rapidamente, reduzindo drasticamente o tempo de *deployment* (Turnbull, 2016). O *Docker* oferece uma API com variadas funcionalidades que podem ser manipuladas via arquivos de especificação ou interface de linha de comando (CLI - *command-line interface*) que permitem acessar, criar, inicializar, parar ou destruir contêineres, bridge de rede, volumes de armazenamento, imagens entre outros. No caso das imagens, os comandos podem construí-las (ou reconstruí-las), renomeá-las, apagá-las, baixá-las ou compartilhá-las em repositórios locais ou remotos, bem como públicos ou privados (Nickoloff; Fisher, 2019). Outro funcionalidade do *Docker* é a criação de imagens a partir de instruções predefinidas em arquivos denominados de *Dockerfiles*. Esse arquivo pode conter instruções relacionadas aos parâmetros de configuração do ambiente, bem como os artefatos da aplicação (Nickoloff; Fisher, 2019). Adicionalmente, o *Docker* implementa uma funcionalidade de orquestração que permite provisionar rapidamente um lote de contêineres que contemplem uma pilha completa de uma ou várias aplicações. Esse processo ocorre de forma de automatizada através de uma extensão denominada de *Docker Compose*. Essas pilhas são implementadas com base em conjunto de especificações definidas em arquivos no formato YAML (yml) cujos conteúdos representam todos os parâmetros de configuração necessárias para inicializar automaticamente o ambiente da aplicação (Nickoloff; Fisher, 2019; Turnbull, 2016).

As aplicações baseadas no AVA Moodle tendem a receber elevadas quantidades de requisições dada a sua vasta adesão pelas instituições de ensino em virtude de oferecer experiências positivas aos alunos. No entanto, equilibrar a capacidade da infraestrutura com a demanda dos usuários sem causar queda no desempenho ou desperdícios de recursos computacionais e financeiros é uma tarefa complexa. É importante balancear a experiência positiva dos alunos com a redução de excessivos de custos para manter a qualidade na entrega dos serviços, eliminando eventuais prejuízos injustificáveis para os negócios. Assim, as redes de Petri estocásticas podem ser adotadas para auxiliar na modelagem e análise de cenários de acordo determinadas cargas de trabalhos que melhor garantam os trade-offs de desempenho, consumo de energia e custo.

2.4 Modelagem e análise utilizando redes de Petri

O conceito das RdP foi idealizado pelo cientista Carl Adam Petri através de sua tese de Doutorado, intitulada como Comunicação com Autômatos, apresentada no ano de 1962, na Universidade de Bonn, na Alemanha (PETRI, 1962). As RdP podem ser definidas como um conjunto de formalismos adotados para representar graficamente os estados de sistemas cujos comportamentos são caracterizados pela: concorrência, paralelismo, assincronicidade, distribuição, determinismo (MURATA, 1989; MARSAN et al., 1994). Desde então, esse formalismo tem sido amplamente utilizado em diferentes áreas, tais como Ciência da Computação, Engenharia Elétrica, Administração, Química, entre outras. Desde sua criação, diversas extensões propostas tiverem o intuito de adicionar novas características suportadas por esse formalismo. Algumas extensões permitiram que às redes suportassem eventos com tempos, classificando elas como redes de Petri temporizadas e estocásticas. Outras extensões adicionaram mecanismos de alto nível, como *tokens* coloridos, objetos ou a noção de hierarquia (MURATA, 1989). A inclusão do tempo nas redes de Petri ampliou o seu poder de análise, possibilitando a obtenção de métricas de avaliação de desempenho e dependabilidade. Essas variantes surgiram devido à necessidade de suprir as diferentes áreas de aplicação, por exemplo sistemas de apoio à decisão ou planejamento de capacidade (REISIG, 2014).

2.4.1 Redes de Petri Elementar

O tipo básico de redes de Petri é denominado de *Place/Transition* que também pode ser definida pelo termo rede de Petri elementar, ou simplesmente rede de Petri (REISIG, 2014). A notação formal das RdP fornece um conjunto de elementos gráficos que constroem modelos de abstrações de sistemas complexos. Atrelado aos elementos gráficos, as RdP possuem uma base matemática consolidada por equações de estados e algébricas que servem para demonstrar todo o comportamento do sistema (BALBO, 2007; SILVA et al., 2013). Embora, as RdPs possuam um forte alicerce na matemática para construir os modelos que representam os sistemas complexos, as demostrações matemática podem tornar as análises numéricas menos intuitivas. Dessa forma, as representações de sistemas expressos nos elementos gráficos da RdP facilitam uma melhor compreensão entre as atividades de cada componente do sistema (REISIG, 2014). Além disso, ao longo dos anos, desde a

primeira versão das RdP (PETRI, 1962) até hoje, elas vêm recebendo várias atualizações através de contribuições propostas por outros pesquisadores, incluindo características de temporização, orientação a objetos, estrutura de dados, entre outras (BALBO, 2007). As RdP são constituídas por 4 elementos gráficos básicos, tais como lugar, transição, arco e *token* (marca). Os lugares são representados por círculos, as transições por retângulos pretos, os arcos por setas e os *tokens* por pontos (ver Figura 3).

Os lugares (Figura 3 (a)) são repositórios utilizados para armazenar os *tokens* que, por sua vez, indicam marcações para representar o estado do sistema em determinado instante. As transições (Figura 3 (b)) são graficamente representadas por barras que atuam como elementos ativos da rede, ou seja, as transições são ações (ou eventos) realizadas para alterar o estado do sistema. Uma transição deve estar habilitada se todas as suas pré-condições estiverem satisfeitas. Uma vez que a transição satisfaz todas as pré-condições, ela poderá efetuar o disparo dos *tokens*. Essa ação remove uma determinada quantidade de *tokens* de lugares de origem e armazena-os em outros lugares de destino, através de arcos conectados entre as transições, gerando então as pós-condições. As relações entre os lugares e as transições são especificadas por arcos (Figura 3 (c)) que são representados graficamente por setas que indicam a direção dos *tokens* entre os lugares. Os arcos de entrada conectam as transições aos lugares para representar a ocorrência de eventos no sistema, modificando os números de *tokens* contidos nos lugares. Dessa forma, os arcos representam o fluxo de passagem dos *tokens* pela rede. Os *tokens* (Figura 3 (d)) são representados graficamente por círculos pequenos preenchidos que residem nos lugares para especificar o estado da RdP. Os *tokens* distribuídos entre os lugares da rede determinam o estado em que o sistema se encontra em determinado instante. Após o disparo de uma transição, lugares têm suas informações alteradas *(tokens)*, ou seja, ocorrerá uma pós-condição para que outras transições do modelo possam ser habilitadas para disparar os demais eventos do sistema (BALBO, 2007; MURATA, 1989; MARSAN et al., 1994; SILVA et al., 2013).

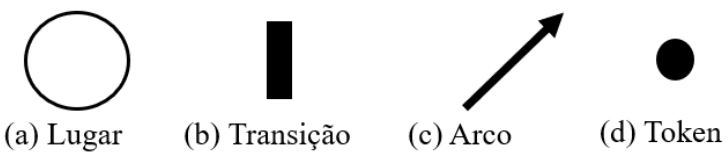

Figura 3 – Elementos da rede de Petri

A representação formal de uma rede de Petri é uma 5-tupla descrita pela álgebra $PN = \{P, T, F, W, M_0\}$, a saber:

- $P = \{p_1, p_2, p_3, ..., p_n\}$ é um conjunto finito de lugares;
- $T = \{t_1, t_2, t_3, ..., t_n\}$ é um conjunto finito de transições;
- $F \subseteq (P \times T) \cup (T \times P)$ é um conjunto de arcos;
- $W : F \rightarrow \{1, 2, 3, ...\}$ é uma função de peso do arco;
- $M_0 : P \rightarrow \{1, 2, 3, ...\}$ é a marcação inicial, $P \cap T = \emptyset$ e $P \cup T = \emptyset$

A Figura 4 mostra uma representação gráfica de um modelo que pode assumir dois tipos estados (ON e OFF). Consequentemente, o modelo é composto pelos lugares `ON` e `OFF` que representam os estados do sistema. As transições `Ligar` e `Desligar` representam os eventos que alteram os estados do sistema. A Figura 4 (a) mostra o estado inicial do sistema onde o lugar `ON` contém um `token`. O arco saindo do lugar `ON` em direção à transição `Desligar`, a torna apta para o disparo em virtude do `token` armazenado no lugar `ON` satisfazer a precondição de habilitação dela. O modelo mostrado na Figura 4 (b) representa o outro possível estado assumido pelo sistema quando ocorrer o disparo do `token` pela transição `Desligar` em direção ao Lugar `OFF`. Nesse caso, a transição `Desligar` ficará inabilitada e o `token` armazenado no lugar `OFF` habilitará o disparo da transição `Ligar`, tornando-a apta a retirar um `token` do lugar `OFF` e armazená-lo novamente no lugar `ON`.

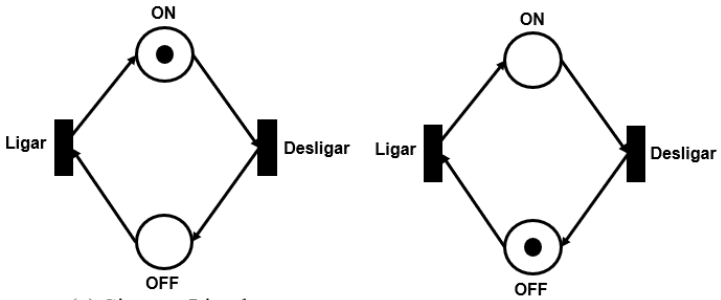

Figura 4 – Exemplo de uma rede de Petri

2.4.2 Redes de Petri Estocásticas

A SPN (*Stochastic Petri Net*) é uma variante das redes de Petri que propõem a incorporação de tempo associados aos atrasos das transições. Dentre os modelos

estocásticos, as SPNs possuem uma destacada adesão em vários campos do conhecimento devido a sua extensa aplicabilidade nos contextos da avaliação de desempenho, da disponibilidade e da dependabilidade (MURATA, 1989). Com as SPNs, tornou-se possível a atribuição de tempos às transições para que fosse possível modelar eventos temporizados intervalares, determinísticos, não determinísticos ou estocásticos (SILVA et al., 2013). Nessas redes, destacam-se as transições imediatas e as exponenciais. As imediatas são caracterizadas por tempos de disparos igual a zero ou nulo, ao passo que as exponenciais são caracterizadas por tempos de disparo atribuídos à variáveis aleatórias com distribuição exponencial (MARSAN et al., 1994). Na perspectiva de desempenho, as transições exponenciais possuem uma maior relevância na representação de eventos do que as transições imediatas, as quais representam eventos sem gerar atrasos nas ações do modelo (GERMAN, 2000).

Com a necessidade de analisar eventos ocorridos em função do tempo, as transições temporizadas tiverem a atribuição de um novo conceito denominada de grau de habilitação, o qual define o número de vezes que uma transição pode ser disparada conforme uma quantidade marcações (*tokens*), antes de se tornar desabilitada. Dessa forma, as transições temporizadas podem representar eventos que ocorrem paralelamente por unidade de tempo. Isso é chamado de semânticas de temporização, as quais se dividem em três tipos comuns, como mostra a seguir. Na *Single-server (SS)*, o tempo de disparo é contado quando a transição é habilitada, após o disparo da transição um novo tempo será contado se a transição ainda estiver habilitada. Sendo assim, apenas um token é disparado por vez, ou seja, a capacidade de um lugar/transição é 1. Na *Multiple-server*, todo o conjunto de *tokens* é processado em paralelo até o máximo grau de paralelismo (k) definido para essa semântica. Assim, é possível fazer k disparos por vez, quando a capacidade de um lugar/transição for um k inteiro. Na *Infinite-server (IS)*, todo o conjunto de *tokens* da transição habilitada é processado paralelamente na mesma janela de tempo, ou seja, é possível executar infinitos disparos de uma única vez. A Figura 5 (a) corresponde à transição estocástica cujo elemento gráfico é representado por um retângulo branco. Nesta transição é possível atribuir tempos exponencialmente distribuídos relacionados a cada evento do sistema. Ao passo que, a Figura 5 (b) corresponde à transição imediata cujo elemento gráfico é representado por um retângulo preto fino. Essa transição não possui o atributo de tempo associado ao disparo de transições. Porém, possui prioridade sobre

as transições temporizadas para realizar os disparos (BALBO, 2007; MURATA, 1989; MARSAN et al., 1994; SILVA et al., 2013).

(a) Transição Estocástica (b) Transição Imediata

Figura 5 – Elementos da GSPN

De acordo com (GERMAN, 2000), a representação formal da SPN corresponde a uma 9-tupla descrita pela álgebra $SPN = \{P, T, I, O, H, \Pi, G, M_0, Atts\}$, onde:

- $P = \{p_1, p_2, p_3, ..., p_n\}$ é o conjunto de lugares onde n é a quantidade de lugares;
- $T = \{t_1, t_2, ..., t_m\}$ é o conjunto de transições imediatas e temporizadas, $P \cap T = \emptyset$; m é a quantidade de transições;
- $I \in (\mathbb{N}^n \to \mathbb{N})^{n \times m}$ é a matriz que representa os arcos de entrada (que podem ser dependentes de marcações);
- $O \in (\mathbb{N}^n \to \mathbb{N})^{n \times m}$ é a matriz que representa os arcos de saída (que podem ser dependentes de marcações);
- $H \in (\mathbb{N}^n \to \mathbb{N})^{n \times m}$ é a matriz que representa os arcos de inibidores (que podem ser dependentes de marcações);
- $\Pi \in \mathbb{N}^n$ é o vetor que associa o nível de prioridade a cada transição;
- $G \in (\mathbb{N}^n \to \{true, false\})^n$ é o vetor que associa uma condição de guarda relacionada à marcação do lugar a cada transição;
- $M_0 \in \mathbb{N}^n$ é o vetor que associa uma marcação inicial de cada lugar (estado inicial);
- $Atts = (Dist, W, Markdep, Policy, Concurrency)^m$ compreendem o conjunto de atributos para transições, onde:
  - $Dist \in \mathbb{N}^n \to \mathcal{F}$ é uma função de distribuição de probabilidade associada ao tempo de cada transição, sendo que $\mathcal{F} \leq \infty$. Esta distribuição pode ser dependente de marcação,
  - $W \in \mathbb{N}^m \to \mathbb{R}^+$ é a função peso, que associa um peso (w_t) às transições imediatas e uma taxa λ_t às transições temporizadas, onde:

$$W(t) = \begin{cases} w_t \geq 0, \text{ se } t \text{ é uma transição imediata;} \\ \lambda_t > 0, \text{ Caso Contrário.} \end{cases}$$

  - $Markdep \in \{constant, enabdep\}$) define se a distribuição de probabilidade associada

ao tempo de uma transição é constante *(constant)* ou dependente de marcação *(enabdep)*,

– $Policy \in \{prd, prs\}$ define a política de memória adotada pela transição (*prd - preemptive repeat different*, valor padrão, de significado idêntico à *enabling memory policy; prs -preemptive resume*, corresponde a *age memory policy*);

– $Concurrency \in \{ss, is\}$ é o grau de concorrência das transições, onde ss representa a semântica *single server* e is representa a semântica *infinite server*.

2.5 Considerações Finais

Este capítulo apresentou os principais conceitos que envolvem esta dissertação. Primeiramente, foi apresentada de forma sucinta os conceitos referentes aos AVAs e alguns aspectos importantes no processo de aprendizagem. Em seguida, foram introduzidos os principais conceitos referentes à virtualização e a conteinerização. Por fim, as redes de Petri foram introduzidas mostrando que elas são uma ferramenta bem estabelecida para modelagem e análise de vários tipos de sistemas, tais como sistemas concorrentes, assíncronos, distribuídos, paralelos, não-determinísticos e estocásticos.

3 Trabalhos Relacionados

Esta seção apresenta alguns trabalhos relacionados que têm sido desenvolvidos para a avaliação de desempenho, custo e consumo de energia de infraestruturas computacionais baseadas em contêineres e VMs. Primeiramente, iremos apresentar os relacionados ao consumo de energia. Em seguida, abordaremos os trabalhos relacionados à avaliação de desempenho. Por fim, serão apresentados trabalhos relacionados aos dois temas citados anteriormente. Tais trabalhos foram realizados com o auxílio de técnicas importantes, como experimentações, metodologias, heurísticas implementadas em algoritmos, modelos e validações. Apesar de boa parte dos trabalhos relacionados terem pontos de convergência com um ou mais temas desta dissertação, eles não cobrem na totalidade os temas explorados neste estudo.

3.1 Consumo de energia

Esta seção apresenta alguns trabalhos que abordaram a avaliação do consumo de energia de ambientes baseados em contêineres ou VMS. Sob ambas perspectivas, as pesquisas encontradas adotaram estratégias baseadas em medições por softwares e heurísticas implementadas com algoritmos de programação dinâmica.

Os autores (FIENI et al., 2020) desenvolveram um software medidor de energia chamado *Smartwatts*. O medidor pode calibrar automaticamente os modelos de potência da CPU e da memória RAM, explorando os dados das medições de energia oriundos do software RAPL, o qual converte os registros dos contadores de desempenho da CPU e da memória RAM para valores de potência. Esse processo de calibração automática conseguiu estimar um modelo de potência estável e preciso conforme intervalos de amostras de frequências operadas pelo processador. O *Smartwatts* é composto por clientes e um servidor. Os clientes monitoram os grupos de controles dos contêineres *Docker* e das VMs no *KVM*. Um servidor de banco de dados *MongoDB* armazena as amostras enviadas dos clientes. As cargas de trabalhos foram geradas pelos *benchmarks STRESS NG* e *NAS Parallel*. Nos experimentos, o *Smartwatts* conseguiu realizar medições com erros limitadas a 5 *Watts* para a CPU e 1 *Watt* para a memória RAM, considerando ciclos de *clocks* de 2 Hz. Com esses resultados, foi possível estimar potências com erros inferiores a 3 *Watts*

para a CPU e 0,5 *Watts* para a memória RAM.

No trabalho de (PHUNG et al., 2020), foi proposto um medidor de energia virtual baseado na arquitetura cliente-servidor no protocolo TCP/IP. O medidor virtual foi chamado de cWatts++ e foi capaz de estimar o consumo de energia através de algoritmos que implementam funções matemáticas usando dados oriundos dos contadores de eventos gerados pelos recursos do computador. O medidor utilizava um programa cliente que executava nos contêineres para monitorava os contadores de eventos não vinculados à CPU e a memória RAM. Um modelo matemático chamado de raplModel monitorou os eventos relacionados ao consumo de energia da CPU e Memória RAM. Esses eventos foram coletados pelo contador RAPL (*Running Average Power Limit*), o qual forneceu dados acerca dos efeitos *Jaule* dos componentes para serem convertidos em unidades de potência que definiram a granularidade da energia consumida. Os autores realizaram experimentos utilizando cinco diferentes tipos de *Benchmarks* para produzir cargas de trabalhos no computador hospedeiro. As medições utilizaram instâncias com seis frequências de CPU: 0,8 GHz, 1,5 GHz, 2,2 GHz, 2,8 GHz, 3,1 GHz e 3,3 GHz. Os resultados obtidos pela ferramenta desenvolvida foram comparadas com as medições reais do medidor *Cabac Power-Mate*, apresentando um erro menor que 5%.

Os autores (ZHENG et al., 2020) adotaram uma serie de algoritmos visando melhor aproveitar a distribuição dos contêineres conforme o percentual de utilização da CPU dos nós alocados para um *cluster* criado pelo simulador de nuvem *CloudSim4.0*. A ideia central foi diminuir a frequência de migrações dos contêineres entre os nós para provocar uma queda no consumo de energia. Os experimentos consideraram três passos implementados por diferentes grupos algoritmos. No primeiro passo, a seleção da origem adotou cinco algoritmos com métodos estatísticos que coletaram periodicamente amostras dos percentuais mínimos e máximos de utilização das CPUs a fim de eleger os nós mais adequados para migrar os contêineres. No segundo passo, a seleção do contêiner para migração utilizou o algoritmo de correlação máxima para identificar os contêineres com maiores percentuais de utilização da CPU e depois migrá-los para outros nós com maiores disponibilidades de recursos. No terceiro passo, a seleção do destino avaliou cinco algoritmos de escalonamento para distribuir os contêineres conforme os critérios anteriores. Os algoritmos *Load Concentration (LC)* e *Resource Load Balancing (RLB)* apresentaram uma redução de aproximadamente 2% no consumo de energia. Os resultados mostraram

que as VMs demandaram cerca de 28% a mais no consumo de energia. Enquanto, os contêineres demandaram apenas 1,3% a mais no consumo de energia.

Em (Chen et al., 2019), foi utilizado o simulador de nuvem *CloudSim4.0* para avaliar uma estratégia de distribuição de instâncias baseadas em contêineres dentro de VMs. Os autores desenvolveram um algoritmo que cria uma lista ordenada que prioriza as VMs mais ociosas em detrimento das VMs menos ociosas, observando a taxa de utilização da CPU e da memória RAM. O algoritmo realiza varreduras repetitivas nas listas de classificação conforme o percentual uso da CPU e da Memória tanto das VMs quanto dos contêineres. A estratégia criada aproveita a disponibilidade dos recursos nas VMs para suportar o máximo possível de contêineres sem comprometer o desempenho geral da operação. Os resultados apresentaram uma otimização da utilização dos recursos e reduziu o consumo de energia em 12,8%.

3.2 Avaliação de Desempenho

A maioria dos trabalhos encontrados na literatura tiveram o foco nas análises de desempenho dos contêineres e das VMs. Boa parte das pesquisas adotaram diversas categorias de *benckmarks* para conduzir os experimentos em ambientes implantados por ambas tecnologias. As análises foram auxiliadas por técnicas estatísticas e modelagem matemática com a finalidade de encontrar as melhores soluções para os cenários propostos.

Em (LIN et al., 2018), foram propostos algoritmos baseados na heurística gulosa e na programação dinâmica de propósitos diferentes para encontrar a solução ideal na distribuição dos contêineres entre máquinas físicas. O algoritmo utilizou o percentual de ociosidade da CPU para realizar tal distribuição. A alocação de K núcleos da CPU com frequências F para N contêineres são tarefas complexas que requerem tempos de execução exponenciais. Os experimentos conduzidos consideraram cenários com 10 contêineres e 5 servidores. A heurística *Consecutive Allocation* se destacou nos experimentos, levando 0,123 segundo para encontrar o consumo de energia ideal. Enquanto, o algoritmo de programação dinâmica e a heurística gulosa *decelerate* levaram 1,824 e 0,990 segundos, respectivamente. Já a heurística gulosa *decelerate* apresentou melhores resultados para cenários com requisitos de escalabilidade.

Os autores (Yadav et al., 2018) propuseram uma metodologia para comparar o

desempenho das VMs *VMWare* e dos contêineres *Docker*. Os ambientes construídos por VMs e contêineres receberam cargas de trabalhos de um *benckmark* denominado *sysbench* que, posteriormente, forneceu resultados sobre o tempo de execução da CPU e da Memória RAM. Os experimentos tiveram variações de cinco cenários com quantidade limitadas a cinco instâncias baseadas em contêineres e VMs. O tempo de execução da memória foi avaliado através de transferências de arquivos com tamanhos variados entre 40GB, 60GB, 80GB, 100GB e 120GB. O tempo de execução da CPU foi observado através dos cálculos de números primos com intervalos variados entre 1-20.000, 1-30.000, 1-40.000, 1-50.000 e 1-60.000. Os resultados demonstraram que o desempenho dos contêineres sobre as VMs representaram diferenças médias de 1,1% e 0,69% no tempo de execução da memória e da CPU, respectivamente. Assim, a diferença do tempo de execução entre os contêineres e as VMs sobre os cenários apresentados não indicaram uma superioridade significativa de ambas tecnologias.

Em (Salah et al., 2017), foi abordada uma avaliação de desempenho utilizando a infraestrutura como serviço (*Infrastructure as a service*) da *Amazon Cloud Plataform* para distribuir serviços em contêineres e VMs. As duas tecnologias executaram os serviços *Web* dentro dos ambientes da *Amazon EC2* e *Amazon ECS*, respectivamente. O ambiente de experimentação foi estruturado em três cenários com variações de escopos contendo uma, duas e três instâncias de servidores *Apache Web*. A análise de desempenho dos serviços *Web* baseados em contêineres e VMs levaram em consideração as seguintes métricas: vazão, tempo de resposta e utilização da CPU. As cargas de trabalhos foram requisições enviadas simultaneamente aos servidores web durante intervalos de 20 segundos. O *benchmark JMeter* simulou grupos de usuários virtuais contendo 10 requisições simultâneas. Os resultados da vazão e do tempo de resposta mostraram melhores desempenhos das VMs avaliadas nos três cenários. O cenário 1 observou a operação de um servidor Web carregado no contêiner e outro na VM. Os resultados da vazão e do tempo de resposta observados no cenário 1 indicaram uma diferença de 20% favor da VM. Posteriormente, o cenário 2 observou a operação dois servidores Web carregados em cada contêiner e VMs. Os resultados da vazão e do tempo de resposta observados no cenário 2 indicaram uma diferença de 50% favor da VM. Por fim, os resultados do cenário 3 apresentaram uma diferença de 27,27% a favor de VM, considerando três servidores carregados nos carregados contêineres e VMs. O percentual de utilização da CPU chegou a 100% considerando

todos cenários. É importante destacar que os contêineres do *Amazon ECS* operam sobre VMs gerenciadas por *hypervisors*. Por essa razão, a inferioridade no desempenho dos contêineres está atrelada aos *overheads* adicionais provocados pelas VMs que hospedaram os contêineres

Os autores (Bhimani et al., 2017) analisaram e compararam o desempenho de aplicativos de *Big Data* executados por máquinas virtuais e contêineres. O *Apache Spark* consiste em um mecanismo de computação para *clusters* cuja finalidade é processar dados compartilhados e escaláveis em nuvens de *Big Data*. Os *clusters*, que são formados pelos Nós do *Spark*, suportam a execução de várias tarefas paralelas, visando concorrer os recursos computacionais dos núcleos das CPUs, memórias e discos rígidos de todo sistema computacional. Os *benchmarks* denominados como *K-Means*, *PageRank* e *SQLDataSource*, além de outros 11 *benchmarks*, tiveram a função de avaliar o desempenho de aplicações classificadas como: aprendizagem de máquinas, computação de gráfica e consultas em SQL, respetivamente. As medições adotaram dois *clusters* distintos compostos por 8 instâncias baseadas em contêineres e VMs. Os experimentos observaram o percentual de utilização da CPU, do disco rígido e da memória em três cenários conforme a classificação do *benchmark*. Os experimentos de aprendizagem de máquinas aplicados aos *cluster* dos contêineres resultaram em 3,67%, 13% e 100% da utilização da CPU, do disco rígido e da memória, respectivamente. Ao passo que, os mesmos testes aplicados aos *clusters* das VMs resultaram em 16,17%, 13,83% e 71,00% da utilização da CPU, do disco rígido e da memória, respectivamente. Os experimentos de computação de gráfica aplicados aos *cluster* dos contêineres resultaram em 5,80%, 19,20% e 92,20% da utilização da CPU, do disco rígido e da memória, respectivamente. Enquanto, os mesmos testes aplicados aos *clusters* das VMs resultaram 23,80%, 23,60% e 72,80% da utilização da CPU, do disco rígido e da memória, respectivamente. Por fim, os experimentos de consultas em SQL aplicados aos *cluster* dos contêineres resultaram em 1,5%, 11% e 100% da utilização da CPU, do disco rígido e da memória, respectivamente. Ao passo que, os testes os mesmos aplicados aos *clusters* das VMs resultaram em 17,50%, 12,00% e 83% da utilização da CPU, do disco rígido e da memória, respectivamente. Os experimentos conduzidos nos contêineres apresentaram resultados melhores para a maioria dos *benchmarks* executados. Todavia, o *benchmark K-Means* demonstrou uma diferença marginal no percentual de utilização da CPU a favor das VMs. Observou-se que os testes de *K-Means* realizados no

cluster das VMs utilizaram 2% da CPU, ao passo que o *cluster* dos contêineres utilizaram 5% da CPU.

Em (Barik et al., 2016), foi apresentada uma avaliação de desempenho entre virtualização e conteinerização através do *Oracle Virtual Box* e o *Docker*. O estudo comparou as duas tecnologias através de medições realizadas pelos *benchmarks AIO Stress* e o *RAM-Speed* que avaliou a largura de banda de leitura e de escrita nas memórias do tipo: *cache* de nível I, *cache* de nível II e a *Random Access Memory* (Memória RAM). Outro *benchmark* utilizado foi o *IO ZONE* que criou conjuntos de arquivos para gerar cargas de trabalhos no disco rígido e, consequentemente, avaliar a largura de banda na leitura e escrita. O desempenho da rede foi medido através das transferências de arquivos via os protocolos *Transmission Control Protocol* (TCP) e *User Datagram Protocol* (UDP) entre instâncias virtuais de servidores e clientes que executaram os *benchmarks iperf* e *tbench* para produzir cargas de trabalhos e mostrar relatórios da vazão da rede. O *benchmark RuBBos* gerou as requisições HTTP para servidores *Apache* com banco de dados, e depois gerou relatórios com o tempo de conexão e a taxa máxima das requisições atendidas. A vazão em MB/s alcançada na operação de escrita da memória alocada para as duas tecnologias atingiu uma diferença de 149% a favor do contêiner. No entanto, a vazão da operação de leitura entre as duas tecnologias apresentou uma leve diferença 1,73% a favor do contêiner. O teste de desempenho da rede utilizou a interface *loopback* para observar o tempo gasto na transferência de um arquivo limitado a 10 gigabytes. O tempo gasto na transferência local do arquivo apresentou uma diferença de 51,74% a favor da VM. O tempo gasto para realizar o teste de requisições HTTP apresentou uma diferença de 83,04% a favor do contêiner. Assim como, o tempo gasto para realizar o teste de renderização de páginas PHP apresentou uma diferença de 112,83% a favor do contêiner. No entanto, a VM superou o contêiner em 58,01%, quando o teste de requisições HTTP incluiu uma criptografia com cifra em bloco e chaves simétricas do tipo AES *(Advanced Encryption Standard)* de 256 bits. No teste de emissão de certificados SSL, o contêiner emitiu um valor percentual de 126,64% a mais do que a VM. As avaliações concluíram que o contêiner superou VM na maioria dos experimentos, exceto nos testes de transferências de arquivos via interface *loopback* e no processamento das requisições HTTP com criptografia simétrica, nos quais das VMs apresentaram melhores resultados.

3.3 Avaliação de Desempenho e de Consumo de energia

Esta seção descreve as pesquisas que conciliaram os temas avaliação de desempenho e consumo de energia aplicados a ambientes baseados em contêineres e VMs. Na perspectiva de ambas tecnologias, o estado da arte apresentou uma quantidade reduzida de referências. Para conduzir os experimentos, os trabalhos encontrados adotaram *benchmarks* e medidores baseados em software e hardware. As análises e demonstrações foram conduzidas com técnicas estatísticas e modelagem matemática.

Os autores (Cuadrado-Cordero et al., 2017) compararam o desempenho entre VMs (gerenciadas pelo KVM) e contêineres *Docker*. O trabalho focou na eficiência energética e na qualidade de serviço (QoS) para uma infraestrutura em nuvem. O tempo de resposta foi a principal métrica utilizada para avaliar o QoS da rede entre as duas tecnologias. Os resultados mostraram que os contêineres *Docker* apresentaram um ganho de 26% no tempo de resposta, se comparados aos resultados das VMs. Com relação ao consumo de energia, os resultados comprovaram que o *Docker* consome menos energia do que o KVM. Todavia, os experimentos de consumo de energia não apresentaram resultados concretos para responder questões relacionadas aos custos financeiros envolvidos na operação. Além do mais, os experimentos não incluíram a vazão para relacionar com o tempo de resposta e o consumo de energia.

Em (Brondolin et al., 2018), foi proposta uma ferramenta de monitoramento denominada de *DEEP-mon*, a qual é capaz de realizar medições de consumo de energia atribuídas aos contêineres em execução no SO hospedeiro. A ferramenta avalia os *trade-offs* entre a potência e o desempenho de um ambiente implantando pelo *Kubernetes*. Esse estudo adotou duas categorias de *benchmarks* onde cada uma deles foram compostos por três tipos de testes, os quais observaram o consumo de energia conforme cargas de trabalhos geradas pelos *benchmarks Embarrassingly Parallel* (EP), *Multi Grid* (MG) e *Conjugate Gradient* (CG), os quais fazem parte da suite *NAS Parallel Benchmark* (NPB). Enquanto os desempenhos foram observados pelos *benchmarks pts/Apache, pts/Nginx e pts/Postmark*, os quais fazem parte da suite *Phoronix Test*. A metodologia avaliou a correlação linear entre o consumo de energia e o desempenho através das amostras obtidas pelas medições dos contadores RAPL.

No trabalho de (Tadesse et al., 2017), foi realizada uma avaliação de consumo de energia para ambientes virtuais baseados em contêineres *Docker*. O objetivo da pesquisa

foi apresentar resultados sobre o percentual de utilização das CPUs e do impacto no consumo de energia quando servidores de contêineres são submetidos a elevadas taxas de transferências de arquivos na rede. O trabalho realizou experimentos considerando diferentes cenários de cargas de trabalhos que produziam transferências de dados para exaurir a capacidade total das interfaces de rede dos contêineres. Os resultados mostraram que o aumento na demanda da banda da rede está associado ao percentual de utilização da CPU e, por consequência, resultou em um aumento no consumo de energia do computador hospedeiro.

A Tabela 1 foi criada a partir dos levantamentos feitos nos trabalhos relacionados apresentados nesta seção. A tabela apresenta as quantidades de contribuições cobertas por todas as pesquisas, incluindo os pontos em comuns delas com esta dissertação. Apesar das importantes contribuições de todos os trabalhos, observou-se que a proposta desta dissertação conseguiu agregar todos os temas expostos na Tabela 1 em único estudo. Além do mais, não foram encontradas trabalhos relacionados às infraestruturas AVA Moodle submetidas a estudos de modelagens de SPN com ênfase em avaliação de desempenho, consumo de energia e custo. Os temas menos cobertos foram modelos, validação, metodologia, aplicações em ambientes reais e custo, sendo esse último o menos explorado em conjunto com outras temas. Dessa forma, a importância desta dissertação será reunir diversos temas de variados trabalhos em um único trabalho.

3.4 Considerações Finais

Os trabalhos apresentados anteriormente apresentaram estratégias para avaliar o desempenho e o consumo de energia de contêineres e VMs através de softwares de medição, modelos matemáticos e algoritmos. Os estudos relacionados à avaliação de desempenho apresentaram resultados mais favoráveis aos contêineres quando os *benchmarks* demandam recursos da CPU, da memória e do disco rígido. A maioria dos experimentos também indicaram tempos de respostas menores a favor dos serviços instanciados nos contêineres. É importante destacar que essas pesquisas sobre avaliação de desempenho realizaram experimentos através de clientes instanciados nos próprios computadores hospedeiros, os quais podem não representar aplicações reais onde as cargas de trabalhos são geradas por clientes externos interligados a enlaces de rede. Além do mais, a maioria dos ambientes

Tabela 1 – Relação entre a proposta desta dissertação e outros trabalhos relacionados

	Avaliação de Desempenho	Avaliação de Consumo de Energia	Avaliação de Custo	Propõe uma Metodologia	Modelos Analíticos	Validação dos Dados	Ambientes em Contêineres	Ambientes em Máquinas Virtuais	Aplicação em Ambiente Real
Esta dissertação	✓	✓	✓	✓	✓	✓	✓	✓	✓
(FIENI et al., 2020)		✓		✓	✓	✓	✓		
(PHUNG et al., 2020)		✓			✓	✓	✓		
(ZHENG et al., 2020)		✓			✓	✓	✓	✓	
(Chen et al., 2019)		✓			✓	✓	✓	✓	
(LIN et al., 2018)	✓				✓		✓		
(Yadav et al., 2018)	✓			✓			✓	✓	
(Brondolin et al., 2018)	✓	✓					✓		
(Cuadrado-Cordero et al., 2017)	✓	✓		✓			✓	✓	
(Salah et al., 2017)	✓						✓	✓	✓
(Bhimani et al., 2017)	✓						✓	✓	
(Tadesse et al., 2017)	✓	✓			✓		✓		
(Barik et al., 2016)	✓						✓	✓	

baseadas em contêineres foram construídos sobre computadores hospedeiros virtualizados nas VMs de nuvens públicas, que pode gerar distorções nos resultados experimentais. Os estudos relacionados ao consumo de energia também foram mais favoráveis aos contêineres. Todavia, boa parte desses estudos apresentaram medidores baseados em softwares para estimar a potência demandada através de contadores de desempenho implementados por sensores embutidos nos componentes da placa mãe, como CPU, memórias RAM e Cache. Apesar da maior parcela da potência ser demandada pela CPU e memória, tal estratégia de medição via software deixa lacunas em relação à precisão exata da energia demandada pelo computador hospedeiro, visto que um sistema computacional possui outros componentes que impactam no consumo de energia global da infraestrutura. Sendo assim, essas lacunas levantadas justificam mais trabalhos acerca da avaliação de desempenho, consumo de energia e custo das operações baseadas em contêineres e VMs. Diferentemente dos trabalhos

apresentados, este trabalho apresenta uma abordagem baseada em modelos e experimentos para avaliar o desempenho, o consumo de energia e o custo de ambientes baseados em contêineres e VMs. Além disso, o trabalho propõe modelos estocásticos baseados em redes de Petri para auxiliar o planejamento de capacidade de ambientes virtualizados considerando métricas como vazão, tempo de resposta e consumo de energia.

4 Metodologia

Este capítulo apresenta a metodologia utilizada para avaliar o desempenho, o consumo de energia e o custo de instâncias computacionais baseadas em contêineres e VMs utilizando um AVA.

4.1 Visão Geral

A metodologia proposta faz o uso de experimentos e modelos SPNs para avaliar as métricas de desempenho, consumo de energia e custo, sendo a mesma composta por nove etapas que orientam o processo, a saber: entendimento da aplicação, seleção das tecnologias, definição das métricas, escolha das ferramentas de monitoramento e coleta, experimentação no ambiente real, geração do modelo analítico, validação do modelo, geração de cenários e análises dos resultados. A seguir, conforme descreve a Figura 6, serão discutidas as etapas e as ferramentas utilizadas para auxiliar na realização de cada uma delas.

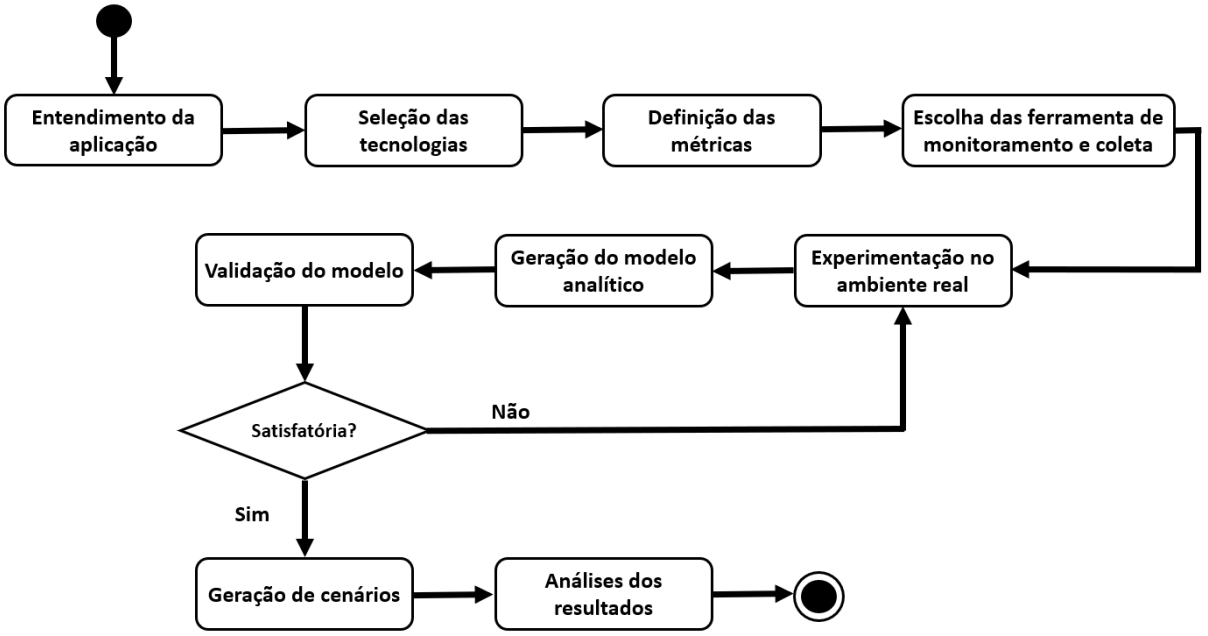

Figura 6 – Metodologia Adotada

Entendimento da aplicação: dada a grande adesão das plataformas baseadas em AVA na educação, esta etapa visa o estudo aprofundado do Moodle (DEHON et al., 2018), que é uma das ferramentas de ensino online mais utilizadas para ministrar aulas de cursos

oferecidos pelas instituições públicas e privadas. O objetivo deste estudo é compreender as funcionalidades essenciais da ferramenta, bem como todas as etapas de sua implantação, de modo a construir um ambiente similar a uma plataforma de ensino à distância (EAD).

Seleção das tecnologias: nesta etapa, são identificadas todas as tecnologias necessárias para construir o ambiente do Moodle e distribuí-lo através de instâncias baseadas em contêineres e VMs. Neste momento, são feitos estudos prévios acerca de todas as tecnologias envolvidas no processo de montagem e configuração dos ambientes experimentais. As tecnologias estudadas compreendem medidores de potência elétrica, roteadores, sistemas operacionais, *hypervisors*, *containers engines*, sistemas de orquestração, serviços web, serviços de banco de dados, *proxy* reverso, *load balancers*, *benchmarks*, analisadores de desempenho de recursos computacionais e de tráfego de rede.

Definição das métricas: as análises realizadas nesta dissertação consideraram métricas relacionadas ao desempenho, ao consumo de energia e ao custo financeiro. No aspecto de desempenho, a utilização de CPU, a vazão e o tempo de resposta são as métricas escolhidas para obter os resultados deste estudo. A utilização de CPU é observada considerando os valores percentuais atribuídos às variáveis de sistema *User, Sys, Idle* e *Busy*. Para o consumo de energia, a métrica escolhida é o kilowatt-hora variados em períodos mensais e anuais. Essa métrica será referenciada para derivar o custo da energia demandada pelos experimentos observados em cada cenário. Dessa forma, os diferentes cenários permitirão observar os comportamentos dessas métricas durante as operações do Moodle distribuídos pelas instâncias baseadas em contêineres e VMs. Adicionalmente, o modelo terá flexibilidade para calcular outras métricas baseadas nas fórmulas da Lei de Little (LITTLE, 1961) ou customizadas conforme a necessidade do projeto.

Escolha das ferramentas de monitoramento e coleta: para realizar os experimentos em um ambiente real, é necessário utilizar *benchmarks* ou ferramentas que produzam cargas de trabalhos, monitorem o desempenho dos recursos, coletem as amostras referentes à vazão da aplicação e meçam o consumo de energia. Os *benchmarks* e as ferramentas adotadas neste trabalho são descritas a seguir:

a) **Ifstat Network Monitor:** é uma ferramenta de monitoramento de tráfego de rede (IFSTAT, 2020). O *ifstat* é utilizado para monitorar o desempenho da interface *Gigabit Ethernet* do computador que hospeda as instâncias de contêineres e VMs, a fim de identificar eventuais gargalos na rede.

b) **Iperf:** é o *benchmark* que foi selecionado para estressar os contêineres e as VMs através das cargas de trabalhos geradas pelos clientes. Adicionalmente, o *iperf* estabeleceu conexões entre os clientes e os servidores para gerar trafego na rede. Consequentemente, essa ferramenta também foi utilizada para medir e coletar amostras de ambientes provisionados pelas duas tecnologias.

c) **Nigel's Monitor (Nmon):** é uma ferramenta de monitoramento de desempenho para computadores com sistemas operacionais AIX e *Linux/Unix* (NMON, 2020). O Nmon possui dois modos de funcionamento onde primeiro exibe uma tela com resumos estatísticos das métricas de desempenho do sistema em utilização e o segundo funciona em *background* onde as saídas do processo equivalem às amostras compondo as estatísticas de desempenho armazenadas em arquivos de tabulação. Essas amostras podem ser importadas em um arquivo de *Excel* com macros que transformam os dados em gráficos para auxiliar na análise e na compreensão das métricas vinculados aos recursos do computador hospedeiro que provisionou as instâncias de contêineres e VMs.

d) **Jmeter:** permite medir o desempenho de serviços fornecidas por diversos protocolos da camada de aplicação que utilizam a arquitetura TCP/IP, como aplicações web, banco de dados, e-mail, FTP, ou LDAP (JMETER, 2020). Para a geração das cargas de trabalhos, os clientes Jmeter podem gerar diversos tipos de requisições, simulando *threads* com vários usuários virtuais. O Jmeter disponibiliza também uma variedade de relatórios (*listeners*) estatísticos compostos por tabelas e gráficos bastantes úteis nas análises dos resultados.

e) **Wattsup Power Meter:** consiste em um equipamento medição de potência elétrica (WATTSUP, 2020). Utilizando esse equipamento, é possível coletar amostras da potência elétrica demandada pela infraestrutura experimental e assim verificar o consumo de energia de tal infraestrutura.

Experimentação no ambiente real: nesta etapa, uma arquitetura experimental é construída para que os recursos computacionais sejam alocados nos ambientes baseados em contêineres e VMs. Nesse caso, os recursos alocados têm a finalidade de provisionar ambientes com capacidades que podem variar conforme o andamento de cada cenário experimental. Esses cenários compreendem quantidades iguais de instâncias baseadas em contêineres e VMs que, por sua vez, possuem as mesmas configurações de recursos. Nos

experimentos são utilizados um conjunto de ferramentas de medição que realizam coletas de amostras contendo as métricas selecionadas. Sendo assim, a capacidade de cada cenário deve suportar cargas de trabalhos suficientes para gerar amostras relacionados às métricas selecionadas e, posteriormente, os resultados são submetidos a análises e comparações em outra etapa.

Geração do modelo analítico: nesta etapa, reúnem-se todas as informações necessárias para iniciar o passo a passo de criação do modelo analítico que representará o comportamento do ambiente sob análise. Neste momento, considerando finalizado o entendimento da aplicação, as seleções das tecnologias, as escolhas das métricas e a experimentação no ambiente real, então o modelo será criado. Uma vez criado tal modelo, o mesmo será capaz de simular cenários com configurações não possíveis (ou difíceis) de implementar na arquitetura experimental em virtude de sua limitação de capacidade. Neste trabalho, mais especificamente, adotamos a modelagem baseada em SPN (MARSAN et al., 1994), que permite realizar análises a partir de variações de inúmeros parâmetros, como o tempo de chegada e o número de instâncias do Moodle utilizadas em cada cenário. Ainda nesta etapa, foi escolhida a ferramenta Mercury (SILVA et al., 2015) para modelagem e análise dos cenários, visto que a mesma permite calcular as métricas escolhidas na etapa anterior.

Validação do modelo: um modelo é uma representação completa ou parcial de algo do mundo real. Esta etapa verifica se as abstrações implementadas no modelo estão em conformidade e consistentes com sistema real. Para tal fim, a validação confere uma série de métodos matemáticos e estatísticos para checar se o modelo consegue reproduzir o comportamento do mundo real. Portanto, validar um modelo significa verificar se os pressupostos desenhados apresentam resultados aproximados da realidade. Em suma, a validação assegura que as análises reproduzam valores consistentes em relação aos valores de referência. Para que os valores das métricas sejam estatisticamente equivalentes, o sistema real e o modelo SPN devem ser submetidos sob as mesmas condições de reprodutibilidade de experimentos, considerando, por exemplo, igualdade entre os parâmetros de configurações e as cargas de trabalhos. As médias podem ser validadas através de intervalos de confiança com margem de erro predefinida estatisticamente, bem como um teste denominado de T não pareado (JAIN, 1991), o qual compara estatisticamente as equivalências entre as médias do sistema real e do modelo. Caso o modelo não represente adequadamente o

sistema, outros ajustes deverão ser realizados com novas rodadas de experimentação no ambiente real, até que o modelo seja refinado.

Geração de cenários: caso a validação do modelo seja concretizada com êxito, as análises poderão ser extrapoladas para outros cenários com escopos maiores e diversificados. De acordo com a exatidão do modelo, é possível prever resultados das métricas de desempenho e de consumo de energia sem a necessidade de executar experimentos no sistema real. Dessa forma, serão criados cenários hipotéticos com variações de configurações definidas nos parâmetros do modelo, como números de requisições, tempo de chegada, números de instâncias (réplicas), tempo de serviço e Kilowatt sobre variações de tempos. Assim, tanto as análises quanto as simulações poderão identificar influências de determinados parâmetros nas métricas de desempenho e de consumo de energia. As métricas devem ser avaliadas com o objetivo de encontrar cenários que indiquem melhorias significativas na operação de toda infraestrutura.

Análises dos resultados: nesta etapa, os dados serão analisados e interpretados por ferramentas estatísticas amplamente utilizadas nos meios acadêmicos, como o R (KENETT et al., 2014), o RStudio (R, 2021), o Minitab (GRIMA et al., 2012), entre outras. Os resultados das análises serão demonstrados por gráficos e tabelas que apresentarão as relações mais relevantes entre as métricas. Os cenários baseados em contêineres e VMs terão variações nos parâmetros de configuração para que os resultados sejam objetos de análises de desempenho, de consumo de energia e custo. Os resultados obtidos poderão ser interpretados com mais clareza e precisão, no sentido de minimizar as incertezas nas definições de capacidades dos ambientes reais. Isso poderá assegurar uma maior taxa de acertos nas decisões tomadas pelas partes interessadas, podendo trazer melhorias significativas nas métricas selecionadas. Importante destacar que caso os dados coletados não forem satisfatórios pelos testes realizados na validação, uma ou mais condições de retornos podem ocorrer entre as etapas experimentação no ambiente real, validação do modelo e a geração do modelo analítico, a fim de obter uma precisão esperada entre os dados do sistema real e do modelo. Portanto, pode ser necessário realizar novas medições no sistema real até o modelo representar adequadamente o comportamento do sistema.

4.2 Considerações Finais

Neste capítulo foi apresentada a metodologia utilizada para elaboração desta dissertação. A metodologia propõe uma sequência de 9 etapas elementares na concepção de pesquisas com ênfase experimentos e modelagem analítica. De modo geral, as etapas envolvem tarefas relacionadas a estudos aprofundados de aplicações funcionando em casos concretos, as implantações delas em ambientes replicados por contêineres e VMs, identificação de parâmetros e métricas, realizações de experimentações, construção e validação de modelos, até as análises de cenários através dos resultados obtidos. É importante frisar que esta pesquisa apresenta estudos casos aplicados a duas situações complementares, porém com menor abrangência em relação às etapas realizadas. O primeiro estudo caso não realizou as etapas (6) Geração do modelo analítico e (7) Validação do modelo, visto que o enfoque visava comparar as duas tecnologias sem a necessidade de aplicação e modelagem. Nesse estudo caso, as métricas de desempenho, consumo de consumo e custo foram observados apenas por ferramentas de monitoramento e *benchmarks*. Em seguida, o segundo estudo realizou todas as etapas abordadas neste capítulo, permitindo análises com maior abrangência com o auxílio de modelos SPNs.

5 Arquiteturas Experimentais

Esta seção descreve as duas arquiteturas experimentais adotadas na condução dos experimentos referentes às avaliações de desempenho e consumo energia dos contêineres e VMs, onde a arquitetura 01 foi usada no primeiro estudo de caso, enquanto a arquitetura 02 foi usada para o segundo estudo de caso. O estudo de caso 01 adotou uma arquitetura na qual os recursos foram alocados para construir três cenários, de modo que os ambientes variaram de capacidade conforme as quantidades de contêineres e VMs provisionados. Esse estudo teve o propósito de avaliar e comparar as duas tecnologias através de experimentos realizados por ferramentas e *benchmarks*. Nessa etapa, os ambientes provisionados não tiveram aplicações implantadas nas VMs e nos contêineres, visto que as medições levaram em consideração apenas análises preliminares das tecnologias. Posteriormente, o estudo de caso 02 utilizou uma arquitetura composta pela aplicação AVA Moodle provisionada por contêineres e VMs. Esses ambientes foram observados por experimentos submetidos a cinco cenários que variaram de capacidade entre os contêineres e as VMs. Vale destacar que a arquitetura AVA Moodle foi utilizada para as etapas de modelagem e validação para justificar a adotação dos modelos na geração de cenários para aplicações reais.

5.1 Arquitetura 01 - Ambiente de operação das réplicas baseadas em Contêineres e VMs

Na primeira arquitetura, os experimentos utilizaram um computador hospedeiro que forneceu ambientes baseados em contêineres e VMs. Durante as medições foram coletadas amostras que observaram o comportamento da vazão em Mbit/s, da utilização da CPU e da potência elétrica *Watts* demandada conforme os tamanhos das cargas de trabalhos. O kiloWatt-hora foi a unidade de potência adotada nas exposições dos resultados de consumo de energia e custo, visto que as concessionárias de energia elétrica adotam essa unidade de energia nos cálculos de faturamento pelos serviços fornecidos aos consumidores. A criação dos ambientes baseados em VMs utilizou a ferramenta *hypervisor* KVM, ao passo que os ambientes baseados em contêineres foram criados pelo *Docker Engine*. Sendo assim, tanto as instâncias do *hypervisor* KVM, quanto as instâncias do *Docker* alocaram proporcionalmente os recursos da CPU com 4 núcleos, a memória RAM, a interface *gigabit*

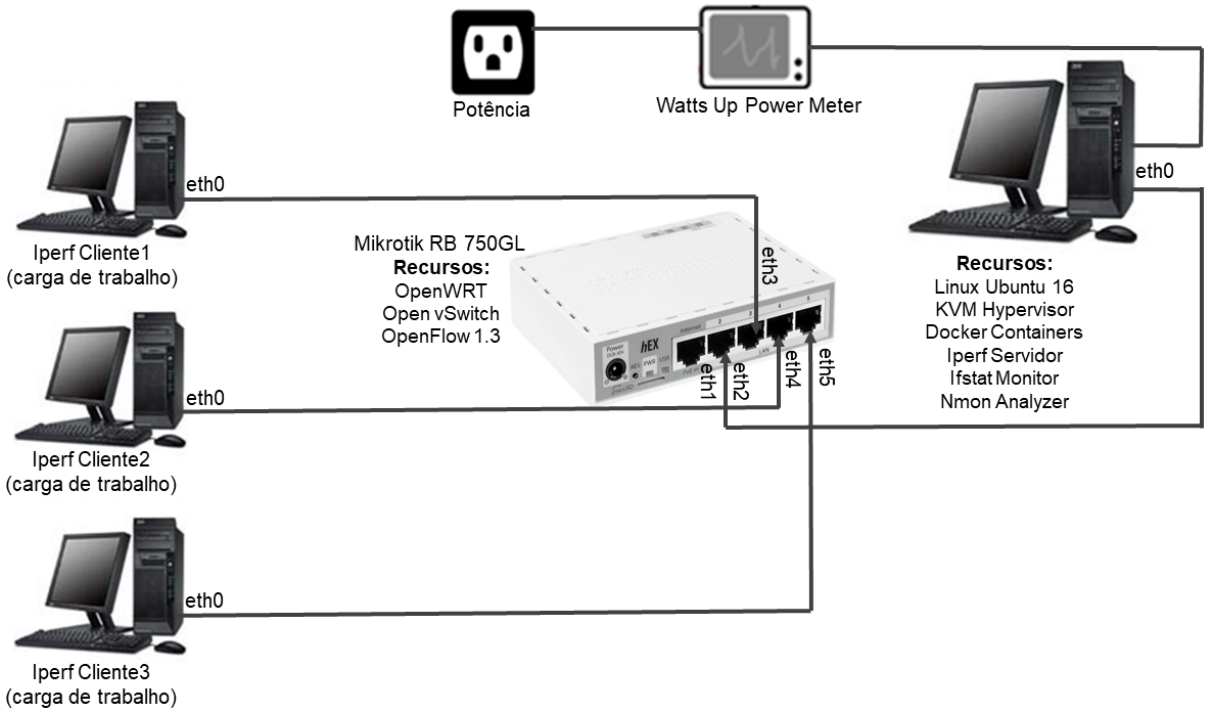

Figura 7 – Primeira arquitetura experimental baseada em contêineres e VMs

ethernet e os demais componentes do computador hospedeiro. O objetivo das medições foi assegurar a imparcialidade entre as coletas das amostras relacionados às duas tecnologias.

A arquitetura experimental está representada pela topologia descrita na Figura 7. Nesta topologia, um roteador *Mikrotik* 750GL estabeleceu a conexão de rede com suporte a 1 Gbit/s de largura de banda entre todos os computadores. Os clientes 1, 2 e 3 estiveram conectados às portas eth3, eth4 e eth5, respectivamente. Já o computador hospedeiro esteve conectado à porta eth2. Os clientes das cargas de trabalhos possuíram as suas interfaces *ethernet* limitadas a uma largura de banda de 100 Mbit/s. Como a interface de rede do computador hospedeiro suportava uma largura de banda de 1 Gbit/s, essa capacidade do *link* foi compartilhada uniformemente para as instâncias de contêineres e VMs ativas. Sendo assim, cada instância teve a interface *ethernet* virtual configurada para transferir 100 Mbit/s. Apesar desta pesquisa não focar no tema *Software defined networking* - SDN (RYU Project Team, 2018), o roteador implementou uma rede local baseada na arquitetura SDN, sendo controlada pelo *Ryu Controller* (RYU Project Team, 2018). A Tabela 2 descreve as configurações dos equipamentos usados nos experimentos. O servidor do ambiente experimental utilizou um computador *Dell Inspiron* 3647 com processador *Intel Core i5* (4 núcleos) de 3.4 Ghz, memória RAM de 8 GB, Disco Rígido de 1 TB e com interface de rede 1 Gbit/s. Os clientes geradores das cargas de trabalhos

foram computadores da marca *Hewlett-Packard* (HP) com processador *Intel Dual Core* de 2.0 Ghz, memória RAM de 2GB, Disco Rígido de 500 GB e placa de rede *Fast Ethernet* de 100 Mbit/s. Os três computadores clientes tiverem uma instalação da distribuição *Linux Ubuntu 16.10*. O computador hospedeiro teve o seu disco rígido (HD) de 1 TB particionado de modo a conter duas instalações da distribuição *Linux Ubuntu 16.10*, sendo uma partição SDA1 de 500 GB destinada à criação das VMs através do *KVM* e outra partição SDA2 de 500 GB destinada à criação dos contêineres através do *Docker*.

Tabela 2 – Especificações dos sistemas

Sistema Computacional	**Configuração**	**SO**	**Hosts**
Hardware Hospedeiro - SDA1	Intel i5, 8GB RAM, 500GB 7200RPM, Ethernet 1000 Mbit/s	Linux Ubuntu 16.10	Servidor-KVM
KVM - Máquina Virtual	Intel i5, 2GB RAM, 100GB 7200RPM, Ethernet 100 Mbit/s	Linux Ubuntu 16.10	VM1, VM2, VM3
Hardware Hospedeiro - SDA2	Intel i5, 8GB RAM, 500GB 7200RPM, Ethernet 1000 Mbit/s	Linux Ubuntu 16.10	Servidor-Docker
Docker - Contêiner	Intel i5, 2GB RAM, 100GB 7200RPM, Ethernet 100 Mbit/s	Linux Ubuntu 16.10	C1, C2, C3
Hardware - Clientes PCs	Intel Dual Core 2.0 Ghz, 2GB RAM, 500GB 5400RPM, Ethernet 100 Mbit/s	Linux Ubuntu 16.10	Cliente 1, Cliente 1, Cliente 3

Tabela 3 – Números de instâncias alocadas para os experimentos

N. de Instâncias do Agregadas	**Níveis de Cenários**	**Cargas de Trabalhos**
C1	Cenário I - Escopo I	1 (Cliente 1)
VM1	Cenário I - Escopo II	1 (Cliente 1)
C1+C2	Cenário II - Escopo I	2 (Cliente 1 e Cliente 2)
VM1+VM2	Cenário II - Escopo II	2 (Cliente 1 e Cliente 2)
C1+C2+C3	Cenário III - Escopo I	3 (Cliente 1, Cliente 2 e Cliente 3)
VM1+VM2+VM3	Cenário III - Escopo II	3 (Cliente 1, Cliente 2 e Cliente 3)

As medições levaram em conta três cenários com quantidades equivalentes de contêineres e VMs. Adicionalmente, os experimentos realizados nos três cenários também observaram a vazão da rede e os volumes de arquivos transferidos conforme a largura de banda suportada por cada escopo. A Tabela 3 descreve os ambientes provisionados por instâncias destinadas à realização dos experimentos. Estes cenários apresentaram três tipos de configurações que visaram estressar os recursos do computador hospedeiro e, posteriormente, resultar em amostras relativas à utilização da CPU, o consumo de energia e a vazão da rede. Quando cada contêiner ou VM foi agregado para executar uma rodada de experimento, o cenário expandia de capacidade de processamento conforme a quantidade de instâncias. Dentro de cada cenário, o Escopo I e o Escopo II representaram rodadas alternadas de experimentos realizados nos ambientes provisionados por cada tecnologia. Basicamente, os cenários estão associados à capacidade do ambiente que cresce conforme

os quantitativos de instâncias provisionadas, ao passo que os escopos I e II estão associados às instâncias baseadas em contêineres e VMs, respectivamente.

Os experimentos realizados nos ambientes provisionados no computador hospedeiro utilizaram *benchmarks* que produziram cargas de trabalhos relativas aos tamanhos de arquivos transferidos. As ferramentas de monitoramento fizeram as coletas das amostras relativas à vazão da rede, a utilização da CPU e o consumo de energia. O *Iperf* foi inicializado no modo *TCP server* dentro das instâncias baseadas em contêineres ou VMs, fornecendo serviços de transferências de arquivos na rede local. As demandas destinadas a esses serviços ocuparam toda largura de banda suportada pelas interfaces *ethernet* de cada cliente que executou o *Iperf* no modo *TCP client* para gerar as cargas de trabalhos. Esse processo garantiu elevadas taxas na vazão da rede, permitindo estressar o processamento geral do computador hospedeiro. É importante frisar que os clientes tiveram as suas interfaces *ethernet* limitadas a 100 Mbit/s de largura de banda. Dessa forma, interface *gigabit ethernet* do computador hospedeiro teve disponibilidade suficiente para atender as requisições oriundas dos clientes. Os experimentos de transferências de arquivos compuseram 60 amostras com intervalos de 30 segundos entre cada observação.

Os clientes das cargas de trabalhos foram utilizados à medida que uma instância de contêiner ou VM era adicionada ao experimento. Os cenários tiveram três agrupamentos de contêineres ou VMs cujas capacidades dos *links* da rede aumentaram as larguras de banda para 100 Mbit/s, 200 Mbit/s e 300 Mbit/s conforme aumentava a capacidade do agrupamento para suportar um maior volume de carga de trabalho. Além disso, os escopos alternavam entre contêineres e VMs dentro de cada cenário. No Cenário I, o Escopo I utilizou apenas um cliente de carga de trabalho (Cliente 1) e uma instância baseada em contêiner (C1) para executar o *Iperf* no modo servidor e as demais ferramentas de medição. Enquanto, o Escopo II utilizou apenas um cliente de trabalho (Cliente 1) e uma instância baseada em VM (VM1). No Cenário II, o Escopo I utilizou dois clientes (Cliente 1 e Cliente 2) para gerar as mesmas cargas de trabalhos em paralelo e duas instâncias baseadas em contêineres (C1+C2) para executar *iperf* no modo servidor. Seguindo raciocínio anterior, o Escopo II adotou o mesmo procedimento utilizando duas instâncias baseadas em VMs (VM1+VM2). Finalmente, no Cenário III, o Escopo I utilizou três clientes que geraram as cargas de trabalhos (Cliente 1, Cliente 2 e Cliente 3) e três instâncias baseadas em contêineres (C1+C2+C3). Enquanto, o Escopo II utilizou três instâncias baseadas em

VMs (VM1+VM2+VM3). Após a conclusão dos experimentos na primeira arquitetura, os passos seguintes focaram na preparação da segunda arquitetura com a finalidade da realização de experimentos na aplicação AVA Moodle provisionadas pelos contêineres e VMs.

5.2 Arquitetura 02 - Ambientes AVA Moodle baseados em Contêineres e VMs

Esta seção descreve todos recursos utilizados na composição da arquitetura experimental da aplicação Moodle provisionada por contêineres e VMs. Esses ambientes foram observados por experimentos submetidos a cinco cenários que variaram de capacidade entre os contêineres e as VMs. Os experimentos forneceram amostras necessárias para dar início às etapas de modelagem e validação que deram consequência a criação dos modelos propostos nesta dissertação. Os ambientes provisionados por VMs utilizaram o *hypervisor* KVM, ao passo que os ambientes provisionados por contêineres foram criados pelo *Docker Engine*.

A Figura 8 apresenta uma visão geral da arquitetura montada para realizar os experimentos. Na arquitetura AVA Moodle, o computador hospedeiro dos contêineres/VMs teve um aumento na capacidade do processamento geral, precisamente no processador e na memória RAM, sendo substituído por um computador *Dell Latitude-5480*, com processador *Intel Core i7* (4 núcleos de 3.9 Ghz), 16 GB de memória RAM, HD de 1 TB e interface de rede com 1 Gbit/s. Esse computador teve o seu HD particionado de modo a conter duas instalações da distribuição *Linux Ubuntu 16.10*, sendo uma partição SDA1 de 500 GB destinada à criação das VMs através do *KVM* e outra partição SDA2 de 500 GB destinada à criação dos contêineres através do *Docker*. Cada instância da aplicação Moodle utilizou o serviço *Apache HTTP Server* (APACHE, 2021) em conjunto com o sistema de gerenciamento de banco de dados *MySQL* (MYSQL, 2021), o qual persistiu os dados em um volume compartilhado no mesmo servidor hospedeiro. Como o *back-end* do Moodle é implementado na linguagem PHP (PHP, 2021), foi instalada a extensão PHP no servidor *Apache* para processar as páginas enviadas nessa linguagem. As instâncias dos contêineres utilizaram uma nova imagem construída a partir da imagem padrão, acrescida da instalação e configuração do Moodle, bem como as dependências

necessárias para execução da aplicação. No caso das VMs, o modelo da VM padrão foi atualizado com a instalação e configuração do Moodle e as dependências requeridas pela aplicação. Os computadores clientes foram configurados com processador *Intel Dual Core* de 2.0 Ghz, 4 GB de memória RAM, HD de 500 GB e interface de rede com 1 Gbit/s. Neste estudo de caso, os clientes tiveram instalações do *Jmeter* para gerar as cargas de trabalhos através das requisições *http*, a fim de demandar processamento das instâncias do Moodle provisionadas no computador hospedeiro.

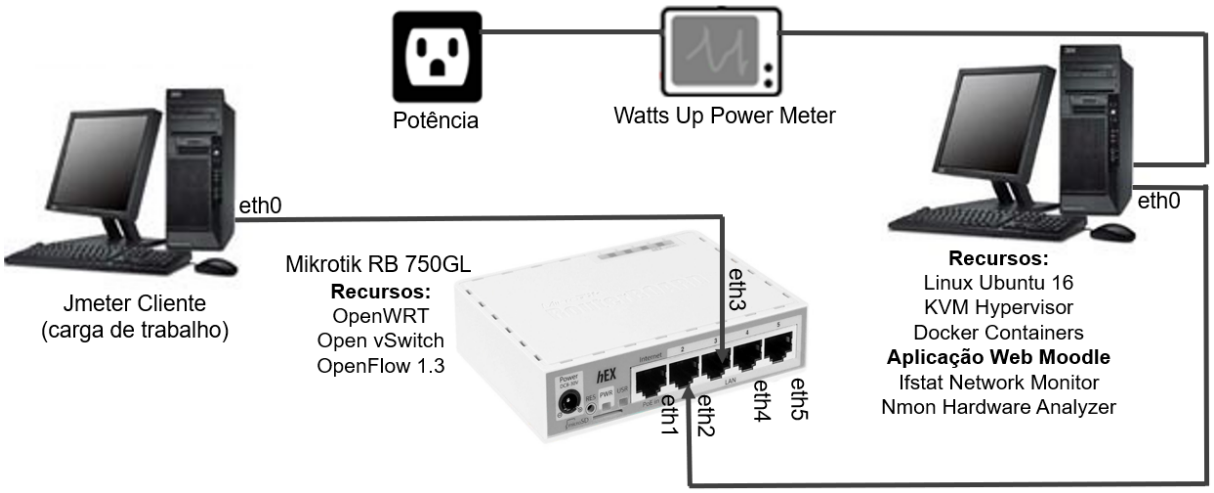

Figura 8 – Segunda arquitetura experimental AVA Moodle provisionada por contêineres e VMs

A topologia foi configurada com um roteador *Mikrotik* 750GL foi utilizado para estabelecer as conexões de rede com suporte a 1 Gbit/s de largura de banda para todas as portas. O computador cliente *Jmeter* fez uma conexão entre sua interface de rede de 1 Gbit/s com a porta *eth3* do roteador, estabelecendo um *link* de 1 Gbit/s. Da mesma forma, o computador hospedeiro fez uma conexão entre sua interface de rede de 1 Gbit/s com a porta *eth2* do roteador, estabelecendo outro *link* de 1 Gbit/s. As instâncias de contêiner ou VM tiveram as suas interfaces *Ethernet* virtuais configuradas com taxa de transferência correspondente a 100 Mbit/s. Dessa forma, não houve risco de ocorrer gargalos na interface de rede computador hospedeiro em virtude do *link* ter disponível 1 Gbit/s de largura de banda para suportar as demandas dos clientes. As ferramentas de monitoramento realizaram as coletas das amostras no computador hospedeiro, durante o período que o *Jmeter* gerou as requisições destinadas às instâncias do Moodle em operação. Ao mesmo tempo, o medidor *Watts UP* (WATTSUP, 2020) fez as coletas das amostras de potências elétricas demandadas pelo computador hospedeiro.

Os experimentos foram executados através de requisições http que representaram 13 interações frequentemente utilizadas pelos alunos do Moodle. Isso significou que um aluno representou um conjunto de 13 requisições enviadas para as instâncias do Moodle. As cargas de trabalhos foram definidas conforme cinco turmas com diferentes quantidades de alunos que demandaram recursos dos computador hospedeiro. Para cada turma, os experimentos forneceram 60 amostras da vazão de requisições por segundo e a potência elétrica demandada. O kiloWatt-hora foi a unidade de potência adotada nas exposições dos resultados de consumo de energia e custo, visto que as concessionárias de energia elétrica adotam essa unidade de energia nos cálculos de faturamento pelos serviços fornecidos aos consumidores.

5.3 Considerações Finais

Este capítulo abordou detalhes das configurações aplicadas nas duas arquiteturas experimentais adotadas nesta dissertação. A primeira arquitetura teve propósito de avaliar cenários com ambientes provisionados por contêineres e VMs, sem a implantação de aplicações. Nessa arquitetura, foram criados três cenários experimentais com diferentes níveis configurações entre os contêineres e as VMs. Os experimentos foram realizados apenas com a utilização de ferramentas de monitoramento e *benchmarks*. As amostras geradas por cada experimento serão analisadas e apresentadas com auxílio de ferramentas e técnicas estatísticas. A segunda arquitetura teve o propósito de estender os experimentos a novos cenários bases para a geração dos modelos analíticos. Nessa arquitetura, foram criados cinco cenários experimentais com diferentes configurações da aplicação Moodle implantadas em contêineres e VMs.

6 Modelo Analítico

Este capítulo apresenta os modelos concebidos para a Arquitetura 02 (Figura 8). Os dois modelos foram criados para representar alguns comportamentos da aplicação AVA Moodle. Sendo um modelo focado em ambientes provisionados por contêineres e o outro modelo focado em ambientes provisionados por VMs. Vale destacar que tais modelos podem ser usados para representar cenários que são difíceis (ou impossíveis) de serem analisados em infra estruturas reais.

6.1 Modelos para a Arquitetura 02

Os dois modelos SPNs permitem calcular a vazão, o tempo de resposta e o consumo de energia demandado por instâncias baseadas em contêineres e VMs. Assim, os modelos podem facilitar os ajustes de parâmetros de entrada para encontrar as configurações mais adequadas de acordo com a carga de trabalho da aplicação. Dessa forma, esses modelos podem estimar os efeitos de eventuais aumentos ou reduções da capacidade da operação no sistema real. Os cenários foram subdivididos em escopos de contêineres e VMs cujos ambientes possuem configurações semelhantes para que os resultados sejam analisados nas mesmas condições. Apesar das duas tecnologias apresentarem tempos computacionais distintos em relação ao desempenho e o consumo de energia, os comportamentos dos eventos de ambos modelos são similares. Por isso, para evitar redundâncias nas exposições sobre os comportamentos dos modelos, as abordagens foram dadas apenas aos eventos relacionados ao modelo contêiner, no entanto, as explicações também são extensíveis ao modelo VM. À vista disso, daqui por diante, os rótulos iniciados com as letras CT (Figura 9 (a)) descrevem os elementos que fazem referência ao modelo contêiner, enquanto os rótulos inciados com as letras VM (Figura 9 (b)) descrevem os elementos relacionados ao modelo VM.

Nesta dissertação, uma carga de trabalho corresponde a um aluno acessando 13 requisições http pertinentes às funcionalidades bem recorrentes do *Moodle*, como acessar o frontend da aplicação, efetuar logon, visualizar um curso, visualizar páginas de atividades, submeter um questionário de atividades, entrar no fórum de discussões, responder a uma discussão, enviar um arquivo de texto, efetuar

logout, entre outras. Cada requisição representa um *token* armazenado na constante `CT-Número_de_Requisição`, a qual representa a quantidade de alunos realizando acessos simultâneos aos recursos do *Moodle*. Consequentemente, o lugar `CT-Clientes` que faz referência aos *tokens* definidos na constante `CT-Número_de_Requisição` para que eles sejam disparados pela transição exponencial `CT-T_Chegada_Fila` cujo *delay* é definido pela constante `CT-Tempo_de_Chegada`. O disparo da transição `CT-T_Chegada_Fila` também está condicionado à presença de *tokens* no lugar `CT-Fila` cujo tamanho é definido pela constante `CT-Tamanho_da_Fila`. Dessa forma, quando ocorre o disparo da transição `CT-T_Chegada_Fila`, os *tokens* são consumidos ao mesmo tempo pelos lugares `CT-Clientes` e `CT-Fila` e, consequentemente, um *token* é gerado no lugar `CT-Processamento_Fila`. Os *tokens* presentes no lugar `CT-Fila` representam a capacidade disponível do servidor (contêiner ou VM) para enfileirar as requisições. Os *tokens* do lugar `CT-Fila` são decrementados à medida que requisições de entrada são processadas e enfileiradas no lugar `CT-Processamento_Fila`, o qual habilita o disparo da transição imediata `CT-T_Saída_Fila` que representa o envio das requisições dos usuários para dentro da fila do servidor responsável por processar as requisições dos usuários. A presença de *tokens* nos lugares `CT-Processamento_Fila` e `Docker` (ou `KVM`) habilitam a transição imediata `CT-T_Saída_Fila`. Quando `CT-T_Saída_Fila` dispara, um *token* é consumido do lugar `CT-Processamento_Fila` e um *token* é consumido do lugar `Docker`. Ao mesmo tempo, um *token* é gerado no lugar `CT-Fila`, devolvendo a capacidade da fila atender novas requisições, um *token* é gerado no lugar `CT-Req_Processamento`, indicando que uma ou mais instâncias (contêineres ou VMs) estão processando as requisições. O número de *tokens* presentes no lugar `Docker` define a quantidade de instâncias que são hospedadas no servidor hospedeiro. O lugar `Docker` faz referência à constante `CT-Número_de_Instâncias`, a qual representa a capacidade do sistema suportar a carga de requisições conforme limites estabelecidos no mecanismo de auto-escalonamento do *cluster*. Quando o lugar `CT-Req_Processamento` termina de processar uma ou mais requisições, o sistema devolve as instâncias novamente para o cluster `Docker`. Esse processo é representado pelo disparo da transição exponencial `CT-T_Serviço`, a qual faz referência à constante `CT-Tempo_de_Serviço` que corresponde ao tempo necessário para que uma instância (um *token*) possa processar uma ou mais requisições conforme as cargas de trabalhos geradas pelas turmas.

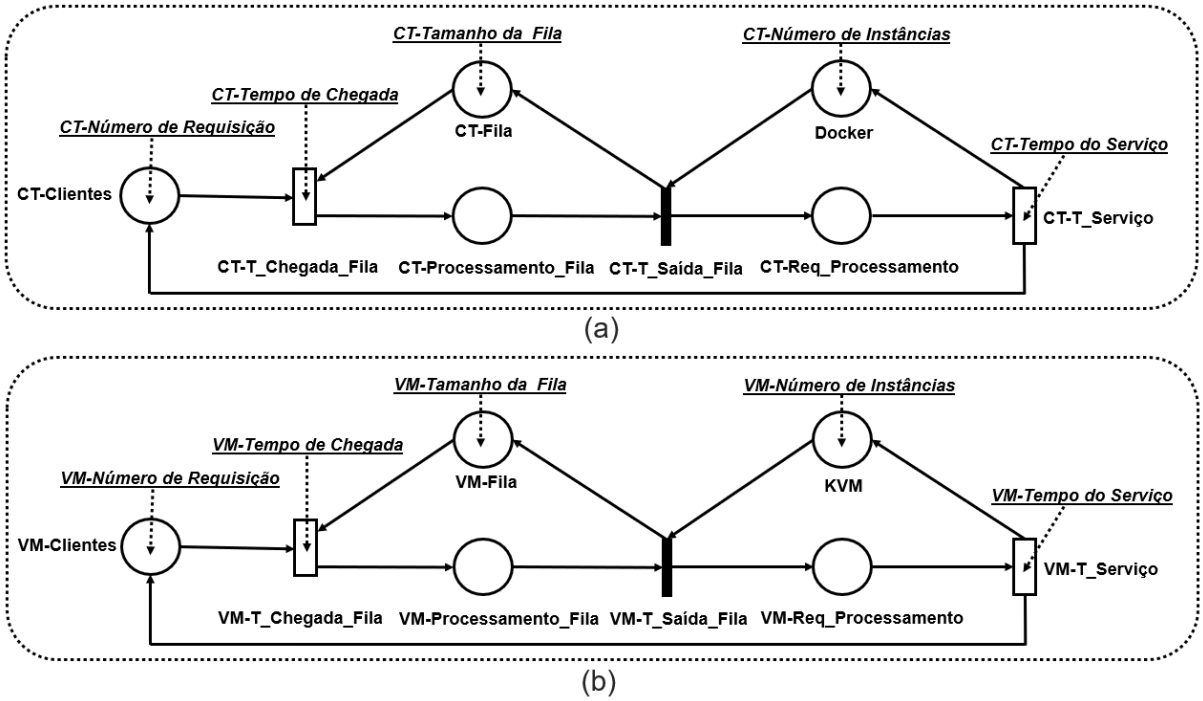

Figura 9 – Modelos representando os conjuntos de requisições do Moodle para ambientes baseados em contêineres (a) e VMs (b).

O tempo médio para processar uma requisição de um usuário é utilizado como *delay* da transição exponencial `CT-T_Serviço` que representa a conclusão do processamento das requisições dos usuários pelas instâncias do *Moodle* (Contêineres ou VMs). O tempo de serviço corresponde ao atraso gerado para uma instância processar uma carga de requisições simultâneas com determinado tempo de chegada entre elas. Por exemplo, considerando que apenas uma instância de contêiner receba uma carga de trabalho correspondente a 130 requisições por segundos, simultaneamente, onde cada requisição possui um tempo de chegada equivalente a 100 ms. O processamento dessa operação produziu uma vazão equivalente a 1.95690 req/s. A partir da vazão gerada pela instância pode ser calculado o seu tempo de serviço para processar uma requisição individual em unidades de segundos. Então, o tempo de serviço é calculado entre a razão de uma unidade de segundo e a vazão de requisições por segundos, ou seja, 1/1.95690 req/s que resulta em 0.51101 segundos de processamento para que uma instância de contêiner atenda cada requisição individual dentre as 130 requisições. O tempo de serviço está associado diretamente à capacidade de processamento do sistema responder uma certa carga de trabalho. O poder computacional dos nós têm influência direta no tempo de serviço de um determinado *cluster* associado a uma aplicação. Supondo que os nós disponham de recursos suficientes, cada instância (`Docker ou KVM`) atribuída ao *cluster* tenderá a reduzir o tempo de serviço da

aplicação e aumentar a taxa da vazão. Por outro lado, o tempo de chegada das requisições influencia diretamente na taxa da vazão e no tempo de resposta de todo o sistema. É importante destacar que o tempo de chegada é um parâmetro atribuído à constante `CT-Tempo_de_Chegada`, o qual serve de referência para a transição `CT-T_Chegada_Fila`.

Tabela 4 – Atributos das transições utilizadas nos modelos SPNs contêineres e VMs.

Transição	**Tipo**	**Semântica**	**Peso**	**Prioridade**
CT-T_Chegada_Fila	Temporizada	Single Server	-	-
CT-T_Saída_Fila	Imediata	-	1	1
CT-T_Serviço	Temporizada	Infinite Server	-	-
VM-T_Chegada_Fila	Temporizada	Single Server	-	-
VM-T_Saída_Fila	Imediata	-	1	1
VM-T_Serviço	Temporizada	Infinite Server	-	-

Tabela 5 – Expressões para calcular as métricas nos modelos SPNs contêineres e VMs.

Métricas	**Expressões**
Vazão-CT	((E{#CT-Req_Processamento}) × (1/CT-Tempo_do_Serviço))
Tempo_de_Resposta-CT	((E{#CT-Processamento_Fila})+(E{#CT-Req_Processamento})) /((E{#CT-Req_Processamento})/CT-Tempo_do_Serviço)
kiloWatt_por_Tempo-CT	PB+(((PD/1000) × (Δt)) × (E{#CT-Req_Processamento}))
Vazão-VM	((E{#VM-Req_Processamento}) × (1/VM-Tempo_do_Serviço))
Tempo_de_Resposta-VM	((E{#VM-Processamento_Fila})+(E{#VM-Req_Processamento})) /((E{#VM-Req_Processamento})/VM-Tempo_do_Serviço)
kiloWatt_por_Tempo-VM	PB+(((PD/1000) × (Δt)) × (E{#VM-Req_Processamento}))

Tabela 6 – Descrição dos parâmetros utilizados nos modelos SPNs contêineres e VMs.

Variável	**Parâmetro**
CT-Número_de_Requisição	Quantidades de requisições por alunos ou turmas
CT-Tempo_de_Chegada	Tempo de chegada da requisição
CT-Número_de_Instâncias	Quantidades de instâncias disponíveis
CT-Tempo_de_Serviço	Tempo de serviço da aplicação
CT-kilowatt	Potência elétrica convertida para KW (PB+PD)/1000
VM-Número_de_Requisição	Quantidades de requisições por alunos ou turmas
VM-Tempo_de_Chegada	Tempo de chegada da requisição
VM-Número_de_Instâncias	Quantidades de instâncias disponíveis
VM-Tempo_de_Serviço	Tempo de serviço da aplicação
VM-kilowatt	Potência elétrica convertida para KW (PB+PD)/1000

A Tabela 4 apresenta as transições utilizadas nos modelos contêiner e VM. O tempo

de chegada entre as requisições do *Jmeter* assumiu uma distribuição exponencial, assim como os *tokens* gerados pela transição `CT-T_Chegada_Fila` ou `VM-T_Chegada_Fila`. As transições temporizadas `CT-T_Chegada_Fila` ou `VM-T_Chegada_Fila` foram definidas com políticas de disparos *Single Server*, enquanto as transições temporizadas `CT-T_Serviço` ou `VM-T_Serviço` foram definidas com políticas de disparos *Infinite Server*. Na semântica *Infinite Server*, todos os *tokens* são disparados pela transição `CT-T_Serviço` na mesma janela de tempo, indicando um comportamento de processamento paralelo das requisições, ao passo que, na semântica *Single Server*, os *tokens* são o disparados sequencialmente em janela de tempo não compartilhada. Esse comportamento indica que as instâncias não operam paralelamente. As transições imediatas `CT-T_Saída_Fila` ou `VM-T_Saída_Fila` tiveram os seus pesos e prioridades com valores 1. A Tabela 5 apresenta as expressões utilizadas na ferramenta *Mercury* (SILVA et al., 2015) para calcular as métricas de vazão, tempo de resposta, consumo de energia, enquanto a Tabela 6 apresenta os parâmetros utilizados nos modelo SPNs propostos. As expressões citadas na Tabela 5 são baseadas nas fórmulas da Lei de Little (LITTLE, 1961).

A `Vazão` é dada pelo produto do número de *tokens* esperados no lugar `CT-Req_Processamento` multiplicado pela taxa atrelada ao processamento da requisição. A métrica `CT-Tempo_de_Resposta` é obtida pelo resultado do somatório dos *tokens* esperados nos lugares `CT-Processamento_Fila` e `CT-Req_Processamento`, dividindo pelo resultado da expressão com *tokens* esperados no lugar `CT-Req_Processamento` dividido pelo tempo de serviço atribuído ao parâmetro `CT-Tempo_do_Serviço`. A métrica `kiloWatt_por_Tempo_CT` representa o consumo de energia calculado através do produto entre o kiloWatt atribuído ao parâmetro `CT-kilowatt` e as variações de tempos definidas na variável $\triangle t$, multiplicando pelo resultado da expressão com *tokens* esperados no lugar `CT-Req_Processamento`. Cada *token* armazenado no lugar `CT-Req_Processamento` representa um aumento na demanda de energia elétrica em decorrência das instâncias agregadas no processamento da demanda.

A energia consumida foi calculada de acordo com a equação 6.1, onde a variável *PD* representa a potência (W) demandada pelas instâncias em processamento. O consumo de energia fixo associado à infraestrutura em período de inatividade foi referenciado pela variável PB para indicar a potência base. A constante 1000 é empregada na conversão da unidade de potência *Watt* para a unidade *kilowatt* (kW). A variável $\triangle t$ representa as

variações de tempos com períodos relacionado às horas, dias, meses e anos. Os mesmos raciocínios aplicam-se aos cálculos das métricas iniciadas com as letras VM. A soma das variáveis PB e PD representa a potência total (PT) da operação, a qual corresponde à energia fixa consumida pela infraestrutura vinculada a um agrupamento de instâncias sendo demandadas por cargas de trabalhos. A constante 1000 é empregada na conversão da unidade de potência *Watt* para a unidade *kilowatt* (kW). A variável $\triangle t$ representa as variações de tempos com períodos relacionado às horas, dias, meses e anos.

$$E(kWh) = \frac{PB + PD}{1000} \times \triangle t \tag{6.1}$$

6.2 Considerações Finais

Este capítulo apresentou os modelos SPNs adotados neste trabalho. No decorrer do capítulo foram abordados com detalhes o passo a passo de todos elementos gráficos, expressões lógicas e parâmetros de configurações utilizados para calcular as métricas oriundas da arquitetura AVA Moodle. Primeiramente, foi explicado o cenário adotado. Em seguida, foram explicadas as movimentações dos *tokens* entre os lugares à medida que as transições são disparadas. Por fim, foram explicados os parâmetros de configuração e as expressões adotadas nos cálculos das fórmulas das métricas dos modelos SPNs.

7 Resultados e Discussão

Este capítulo apresenta e discute os resultados obtidos neste trabalho, onde os mesmos foram divididos em dois estudos casos compostos cenários experimentais complementares. O Estudo de Caso I avalia e compara o desempenho e o consumo de energia dos ambientes provisionados por contêineres e VMs através de *benchmarks*. Nesse estudo caso, foram utilizados os recursos alocados na primeira arquitetura conforme mostra a Figura 7. As análises foram realizadas considerando os tamanhos de arquivos transferidos em MBytes, a vazão da rede com taxa na ordem de Megabit por segundo (Mbit/s), a utilização de CPU conforme variáveis de sistema, a potência elétrica demandada e o custo associado ao consumo de energia. O Estudo de Caso II utilizou os modelos SPNs para encontrar os cenários com os melhores parâmetros de configurações de capacidade adequados às demandas dos ambientes AVA Moodle implantados em contêineres e VMs. Para esse estudo de caso, foi utilizado a arquitetura apresentada na Figura 8.

7.1 Estudo de Caso I: Avaliação e Comparação de Contêineres e VMs

Nesta seção serão apresentadas as análises e as comparações das métricas referente aos cenários e escopos criados a partir da primeira arquitetura. A composição das amostras se basearam nos valores médios: dos tamanhos dos arquivos transferidos em unidades de Megabytes; da vazão da rede observadas em Megabit por segundo (Mbit/s); da utilização da CPU resultante da demanda pelo serviço; e da potência demandada pela operação. Os arquivos transferidos pelo *iperf* aumentaram conforme a quantidade de instâncias baseadas em contêineres ou VMs que estiveram com os serviços aptos a responder as requisições dos clientes. É importante destacar que o *iperf* estabeleceu as conexões por meio do protocolo TCP, o qual transportou os arquivos provenientes das cargas de trabalhos. Cada escopo selecionado entre os cenários compreenderam experimentos cujos períodos de observações duraram 1800 segundos (30 minutos) divididos em 60 amostras de 30 segundos.

7.1.1 Cargas de Trabalhos

As intensidades das cargas de trabalhos foram caracterizadas pelos volumes de arquivos transferidos através das interfaces *ethernet* configuradas para suportar 100 Mbit/s de largura de banda. Sendo Assim, no Cenário I, duas medições foram realizadas, sendo uma destinada à instância baseada em contêiner e outra à instância baseada em VM, que resultaram em arquivos cujos tamanhos médios das 60 amostras corresponderam a 336,97 MBytes e 337,00 MBytes, respectivamente. Em seguida, o Cenário II avaliou as duas tecnologias através de dois grupos distintos cujas composições se caracterizaram por duas instâncias de contêineres e duas de VMs, sendo ambas capazes de processar as demandas relativas às duas cargas de trabalhos em paralelo, as quais resultaram em arquivos cujos tamanhos médios corresponderam a 672,37 MBytes e 673,75 MBytes, respectivamente. Por fim, o Cenário III adotou dois experimentos relativos às três cargas de trabalhos em paralelo que demandaram a capacidade de processamento de dois agrupamentos compostos por três instâncias de contêineres e três de VMs que produziram amostras caracterizadas por arquivos de 1004,85 MBytes e 1010,87 MBytes, respectivamente.

Tabela 7 – Tamanhos dos arquivos transferidos por cargas de trabalhos entre os cenários

MBytes	**Cenario I**		**Cenario II**		**Cenario III**	
Estatísticas	Escopo I	Escopo II	Escopo I	Escopo II	Escopo I	Escopo II
Médias (50 amostras)	337.00	336.97	673.75	672.37	1010.87	1004.85
Desvio Padrão	0.00	0.181	0.75	0.801	0.343	7.412
Intervalo de Confiança (95%)	[337.00;337.00]	[336.92;337.01]	[673.56;673.94]	[672.16;672.57]	[1010.778;1010.955]	[1002.93;1006.76]

7.1.2 Vazão

De acordo com a Tabela 7, considerando que os experimentos mantiveram as equivalências numéricas entre as instâncias do Escopo I e Escopo II em relação ao Cenário I, Cenário II e Cenário III, observou-se que as médias das amostras referente aos tamanhos dos arquivos apresentaram diferenças numericamente pequenas. Por isso, devido às proximidades dos tamanhos de arquivos, pode-se afirmar que os contêineres e as VMs conseguiram processar as requisições dos clientes com desempenhos aproximados. De forma semelhante, as amostras da vazão na rede produziram taxas de transferências médias que apresentaram comportamentos semelhantes aos valores das médias relativos aos tamanhos

dos arquivos resultantes das cargas de trabalhos.

Sendo assim, os volumes de MBytes transportados pelas conexões TCP estabelecidas entre os clientes e as instâncias dos servidores utilizaram as interfaces *ethernet* com capacidade para suportar 100Mbit/s de largura de banda nominal. Logo, devido à uniformidade na capacidade de transmissão das interfaces *ethernet* situadas nos contêineres e VMs, as vazões atingiram valores médios com leves diferenças entre as instâncias dos Escopo I e Escopo II para todos os cenários. Considerando que os *links* aumentaram a largura de banda para 100 Mbit/s, 200 Mbit/s e 300 Mbit/s conforme as quantidades de instâncias agregadas nos Cenário I, Cenário II e Cenário III, respectivamente. Portanto, é importante ressaltar que as medições observaram amostras cujas vazões atingiram taxas de transferências próximas às larguras de bandas disponibilizadas nos *links* estabelecidos entre os clientes e contêineres ou VMs instanciadas no computador hospedeiro. A Figura 10 representa graficamente as comparações entre as taxas das vazões médias atingidas pelas instâncias baseadas em contêineres e VMs. As vazões atingiram altas taxas de transferências e elevados níveis de utilização da rede conforme a capacidade dos *links*.

Ao comparar as médias entre o Escopo I e Escopo II referente ao Cenário I, observou-se que um experimento relativo a um contêiner atingiu uma taxa de 94,10 Mbit/s, enquanto que uma VM atingiu uma taxa de 96,13 Mbit/s. Em seguida, o Cenário II observou que as operações paralelas relativas às duas instâncias baseadas em contêineres atingiram uma taxa média de 189,21 Mbit/s. Ao passo que, as operações paralelas relativas às duas instâncias baseadas em VMs atingiram uma taxa média de 192,18 Mbit/s. De forma semelhante, o Cenário III observou que as operações paralelas referentes às três instâncias baseadas em contêineres atingiram uma taxa média de 284,60 Mbit/s. Já as operações paralelas referentes às três instâncias baseadas em VMs atingiram uma taxa média de 286,42 Mbit/s.

As vazões obtidas pelas instâncias agrupadas em todos os cenários experimentais apresentaram diferenças pequenas entre as médias. As diferenças entre as taxas médias das vazões foram de 2,03Mbit/s, 2,97Mbit/s e 1,82Mbit/s. Em termos percentuais, considerando as vazões atingidas pelos experimentos realizados nos escopos II dos cenários I, II e III, as VMs superaram os contêineres em 2,16%, 1,57% e 0,64%, respectivamente. Seguindo o mesmo raciocínio, todavia, a favor dos contêineres, as diferenças percentuais entre as médias dos arquivos transferidos foram 0,01%, 0,21% e 0,6%, respectivamente.

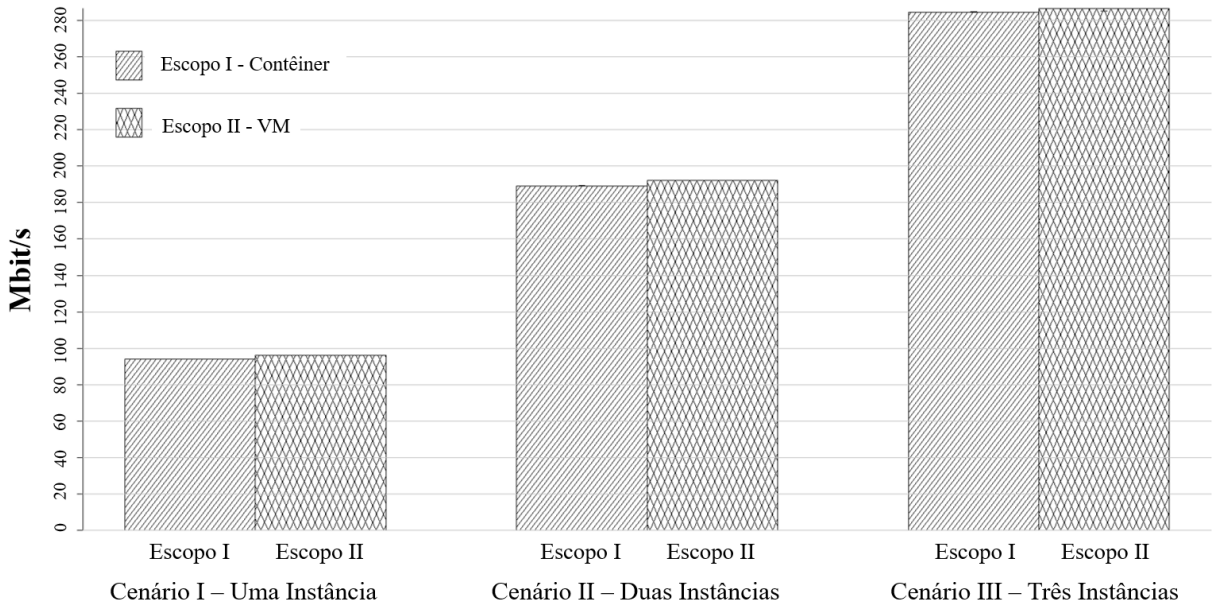

Figura 10 – As taxas de transferências relativas às vazões produzidas por *links* de 100 Mbit/s, 200 Mbit/s e 300 Mbit/s de largura de banda

A fim de verificar se as diferenças são estatisticamente significante, testes estatísticos foram realizados. Primeiramente, foi verificado a normalidade das amostras, de modo que o teste de *Shapiro-Wilk* (SHAPIRO; WILK, 1965) avaliou a conformidade das médias com uma distribuição normal com um nível de significância de 0,05 (5%). Os valores das probabilidades de significâncias (valor-p) foram inferiores ao nível de significância de 0,05. Sendo assim, as médias não seguem uma distribuição normal. Uma vez que as amostras não seguem uma distribuição normal, o teste não-paramétrico chamado de *Mann-Whitney* (MANN; WHITNEY, 1947) foi adotado para comparar os escopos com um nível de significância de 0,05 (5%). Os resultados dos testes aplicados em todas as amostras indicaram uma probabilidade de significância (valor-p) inferior ao nível de significância de 0,05. Dessa forma, o resultado do teste revela que existem diferenças estatisticamente significante entre os escopos comparados. Todavia, no ponto de vista dos desempenhos dos contêineres e VMs, as evidências empíricas apontaram que as diferenças entre as médias das taxas apresentaram valores mínimos, visto que, a favor dos contêineres, as diferenças entres as médias dos arquivos transferidos foram 0,03MBytes, 1,38MBytes e 6,02MBytes, respectivamente.

Os resultados dos experimentos demonstraram que as taxas médias das vazões e os tamanhos médios dos arquivos transferidos na rede apresentaram crescimentos proporcionais aos números de clientes nas cargas de trabalhos e os quantitativos de

instâncias baseadas em contêineres e VMs utilizadas nos Cenário I, cenario II e Cenário III. Nesse caso, as taxas médias das vazões dos *links* aumentaram nas proporções de 94,10Mbit/s e 96,13Mbit/s em virtude da inclusão de uma instância baseada em contêiner ou uma baseada em VM, respectivamente. Além disso, os volumes de arquivos transferidos com base na vazão aumentaram na proporção média de 337,00 MBytes para cada cliente, contêiner ou VM inseridos nos experimentos.

7.1.3 Utilização da CPU

Basicamente, as proporções dos volumes de MBytes e largura de banda dos *links* inerente aos experimentos produziram efeitos práticos no desempenho geral do computador hospedeiro. As medições realizadas através da ferramenta *NMON Analyzer* coletou amostras oriundas das cargas de trabalhos tratadas pela CPU que foi compartilhada entre todas as instâncias baseadas em contêineres e VMs. As análises das amostras levaram em consideração os valores médios das variáveis de ambientes responsáveis pelo monitoramento do desempenho do hardware da CPU. Os níveis de utilização e ociosidade da CPU foram observados através de quatro variáveis de ambiente representadas por valores percentuais. Os resultados representados na Figura 11 (a) mostraram que os contêineres dos escopos I apresentaram percentuais de utilização da CPU consideravelmente inferiores quando comparados às VMs dos escopos II.

Tabela 8 – Percentual de ociosidade (*idleness*) e utilização da CPU entre os cenários

Utilização	**User%**			**Sys%**			**Idle%**			**Busy%**		
QTD de instâncias	**Média**	**SD**	**IC**	**Média**	**SD**	**IC**	**Média**	**SD**	**IC**	**Média**	**SD**	**IC**
Cenário I - Escopo I	1.23	0.19	[1.19;1.28]	4.48	0.25	[4.41;4.54]	94.16	0.32	[94.07;94.24]	5.71	0.32	[5.63;5.80]
Cenário I - Escopo II	6.09	0.42	[5.99;6.20]	23.55	0.40	[23.45;23.65]	70.02	0.49	[69.90;70.14]	29.64	0.49	[29.51;29.77]
Cenário II - Escopo I	1.56	0.57	[1.41;1.70]	7.09	0.43	[7.0;7.2]	91.21	0.72	[91.03;91.40]	8.64	0.74	[8.45;8.83]
Cenário II - Escopo II	13.22	0.44	[13.12;13.34]	47.53	2.40	[46.91;48.15]	38.98	2.50	[38.34;39.62]	60.75	2.50	[60.11;61.40]
Cenário III - Escopo I	2.27	1.02	[2.00;2.53]	8.5	0.46	[8.38;8.62]	89.10	1.06	[88.83;89.34]	10.77	1.06	[10.50;11.04]
Cenário III - Escopo II	22.23	1.49	[21.84;22.61]	65.07	1.56	[64.66;65.47]	12.58	0.78	[12.38;12.78]	87.3	0.77	[87.1;87.5]

A Tabela 8 apresenta os resultados estatísticos que consideraram as médias aritméticas, os desvios padrão (SD) e os intervalos de confiança (IC) de 95% das variáveis de ambiente a seguir: *User*, *Sys*, *Idle* (ociosidade) e *Busy* (ocupação). A varável *User* é definida pelo percentual de utilização da CPU em conformidade com as sessões abertas pelos usuários. Nesse caso, apenas um usuário esteve logado no SO durante as medições. A variável *Sys* representa o percentual de utilização da CPU por aplicações e serviços do SO

ou *Kernel*. A variável *Idle* representa o percentual de ociosidade da CPU. Por fim, a varíavel *Busy* significa o percentual total de utilização da CPU. As medições demonstraram que tanto os contêineres quanto as VMs sem processar as cargas de trabalhos não representaram impacto na utilização CPU, independentemente da quantidade de instâncias utilizadas nos três cenários experimentais. Em virtude da ausência de cargas de trabalhos destinadas aos servidores instanciados pelos contêineres e VMs, a ociosidade da CPU permaneceu próxima de 99,8%. É importante destacar que o comportamento da variável *Idle* é inversamente proporcional ao comportamento da variável *Busy* cujo resultado pode ser obtido pelo somatório das variáveis *User* e *Sys*. A análise de desempenho deste trabalho focou nas variáveis *Busy* e *Idle*, visto que ambas representam quase a totalidade do processamento no ambiente *Linux/Unix*.

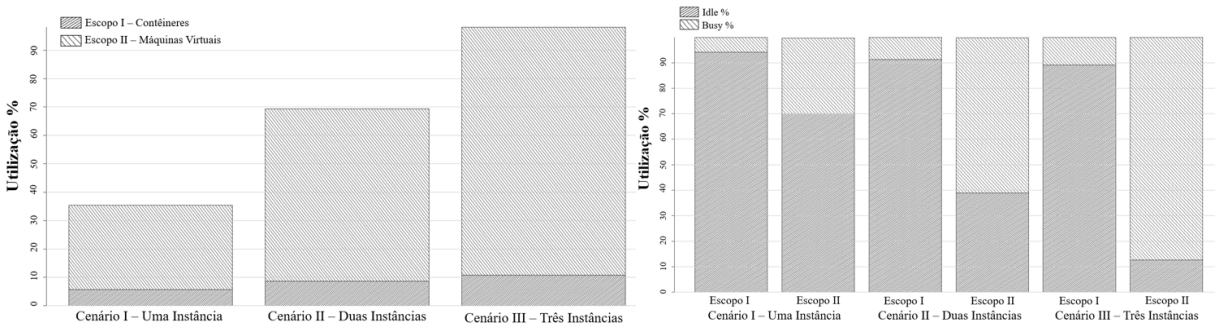

Figura 11 – Comparação da utilização da CPU entre os cenários baseados em Contêineres e VMs: (a) Percentual de utilização da CPU dos escopos I e II - (b) Percentual de utilização das variáveis Idle e Busy

De acordo com a Tabela 8, a coluna média da variável *sys* representou uma maior carga na utilização da CPU em virtude do *Hypervisor* ter executado as chamadas de sistema direcionadas aos processos que compõem o *kernel* do sistema hospedeiro. O *Hypervisor* intermedeia o processo de conversão das instruções entre o sistema convidado e o sistema hospedeiro. Por outro lado, a coluna média da variável *user* representou uma menor carga na utilização CPU, porque os seus processos são executados no espaço do usuário. Dessa forma, em razão das cargas de trabalhos e dos agrupamentos de VMs do escopo II, o *Hypervisor KVM* e os SOs convidados representaram uma sobrecarga adicional no percentual de utilização da variável *sys*.

A Figura 11 (a) mostra as sobrecargas geradas pelas instâncias baseadas em contêineres e VMs em relação ao percentual de utilização da CPU nos três cenários observados. Enquanto a Figura 11 (b) mostra as sobrecargas geradas pelas duas tecnologias

em relação ao percentual de utilização das variáveis *Busy* e *Idle* para os três cenários observados. As cargas de trabalhos geradas pelo *iperf* observou-se que uma instância baseada em contêiner (C1) e uma baseada em VM (VM1) representaram 5,71% e 29,64% de utilização *(busy)* média da CPU, respectivamente. Ao passo que a ociosidade (*Idle*) da CPU correspondeu a 94,16% para uma instância baseada em contêiner e 70,02% para uma baseada em VM. Em seguida, o Cenário II compreendeu dois escopos experimentais caracterizados pelos paralelismos de duas instâncias baseadas em contêineres e duas baseadas em VMs. Os experimentos submetidos aos dois contêineres (C1+C2) e as VMs (VM1+VM2) resultaram no percentual de utilização de 8,64% e 60,75%, respectivamente. Consequentemente, a ociosidade (*Idle*) da CPU relativa às duas instâncias simultâneas baseadas em contêineres e VMs resultaram nas médias de utilização de 91,21% e 38,98%, respectivamente. Por fim, a utilização da CPU relativa aos três contêineres (C1+C2+C3) e as VMs (VM1+VM2+VM3) resultaram nas médias 10,77% e 87,3%, respectivamente. Já a ociosidade da CPU relativa às três instâncias simultâneas baseadas em contêineres e VMs resultaram nas médias de 89,10% e 12,58%, respectivamente.

Os resultados provenientes das análises demonstraram uma sobrecarga drasticamente inferior na utilização da CPU pelos contêineres em comparação às VMs, levando em conta todos os cenários. É importante destacar que os contêineres *Docker* são empacotados em imagens contendo apenas arquivos, dependências e bibliotecas necessárias ao funcionamento da aplicação, bem como são carregados sobre o *kernel* do sistema hospedeiro. Sendo assim, os ambientes provisionados por contêineres apresentaram leves cargas na utilização da CPU, porque a tecnologia dispensa a sobrecarga adicional do *Hypervisor* e do SO convidado. Isto é, uma VM (VM1) do Cenário I representou um acréscimo de 23,42% de sobrecarga na utilização da CPU em relação a um contêiner (C1). Em seguida, considerando os resultados do Cenário II, o impacto das duas VMs (VM1+VM2) na utilização da CPU representaram um acréscimo de 52,21% de sobrecarga em comparação aos dois contêineres (C1+C2). Por fim, as três VMs (VM1+VM2+VM3) do Cenário III representaram um acréscimo de 76,53% de sobrecarga na utilização da CPU em comparação aos contêineres (C1+C2+C3). Portanto, o *Hypervisor*, as VMs e os SOs convidados utilizados nos escopos II dos cenários I, II e III geram em torno de 23,42%, 52,21% e 76,53% de sobrecargas adicionais na utilização da CPU, respectivamente. Além disso, os contêineres apresentaram um melhor equilíbrio na utilização da CPU à

medida que um contêiner era adicionado, a sobrecarga da CPU aumentava levemente em aproximadamente 1,82%, ao passo que, cada VM adicionada aumentava a sobrecarga da CPU em 28,89%.

7.1.4 Consumo de Energia

Os primeiros experimentos levaram em consideração um período de ausência de cargas de trabalhos destinadas aos contêineres e VMs. Dessa forma, o estado de ociosidade do computador hospedeiro foi observado pelo medidor *watts up power meter* que forneceu amostras da demanda de potência média. Durante um intervalo de tempo, a medição da energia consumida observou a ausência de carga de trabalho direcionada à operação fornecida pelo computador hospedeiro. As coletas forneceram 60 amostras contendo a potência média demandada por cada cenário e seus respectivos escopos. As potências médias serviram de referência para derivar novas medidas de consumo de energia relacionando algumas variações de tempo e uma tarifa média cobrada por kWh. O tempo observação do estado de ociosidade teve uma duração de 30 minutos e cada coleta de amostra considerou intervalos de 30 segundos. A relação entre a potência média e a utilização da CPU foram representadas graficamente com base nos resultados oriundos das cargas de trabalhos.

O consumo de energia atribuído apenas à inatividade representou uma potência base média de 27,02 *Watts*. Esse valor representa um custo fixo operacional apenas para manter a infraestrutura de ligado sem receber cargas de trabalhos dos clientes. Logo, o consumo de energia gerado a partir do valor 27,02 *Watts* representa de fato a potência demandada pelo processamento das cargas de trabalhos. Sendo assim. as análises posteriores utilizaram a potência base média (27,02 W) como valor de referência para derivar novos cálculos com base na Equação 6.1. A aplicação da equação 6.1 se baseou na média 27,02W em substituição à variável PB para derivar o valor médio em kilowatt (kW). O resultado da conversão pela fração correspondeu a 0,02702 kW. Essa nova unidade serviu de multiplicador para as variações de tempos substituídos na variável $\triangle t$. Os cálculos da 6.1 adotaram unidades de tempos com períodos variando em horas. Os períodos adotados nas análises de consumo de energia corresponderam a 24h; 720h (24 x 30); e 8,640h (24 x 30 x 12). Sendo esses valores equivalentes a um dia, um mês e um ano, respectivamente. Logo, os

resultados relativos ao consumo de energia corresponderam a 0,648kW/dia; 19,45kW/mês; e 233,45kW/ano. Este estudo tratou como potência demandada (PD) qualquer consumo de energia gerado a partir da potência base (PB ou ocioso) e a potência total (PT) foi derivada da soma entre PD e PB. É importante salientar que o valor da PD tende a crescer em decorrência do acréscimo de instâncias aptas a processar as cargas de trabalhos.

A Tabela 9 representa o consumo de energia em razão das instâncias agregadas ao estado de ociosidade do computador hospedeiro. Então, em virtude das amostras conferidas no Cenário I, a PD pelo contêiner C1 acrescida da PB do estado de ociosidade resultou na PT dada pela soma a seguir: 2,38W (PD) + 27,02W (PB) = 29,40W (PT). Já a PD pela VM1 acrescida da PB do estado de ociosidade resultou na PT dada pela soma a seguir: 15,77W (PD) + 27,02W PB) = 42,79W (PT). Consequentemente, as amostras conferidas às operações dos contêineres C1+C2 do Cenário II resultaram nas PD e PT dadas pelos valores: 3,00W (PD) e 30,02W (PT). No caso das operações das VM1+VM2 do Cenário II resultaram nas PD e PT dadas pelos valores: 22,13W (PD) e 49,15W (PT). Por fim, as amostras conferidas aos contêineres C1+C2+C3 e às VM1+VM2+VM3 do Cenário III resultaram nas PD e PT dadas pelos valores 4,68W (PD) e 31,70W (PT); 27,96W (PD) e 54,98W (PT), respectivamente.

Sob outra perspectiva, as análises posteriores consideraram a variação do tempo e o consumo de energia em relação ao mês e ano. Sendo assim, a análise do Cenário I incluiu apenas uma instância baseada no contêiner C1 e uma baseada na VM VM1 da operação com intuito de calcular a energia consumida de ambas as instâncias isoladamente. Considerando a operação do contêiner C1 no Cenário I, os resultados da PD e PT foram dados pelos valores 1,71kW/mês e 21,17kW/mês; 20,53kW/ano e 253,99kW/ano. Da mesma forma, a operação da VM1 produziu os resultados da PD e PT dados pelos valores 11,36kW/mês e 30,81kW/mês; 136,27kW/ano e 369,72kW/ano. Em termos percentuais do consumo de energia do Cenário I, a operação do contêiner C1 e da VM1 demandaram uma potência demandada de aproximadamente 8,80% e 58,4%, respectivamente.

Tabela 9 – Relação do consumo de energia considerando a quantidade de instâncias baseadas em contêineres e VMs

QTD de instâncias	Potência Demandada (W)	Potência Total (W)	Estatísticas		Potência Demandada (kW/dia)	Potência Total (kW/dia)	Potência Demandada (kW/mês)	Potência Total (kW/mês)	Potência Demandada (kW/ano)	Potência Total (kW/ano)	
			IC Watts	SD Watts							
Cenário I - Escopo I	2.38	29.40	[29.36;29.43]	0.12	0.057	0.71	1.71	21.17	20.53	253.99	8.8%
Cenário I - Escopo II	15.77	42.79	[42.73;42.85]	0.24	0.379	1.03	11.36	30.81	136.27	369.72	58.4%
Cenário II - Escopo I	3.00	30.02	[29.96;30.07]	0.21	0.072	0.72	2.16	21.61	25.92	259.37	11.1%
Cenário II - Escopo II	22.13	49.15	[49.05;49.23]	0.35	0.531	1.18	15.93	35.38	191.16	424.61	81.9%
Cenário III - Escopo I	4.68	31.70	[31.61;31.79]	0.35	0.112	0.76	3.37	22.83	40.45	273.90	17.3%
Cenário III - Escopo II	27.96	54.98	[54.93;55.03]	0.18	0.671	1.32	20.13	39.59	241.59	475.04	103.5%

Considerando a análise de consumo de energia do Cenário II, as operações das duas instâncias paralelas dos contêineres C1+C2 produziram os resultados da PD e PT dados pelos valores: 2,16kW/mês e 21,61kW/mês; 25,92kW/ano e 259,37kW/ano. Ao passo que as operações das duas instâncias paralelas das VM1+VM2 produziram os resultados da PD e PT dados pelos valores 15,93kW/mês e 35,38kW/mês; 191,16kW/ano e 424,61kW/ano. Em termos percentuais do consumo de energia do Cenário II, as operações dos contêineres C1+C2 e das VM1+VM2 demandaram uma potência demandada de aproximadamente 11,1% e 81,9%, respectivamente. Na análise de consumo de energia do Cenário III, as operações das três instâncias paralelas dos contêineres C1+C2+C3 produziram os resultados da PD e PT dados pelos valores 3,37kW/mês e 22,83kW/mês; 40,45kW/ano e 273,90kW/ano. Já as operações das três instâncias paralelas das VM1+VM1+VM3 produziram os resultados da PD e PT dados pelos valores 20,13kW/mês e 39,59kW/mês; 241,59kW/ano e 475,04kW/ano. Em termos percentuais do consumo de energia do Cenário III, as operações dos contêineres C1+C2+C3 e das VM1+VM2+VM3 demandaram uma potência demandada de aproximadamente 17,3% e 103,5%, respectivamente. A Figura 12 (a) representa graficamente os valores médios das energias consumidas em decorrência das demandas destinadas às instâncias agrupadas nos cenários experimentais.

A segunda análise de consumo de energia conferiu as diferenças percentuais em virtude do acréscimo de instâncias na operação. Para este propósito, a subtração efetivada entre o valor da potência demandada pela operação atual e a operação posterior mostraram os valores acrescidos em decorrência do aumento do nível da operação do Cenário I para o Cenário II e, consequentemente, do Cenário II para o Cenário III. Dessa forma, observou-se que a diferença entre a capacidade da operação oferecida pelo contêiner C1 subtraída da capacidade da operação dos contêineres C1+C2 correspondeu a um aumento de 0,62W ou 2,12%. Enquanto a diferença entre a capacidade da operação oferecida pelos contêineres C1+C2 subtraída da capacidade da operação dos contêineres C1+C2+C3 correspondeu a um aumento de 1,68W ou 5,6%. Considerando os cenários provisionados por VMs, observou-se que a diferença entre a capacidade da operação oferecida pela VM1 subtraída da capacidade da operação das VM1+VM2 correspondeu a um aumento de 6,35W ou 14,85%, enquanto a diferença entre a capacidade da operação oferecida pelas VM1+VM1 subtraída da capacidade da operação das VM1+VM2+VM3 correspondeu a um aumento de 5,84W ou 11,88%. Esses resultados indicaram que as operações baseadas em VMs

apresentaram consumo de energia mais acentuados do que contêineres.

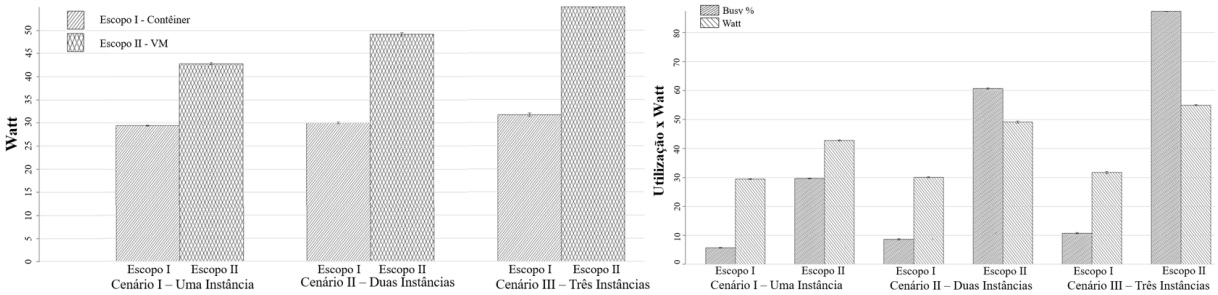

Figura 12 – Relação Utilização da CPU e Consumo de energia dos cenários baseados em VMs e contêineres: (a) Potência demandada pelos escopos - (b) Relação Utilização (Busy) x Potência (Watt)

De acordo com a Figura 12 (b), o relacionamento entre os fatores potência (Watt) e a utilização da CPU (Busy) cresceram conforme a capacidade de operação dos cenários I, II e II suportaram maiores volumes de cargas de trabalhos. No entanto, os cenários baseados em contêineres entregaram os serviços com menores níveis de utilização da CPU e potência demandada. Considerando os Escopos I de todos os cenários, as cargas de trabalhos demandaram praticamente um terço dos recursos do computador hospedeiro e o consumo de energia apresentou uma variação de potência bem equilibrada. Na perspectiva do melhor aproveitamento dos recursos computacionais, os resultados apresentados no Escopo II do Cenário III descartaria a possibilidade da inclusão de uma nova instância baseada em VM, visto que os níveis de utilização da CPU estão próximos da saturação. Nesse caso, seria necessário ampliar a capacidade computacional da infraestrutura ou fazer novos provisionamentos de recursos para balancear as cargas de trabalhos. Em contrapartida, os resultados apresentados no Escopo I do Cenário III permitiria a inclusão de novas instâncias baseadas em contêineres, visto que o computador hospedeiro apresentou cerca de dois terços da sua capacidade de processamento disponível. Além do mais, a economia de energia do Escopo I em relação ao Escopo II do Cenário III foi de 23,28W ou 73,43% a favor dos contêineres.

7.1.5 Custo do Consumo de Energia

A análise de consumo de energia possibilita um maior controle e gestão nos gastos com energia elétrica dos *data centers*. Além do mais, uma boa prática de economia de energia promove uma cultura de preservação do meio ambiente e ainda por cima

proporciona um melhor custo-benefício de médio e a longo prazo na utilização de recursos computacionais. Assim, os resultados obtidos dos consumos de energia permitem realizar uma análise de custo com base no valor médio da tarifa cobrada por kWh. De acordo com agência nacional de energia elétrica (ANEEL, 2020), a tarifa média nacional de energia por kilowatt-hora custa em torno de R$ 0,576. Sendo assim, as projeções de custos deste trabalho levaram em conta uma tarifa de R$ 0,576 por kWh como referência para obter os resultados apresentados na Tabela 9. Para tanto, a equação 7.1 foi uma extensão da equação 6.1 com a inclusão da varável $Tarifa$ que representa o preço do kWh cobrado pela concessionária de energia elétrica (dado em R$).

Conforme apresentado anteriormente, o consumo de energia da operação do computador hospedeiro no estado ocioso correspondeu às seguintes médias: 0,648kW/dia, 19,45kW/mês e 233,45kW/ano. Com a aplicação da equação 7.1 nas três médias anteriores, os valores de custos médio referente ao consumo de energia foram os seguintes: R$ 0,372/dia, R$ 11,16/mês e R$ 133,96/ano, respectivamente. Então, caso o computador hospedeiro permanecesse operando no estado de ociosidade durante 30 dias ou 12 meses, os valores cobrados com base na tarifa seriam, respectivamente: R$ 11,16/mês e R$ 133,96/ano. O custo adicional se refere ao valor do custo atribuído a uma instância e o custo total é o resultado da soma entre o custo adicional e o custo associado ao estado ocioso. Portanto, a operação do Cenário I atribuída ao contêiner C1 produziu um custo adicional de R$ 0,98/mês e R$ 11,78/ano ou um aumento adicional de 8,8% no custo do estado ocioso. Em contrapartida, a operação atribuída à VM1 produziu um custo adicional mais acentuado de R$ 6,52/mês e R$ 78,19/ano, representando um aumento adicional de 58,4% no custo do estado ocioso.

$$Custo(kWh) = \frac{PB + PD}{1000} \times \triangle t \times Tarifa \tag{7.1}$$

As operações fornecidas pelas duas instâncias baseadas em contêineres C1+C2 produziram um custo adicional de R$ 1,24/mês e R$ 14,87/ano ou um aumento de 11,1% no custo do estado ocioso. Por outro lado, as operações das duas instâncias baseadas em VM1+VM2 produziram um custo adicional de R$ 9,14/mês e R$ 109,69/ano, representando um aumento de 81% em relação ao estado ocioso. Por fim, as operações fornecidas pelas três instâncias baseadas em contêineres C1+C2+C3 produziram um custo adicional de R$

1,93/mês e R$ 23,21/ano ou um aumento de 17,3% em comparação ao custo do estado ocioso. Em compensação, as VM1+VM2+VM3 produziram um custo adicional de R$ 11,55/mês e R$ 138,63/ano, representando um aumento de 103,5%. Esse último caso ultrapassou duas vezes a mais o custo relativo ao estado ocioso. A Tabela 10 apresenta o custo adicional e custo total gastos pelas instâncias agregadas ao custo associado ao estado de ociosidade.

Em suma, as operações oferecidas pelos contêineres dos Escopos I de todos os cenários (C1; C1+C2; C1+C2+C3) representaram os custos mensais na ordem de R$ 12,15/mês, R$ 12,40/mês e 13,10/mês, respectivamente. De forma semelhante, os custos anuais para manter as operações dos contêineres de todos os cenários (C1; C1+C2; C1+C2+C3) foram de R$ 145,75/ano, R$ 148,84/ano e 157,17/ano. Por outro lado, as operações oferecidas pelas VMs dos Escopos II de todos os cenários (VM1; VM1+VM2; VM1+VM2+VM3) representaram custos mensais na ordem de R$ 17,68/mês, R$ 20,30/mês e 22,72/mês, respectivamente. Da mesma forma, os custos anuais para manter as operações das VMs de todos cenários (VM1; VM1+VM2; VM1+VM2+VM3) representaram os valores na ordem de R$ 212,16/ano, R$ 243,66/ano e 272,59/ano, respectivamente. Os resultados indicam que os percentuais de custos para cada VM agregada à operação são mais acentuados em relação a cada contêiner agregado à operação.

Tabela 10 – Relação do consumo de energia e custo considerando a quantidade de instâncias baseadas em contêineres e VMs

QTD de instâncias	Consumo Total (W)	Consumo Total (kW)	Custo Adicional por dia (R$)	Custo Total por dia (R$)	Custo Adicional por mês (R$)	Custo Total por mês (R$)	Custo Adicional por ano (R$)	Custo Total por ano (R$)	
Cenário I - Escopo I	29,40	0,02940	0,03	0,40	0,98	12,15	11,78	145,75	8,8%
Cenário I - Escopo II	42,79	0,01577	0,22	0,59	6,52	17,68	78,19	212,16	58,4%
Cenário II - Escopo I	30,02	0,00300	0,04	0,41	1,24	12,40	14,87	148,84	11,1%
Cenário II - Escopo II	49,15	0,02213	0,30	0,68	9,14	20,30	109,69	243,66	81,9%
Cenário III - Escopo I	31,70	0,00468	0,06	0,44	1,93	13,10	23,21	157,17	17,3%
Cenário III - Escopo II	54,98	0,02796	0,39	0,76	11,55	22,72	138,63	272,59	103,5%

Uma outra análise avaliou o quão mais onerosa uma operação seria em relação à outra. Para tanto, as diferenças dos custos associados às operações entre os escopos de cada cenário foram calculados. Dessa forma, as diferenças dos custos das operações dos contêineres e VMs foram calculadas utilizando a substração dos custos vinculados às instâncias das mesmas tecnologia adotados nos dois escopos dos três cenários, ou seja, a diferença entre o custo anual (R$ 14,87) do Escopo I no Cenário II (C1+C2) subtraído pelo custo anual (R$ 11,78) Escopo I no Cenário I (C1). A diferença do custo anual (R$

23,21) do Escopo I do Cenário III (C1+C2+C3) subtraído pelo custo anual (R$ 14,87) Escopo I do Cenário II (C1+C2). As diferenças entre as operações dos cenários baseados em contêineres resultaram nos valores R$ 3,09 e R$ 8,34, respectivamente. Da mesma forma, a diferença do custo anual (R$ 109,69) gerada pela operação das VMs do escopo II no Cenário II (VM1+VM2) subtraído pelo custo anual (R$ 78,19) do Escopo II no Cenário I (VM1). A diferença do custo anual (R$ 138,63) gerada pela operação Escopo II no Cenário III (VM1+VM2+VM3) subtraído pelo custo anual (R$ 109,69) do Escopo II no Cenário II (VM1+VM2). As diferenças entre as operações dos cenários baseados em VMs resultaram nos valores R$ 31,50 e R$ 28,94, respectivamente. Logo, os resultados referentes aos custos atribuídos às operações dos contêineres representaram um impacto financeiro bem inferior em relação às operações das VMs.

Outro ponto observado se referiu à redução do custo para manter as operações em funcionamento. Considerando que ocorressem migrações das operações oferecidas pelas VMs para as operações baseadas contêineres. As reduções dos custos foram calculados através das diferenças dos custos gerados pelos contêineres do Escopo I menos os custos das VMs do Escopo II. Considerando as operações do Cenário I, o custo anual (R$ 78,19) atribuído à VM1 subtraído pelo custo anual (R$ 11,78) do contêiner C1. De forma semelhante, as operações do Cenário II, o custo anual (R$ 106,69) atribuído às VM1+VM2 subtraído pelo custo anual (R$ 14,87) dos contêineres C1+C2. Por fim, o custo anual (R$ 138,63) atribuído às VM1+VM2+VM3 subtraído pelo custo anual (R$ 23,21) dos contêineres C1+C2+C3. Considerando os resultados dos três cenários anteriores, as economias geradas em decorrência das migrações das operações baseadas em VMs para contêineres assegurariam reduções de custos correspondente a R$ 66,41/ano, R$ 94,82/ano e R$ 115,42/ano, respectivamente. Do ponto de vista da redução do custo, uma elevada quantidade de VMs migradas para contêineres poderia representar uma redução considerável no custo do consumo de energia atribuído à operação do *data center*.

Um aspecto relevante pontuado na Figura 13 (a) é a relação entre a vazão da rede (Mbit/s) e o consumo de energia. Apesar das instâncias dos Escopos I e II terem atingido taxas de transferências com pouca disparidade, observou-se que os tempos computacionais das VMs dos Escopos II demandaram mais energia do que os contêineres dos Escopos I. Sendo assim, independentemente do valor da tarifa cobrada pela concessionária de energia elétrica, a discrepância no consumo de energia das VMs refletirá diretamente no

custo operacional da infraestrutura. A Figura 13 (b) mostra a relação entre a vazão da rede (Mbit/s) e o custo no consumo de energia derivado a partir da tarifa de referência aplicada na equação 7.1. Os contêineres dos Escopos I representaram uma redução de custo na ordem de: 43,59% para o Cenário I; 63,71% para o cenário II; e 73,44% para o Cenário III. Portanto, considerando esse contexto avaliado, a adoção de ambientes baseados em contêineres torna-se uma alternativa viável para minimizar o custo operacional da infraestrutura e contribui diretamente na redução da emissão de gases de efeitos estufa.

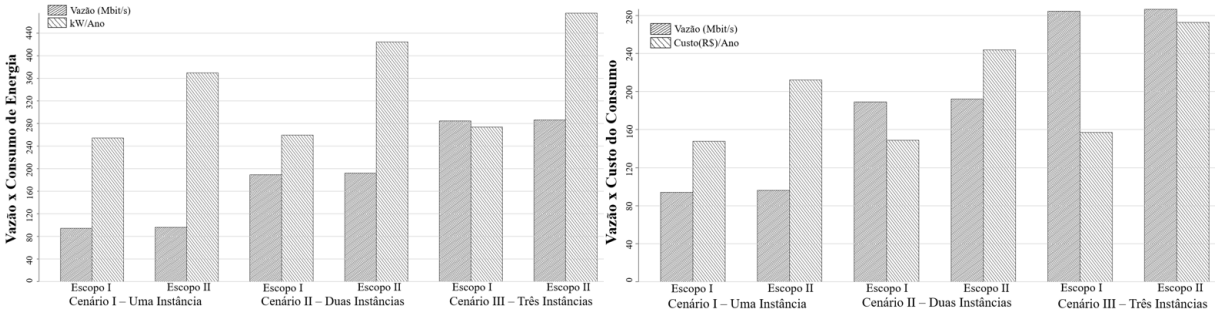

Figura 13 – Relação da demanda (Vazão) pelo serviço com o consumo de energia dos escopos baseados em contêineres e VMs: (a) Vazão x kW/Ano - (b) Vazão x Custo/Ano

7.2 Estudo de Caso II: Cenários derivados dos modelos SPNs para ambientes Moodle provisionados por Contêineres e VMs

Esta seção apresenta os resultados obtidos a partir dos experimentos realizados na segunda arquitetura experimental e das análises feitas a partir do modelos SPNs gerados. Primeiro, é apresentado a validação do modelo e, em seguida, são detalhados os quatro cenários avaliados, a saber: O cenário I analisa o desempenho da vazão em função de alguns tempos de chegados predefinidos em um intervalo. O cenário II mostra os gargalos nos tempos de respostas gerados por sucessivas cargas de trabalhos. O cenário III faz estimativas de capacidade para os agrupamentos de instâncias, visando encontrar os melhores ajustes entre a capacidade e a demanda. Por fim, cenário IV realiza análises de consumo de energia e custo com o objetivo de proporcionar um melhor custo-benefício relação a demanda por energia elétrica.

7.2.1 Validação do Modelo

O processo de validação utilizou os parâmetros e os modelos apresentados no Capítulo 6 (Modelo Analítico). As Tabelas 4, 5 e 6 apresentam as transições, as métricas e os parâmetros adotados nesse modelo, respectivamente. A validação foi conduzida com base em cinco cenários experimentais, classificados como turmas de tamanhos diferentes (de 1 a 20 alunos). Os experimentos realizados nos ambientes baseados em contêineres e VMs produziram amostras com 60 médias relativas à vazão das requisições por segundos. As amostras foram analisadas isoladamente conforme os tamanhos das turmas. As cargas de trabalhos definidas para as cinco turmas assumiram os seguintes montantes de requisições: 13, 65, 130, 195 e 260. Esses valores foram adotados por considerar turmas compostas pelas quantidades de alunos, ao invés do montante de requisições. Dessa forma, cada aluno individual representou uma carga de trabalho relativa à 13 requisições pertinentes às funcionalidades frequentemente acessadas no Moodle, como acesso à página inicial, efetuar logon, visualizar um curso, visualizar páginas de atividades, submeter um questionário de atividades, entrar no fórum de discussões, responder a uma discussão, enviar um arquivo de texto, efetuar logout, entre outras. Portanto, em virtude deste trabalho adotar os termos turmas e alunos para representar os cenários, os montantes de requisições foram divididos pela quantidade requisição demandada por cada aluno, por exemplo: 13/13=01-Aluno, 65/13=05-Alunos, 130/13=10-Alunos, 195/13=15-Alunos e 260/13=20-Alunos. Assim, a validação teve como referência amostras oriundas de experimentos gerados por turmas compostas pelos seguintes quantitativos de alunos: 01, 05, 10, 15 e 20. Os resultados apresentados nos cenários posteriores também adotaram as denominações de turmas e alunos.

A Tabela 11 apresenta as configurações inseridas nos modelos e nos ambientes baseados em contêineres e VMs. Os cenários avaliados no sistema real adotaram apenas uma instância de contêiner e outra de VM. Sendo assim, os modelos do contêiner e da VM utilizaram apenas um *token* atribuído às constantes `CT-Número_de_Instâncias` e `VM-Número_de_Instâncias` para indicar as instâncias contidas nos lugares `Docker` e `KVM`, os quais representam os comportamentos de um contêiner e uma VM utilizada no sistema real. No *Jmeter*, os tempos de chegadas entre as requisições foram definidos seguindo uma distribuição exponencial entre intervalos aleatórios de 100 milissegundos. O *Jmeter* oferece uma funcionalidade na qual são definidos tempos aleatórios entre as

chegadas das requisições. Neste trabalho, foi utilizado um *script* em java que simulou os tempos de chegadas baseados em uma função de distribuição exponencial. Em ambos modelos, as constantes `CT-Tempo_de_Chegada` e `VM-Tempo_de_Chegada` fazem referência a um atraso de 100 milissegundos atribuídos aos parâmetros *delay* das transições `CT-T_Chegada_Fila` e `VM-T_Chegada_Fila`, respectivamente. Já os *delays* definidos nas constantes `CT-Tempo_do_Serviço` e `VM-Tempo_do_Serviço` foram atribuídas às transições `CT-T_Serviço` e `VM-T_Serviço` para representarem os atrasos no processamento das requisições. Considerando que foram observadas cinco turmas com diferentes escopos para cada experimento dos contêineres, os tempos de serviços atribuídos à constante `CT-Tempo_do_Serviço` corresponderam a 0.84854, 0.52849, 0.51101, 0.59981, 0.59119 para turmas compostas por 01, 05, 10, 15 e 20 alunos, respectivamente. Da mesma forma, os experimentos das VMs apresentaram tempos de serviços atribuídos à constante `VM-Tempo_do_Serviço` correspondente 0.80141, 0.50669, 0.49895, 0.57313, 0.59726, respectivamente. Os *tokens* pertinentes as tamanhos das cinco turmas foram armazenados nas constantes `CT-Número_de_Requisição` (e `VM-`).

Tabela 11 – Configuração da validação dos modelos contêineres e VMs

Modelo Contêiner		**Modelo VM**	
Parâmetro	**Valor**	**Parâmetro**	**Valor**
CT-Número_de_Requisição	13 req	VM-Número_de_Requisição	13 req.
CT-Tempo_de_Chegada	100 ms	VM-Tempo_de_Chegada	100 ms
CT-Tempo_de_Serviço	0.8485 seg.	VM-Tempo_de_Serviço	0.8014
CT-Número_de_Instâncias	1 contêiner	VM-Número_de_Instâncias	1 VM

Tabela 12 – Validação do modelo analítico

Alunos	**Experimentos no Contêiner**		**Modelo do Contêiner**	**Experimentos na VM**		**Modelo da VM**
	Média	**IC**	**Média**	**Média**	**IC**	**Média**
01	1.17850	[1.1156;1.2413]	1.12919	1.24780	[1.1917;1.304]	1.20936
05	1.89220	[1.799;1.9854]	1.82472	1.97360	[1.8754;2.0717]	1.90691
10	1.95690	[1.8929;2.0209]	1.92697	2.00420	[1.9247;2.0836]	1.96029
15	1.66720	[1.6223;1.7121]	1.64683	1.74480	[1.6994;1.7902]	1.72798
20	1.69150	[1.653;1.7299]	1.67789	1.67430	[1.6438;1.7048]	1.66626

De acordo com os resultados apresentados na Tabela 12, os experimentos realizados no sistema real e análises dos modelos apresentaram diferenças significativamente pequenas, no que diz respeito à vazão. Considerando os ambientes baseados em contêineres,

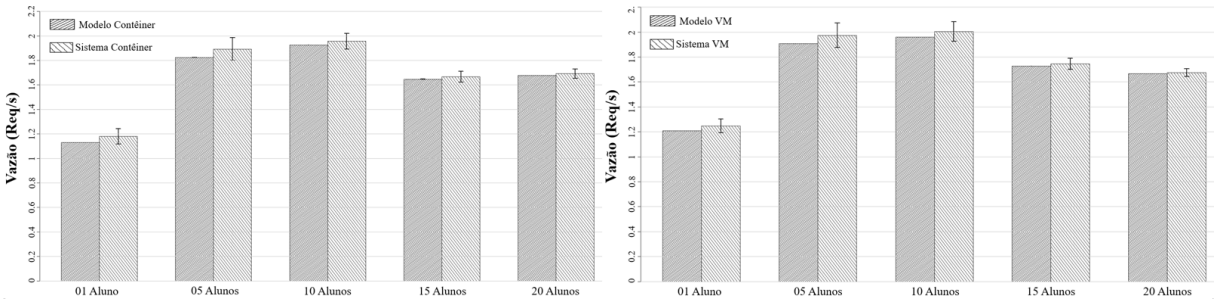

Figura 14 – Comparação dos resultados para a vazão do sistema obtidos através dos experimentos e dos resultados obtidos através do modelo SPNs utilizando contêineres (a) e VMs (b).

observou-se que as turmas contendo 1, 5, 10, 15 e 20 alunos obtiveram os intervalos de confiança correspondentes a [1.1156;1.2413], [1.799;1.9854], [1.8929;2.0209], [1.6223;1.7121], [1.653;1.7299], respectivamente. Ao passo que, o modelo do contêiner obteve as vazões médias correspondentes a 1.17850 req/s, 1.89220 req/s, 1.95690 req/s, 1.66720 req/s, 1.69150 req/s, respectivamente. Consequentemente, os ambientes baseados em VMs consideraram os mesmos tamanhos de turmas, as quais resultaram nos intervalos de confiança correspondentes a [1.1917;1.304], [1.8754;2.0717], [1.9247;2.0836], [1.6994;1.7902], [1.6438;1.7048], respectivamente. Enquanto, o modelo da VM obteve as vazões médias correspondentes a 1.20936 req/s, 1.90691 req/s, 1.96029 req/s, 1.72798 req/s, 1.66626 req/s, respectivamente. De acordo com a Figura 14, os resultados dos modelos conseguiram representar o comportamento do sistema real, uma vez que os resultados estão dentro do intervalo de confiança (IC) de 95% obtidos através dos experimentos. Dessa forma, pode-se afirmar que o comportamento do modelo apresentou resultados precisos e consistentes em comparação o sistema real. Assim, diferentes cenários podem ser analisados a partir dos modelos SPNs proposto, sem a necessidade da execução de novos experimentos. É importante destacar que o desempenho do sistema com contêineres e VMs entregaram os serviços com tempos computacionais aproximados.

7.2.2 Cenário I

Este cenário observou o comportamento da vazão em relação aos tempos de chegadas que variaram de 100 milissegundos (ms) até 10 segundos (seg) com intervalos de 100 ms para cada análise realizada. A ideia deste cenário foi analisar o comportamento da vazão submetida a diferentes tempos de chegadas. Considerando que a capacidade do *link* de

comunicação estivesse adequada e não apresentasse falhas, acentuados intervalos entre uma requisição e outra podem sinalizar tendências de altas ociosidades e baixas demandas de processamento. Essas periódicas quedas de gargalos podem sugerir reduções na capacidade da operação em determinados horários do dia. Este cenário realizou uma demostração desses aspectos de desempenho para que os resultados sejam úteis aos cenários posteriores.

Este estudo pressupôs uma turma com 20 alunos produzindo cargas de trabalhos, considerando apenas uma instância para o contêiner e outra para a VM. O tempo de serviço correspondeu a 0.59119 segundos para o contêiner e 0.59726 segundos para a VM. Cada avaliação forneceu um conjunto composto por 60 amostras das vazões médias influenciadas pelos efeitos dos tempos de chegadas entre as requisições. A primeira simulação adotou um tempo de chegada de 100 ms, a segunda 200 ms, a terceira 300 ms, e assim sucessivamente até a última simulação com tempo de chegada de 10 seg. Todas as simulações totalizaram 19 grupos de amostras relativas às médias das vazões de requisições por segundos. Os tempos de chegadas foram atribuídos igualmente às constantes `CT-Tempo_de_Chegada` e `VM-Tempo_de_Chegada` em referência ao *delay* das transições `CT-T_Chegada_Fila` e `VM-T_Chegada_Fila`, respectivamente. Os 20 alunos das turmas foram representados nas constantes `CT-Número_de_Requisição` e `VM-Número_de_Requisição` em referência às marcações dos lugares `CT-Cliente` e `VM-Cliente`, onde cada um deles armazenaram 260 *tokens* relativos às cargas de trabalhos com 260 requisições. As marcações dos lugares `Docker` e `KVM` forneceram apenas uma instância para as constantes `CT-Número_de_Instâncias` e `VM-Número_de_Instâncias`, respectivamente. Os tempos de serviços dos contêineres e VMs foram atribuídos à variável `CT-Tempo_do_Serviço` e `VM-Tempo_do_Serviço` em referência aos atrasos das transições `CT-T_Serviço` e `VM-T_Serviço`. É importante destacar que os experimentos realizados no sistema real forneceram as taxas de requisições por segundos que permitiram calcular os tempos de serviços de cada turma.

Os resultados para análise deste cenário são apresentados na Figura 15. Eles mostram que a vazão do sistema se manteve estável, em torno de 1.67 req/s, considerando tempos de chegadas variando entre 100 ms a 600 ms. No entanto, a partir do intervalo de 700 ms até 10 segundos, a vazão da turma caiu gradativamente até atingir uma taxa mínima de 0.1 req/s. Esse comportamento pode indicar que, em determinados períodos das 24 horas diárias, a capacidade da operação pode estar superdimensionada em relação

à demanda e, por consequência, ser um indicativo de subutilização dos recursos alocados. Outro aspecto deste cenário é que a vazão máxima do serviço correspondeu à 1.67 req/s para o menor tempo de chegada. Considerando que os parâmetros do modelo não sejam alterados, uma eventual inclusão de novas turmas implicaria no aumento da demanda, porém a vazão do serviço permaneceria inalterada em razão dos limites de processamento da instância. Além disso, a aplicação teria uma queda abrupta no seu desempenho devido ao aumento no tempo do resposta em decorrência do gargalo na operação.

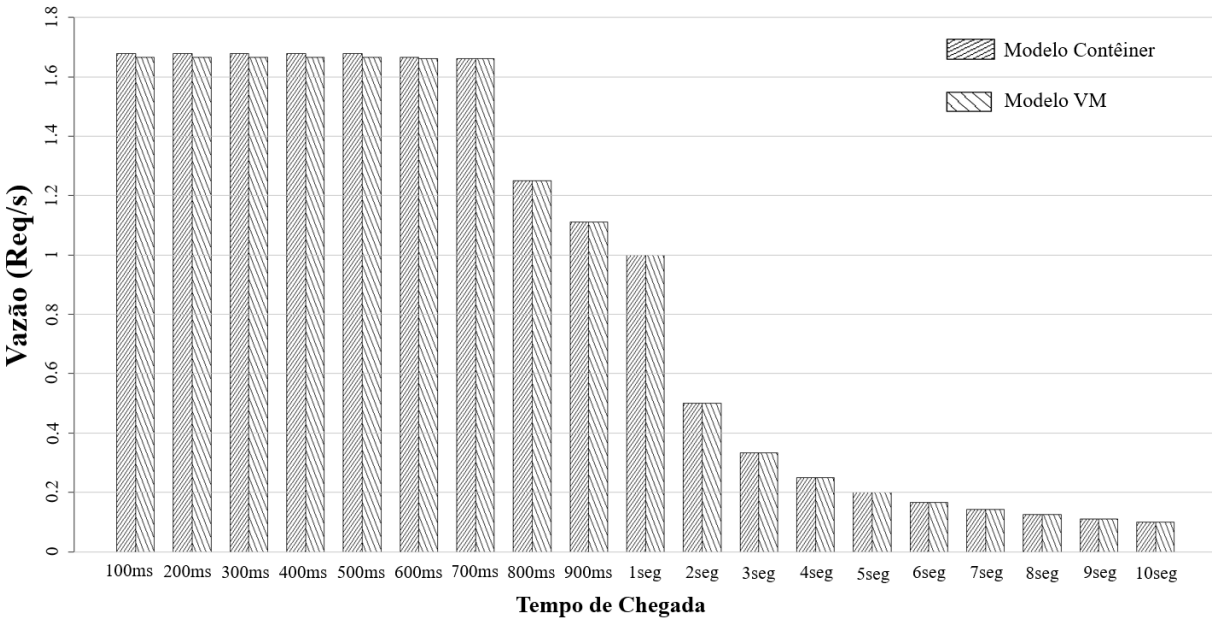

Figura 15 – Vazão por tempo de chegada nas operações baseadas em contêineres e VMs

Uma segunda análise verificou o comportamento da vazão utilizando duas instâncias, ou seja, dois *tokens* nos lugares `Docker` e `KVM`, respectivamente. Neste caso, os resultados indicaram uma estabilidade na taxa da vazão inerentes aos períodos de 100 ms e 200 ms. As taxas máximas atingidas corresponderam à 3.36 req/s para os dois contêineres e 3.33 req/s para as duas VMs. Considerando o período de 300 ms, as taxas da vazão de ambas tecnologias reduziram em aproximadamente 3.32 req/s. Todavia, a queda de desempenho foi percebida a partir do período 400ms, onde a taxa da vazão caiu para 2.50 req/s até atingir a taxa mínima de 0.10 req/s relativa ao último período de 10 segundos. Nos períodos de 100 ms e 200 ms, observou-se que os tempos de respostas corresponderam a 77.9 segundos para os dois contêineres e 78.5 segundos para as duas VMs. Já no período de 300 ms, os tempos de respostas reduziram para 25.9 e 38.6 segundos em relação aos dois contêineres e às duas VMs, respectivamente. Nesse caso, notou-se um gargalo menor

na operação a favor dos contêineres. Por outro lado, a partir do período 400 ms até 10 segundos, os tempos de respostas mantiveram-se estáveis em média de 0.60 segundos. Esse comportamento pode indicar prováveis casos de subutilização dos recursos em virtude da baixa de demanda, sugerindo redução na capacidade da operação, em caso necessidade de otimização ou redução de custo.

7.2.3 Cenário II

Este cenário avaliou o tempo de resposta em relação às cargas de trabalhos oriundas de variadas turmas. Nesta análise foram observados os gargalos gerados na operação em decorrência de aumento sucessivos de turmas com 20 alunos. Em cada simulação iniciada, uma nova turma era incrementada até atingir um limite de 20 turmas. Nesse caso, a primeira análise foi realizada com uma turma, a segunda com duas turmas, a terceira com três turmas, e assim sucessivamente até atingir o último cenário com 20 turmas. Da mesma forma, os quantitativos de alunos cresceram de 20 em 20 unidades em detrimento dos incrementos das turmas. Sendo assim, todos esses cenários geraram gradativas cargas de trabalhos em cada simulação, totalizando a maior carga de trabalho com 400 alunos que demandaram um total de 5200 requisições no processamento da operação. No modelo, as variações das turmas foram definidas nas constantes `CT-Número_de_Requisição` e `VM-Número_de_Requisição` com seus respectivos lugares `CT-Clientes` e `VM-Clientes`. Nas constantes `CT-Tempo_de_Chegada` e `VM-Tempo_de_Chegada` referenciadas pelas suas respectivas transições `CT-T_Chegada_Fila` e `VM-T_Chegada_Fila` foram definidas um atraso de 100 ms, seguindo uma distribuição exponencialmente entre as requisições. A capacidade da operação considerou apenas uma instância atribuída às constantes `CT-Número_de_Instâncias` e `VM-Número_de_Instâncias` referenciadas pelos lugares `Docker` e `KVM`, respectivamente. Em virtude de todos os cenários adotarem turmas com 20 alunos para apenas uma instância, os tempos de serviços atribuídos ao contêiner e a VM corresponderam a 0.59119 e 0.59726 segundos, respectivamente. Dessa forma, esses valores foram atribuídos às constantes `CT-Tempo_do_Serviço` e `VM-Tempo_do_Serviço` em referência aos atrasos das transições `CT-T_Serviço` e `VM-T_Serviço`.

Os resultados de cada escopos de turmas forneceram 60 amostras referentes aos seus respectivos tempos médios de respostas. Dada a análise de uma turma, os resultados

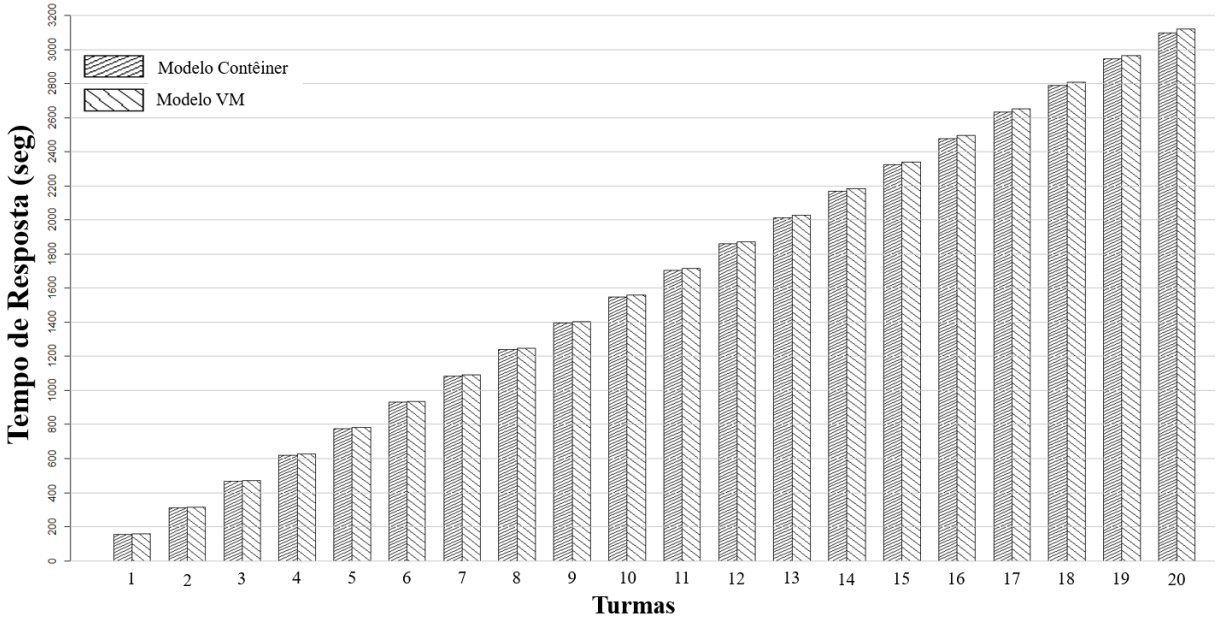

Figura 16 – Tempo de Resposta por Turmas

revelaram que os tempos médios de respostas corresponderam a 155.38 segundos para o contêiner e 156.51 segundos para a VM, uma diferença de 1.13 segundos a favor do contêiner. Considerando os tempos de serviços vinculados à operação do contêiner e VM, o tempo médio de resposta aumentou gradativamente à medida que uma turma foi introduzida para criar um cenário de maior escopo. De acordo com a Figura 16, observou-se que duas turmas em atividades simultâneas corresponderam a 310.28 e 312.54 segundos para o contêiner e VM, respectivamente. Nesse caso, a diferença nos tempos de respostas dos dois ambientes correspondeu a 2.27 segundos a favor do contêiner. No caso da última análise, ou seja, as 20 turmas com atividades paralelas, os tempos médios de respostas resultaram em 3098.48 segundos para o contêiner e 3121.16 segundos para a VM. Uma diferença de 22.68 segundos a favor do contêiner. Em efeitos práticos, essa última análise representou um cenário cuja demanda saturou consideravelmente a aplicação, a ponto de tornar inviável a entrega do serviço em virtude da capacidade da operação está subdimensionada. Essa situação sugere que sejam aplicados ajustes equilibrados nos escopos das instâncias da operação a fim de atender variações de demandas mínimas e máximas. Todavia, os cumprimentos de SLAs no menor tempo de resposta possível não pode representar a alocação de recursos em excessos. Tais decisões podem caracterizar equívocos no planejamento da capacidade, resultando em gastos desnecessários no custo operacional.

7.2.4 Cenário III

O auto-escalonamento é um mecanismo aplicado frequentemente na automação do aumento ou redução da capacidade da operação conforme limiares definidos nos parâmetros de desempenho das instâncias. Dependendo do comportamento da carga de trabalho, os números de instâncias do serviço são ajustados automaticamente, a fim de cumprir o SLA esperado. Essa característica permite que o processamento da carga de trabalho seja balanceado para outras instâncias, sem causar prejuízo ao tempo de resposta e a vazão da operação. Todavia, é importante destacar que o aumento na capacidade da operação é uma tarefa que exige cautela na definição dos limites mínimos e máximos das instâncias, visto que as demandas das turmas podem crescer ou diminuir em períodos sazonais. Caso os limiares das instâncias sejam corretamente definidos no auto-escalonamento, o tempo de resposta não causará gargalos acentuados na operação em virtude da capacidade autoajustar conforme a demanda. Outro aspecto é que o tempo de serviço geral do *cluster* tende a diminuir à medida que cresce o agrupamento de instâncias. Por consequência, a vazão poderá alcançar taxas suficientes para atender a demanda com mínimos gargalos. Dessa forma, este cenário consegue estimar agrupamentos de instâncias suficientes para equilibrar a vazão das turmas, bem como, reduzir os gargalos causados pelo aumento no tempo de resposta.

Para este cenário, foram coletadas 60 amostras relativas a 10 agrupamentos de instâncias baseadas em contêineres e VMs, isto é, cada instância provisionada no *cluster* representou um aumento no sua capacidade. O maior agrupamento foi limitado a 10 instâncias. Nesse caso, a primeira análise foi realizada com uma instância, a segunda com duas instâncias, a terceira com três instâncias, e assim sucessivamente até atingir o último cenário com 10 instâncias. A carga de trabalho foi fixada em 10 turmas com 20 alunos. Sendo assim, um total de 2600 requisições representaram as demandas das 10 turmas. Esse montante foi atribuído à variável `CT-Número_de_Requisição` (ou `VM-`), que representa o número de *tokens* no lugar `CT-Clientes` (ou `VM-`). O tempo de chegada entre as requisições considerou atrasos próximos a 100 ms conforme uma distribuição exponencial. Esse valor foi atribuído igualmente as constantes `CT-Tempo_de_Chegada` e `VM-Tempo_de_Chegada`, representando os `atrasos` das transições `CT-T_Chegada_Fila` e `VM-T_Chegada_Fila`, respectivamente. Nas constantes `CT-Tempo_do_Serviço` e `VM-Tempo_do_Servi ço` que representam os *atrasos*, foram definidas com 0.59119 e 0.59726 segundos,

respectivamente, para o tempo médio de processamento de uma requisição por uma instância de contêiner e de VM. Este cenário submeteu à análises 10 agrupamentos de instâncias que variaram entre 1 a 10 contêineres ou VMs. Esses quantitativos de instâncias foram incrementados nas constantes `CT-Número_de_Instâncias` e `VM-Número_de_Instâncias`, as quais foram referenciadas nos atributos *Marking* dos lugares `Docker` e `KVM`.

A Figura 17 apresenta os resultados obtidos nas simulações realizadas nos 10 escopos de grupamentos de instâncias. A vazão do primeiro caso resultou em 1.68 req/s para um contêiner e 1.67 req/s para uma VM. Já o tempo de resposta foi de 1549.48 e 1560.81 segundos, para um contêiner e uma VM, respectivamente. Esse resultado mostra uma degradação do desempenho da aplicação em decorrência de apenas uma instância realizar o processamento de todas as 2600 requisições demandadas. Dessa forma, uma eventual demanda oriunda de 10 turmas concorrentes tornaria praticamente inviável o funcionamento da aplicação. Na análise do segundo caso, a vazões das instâncias agrupadas atingiram as taxas 3.36 req/s para dois contêineres e 3.33 req/s para duas VMs. Já o tempo de resposta foi de a 774.95 e 780.62 segundos, considerando dois contêineres e duas VMs, respectivamente. É possível notar que o aumento no número de instâncias reduziu o gargalo da operação em aproximadamente 50%. No quarto caso, as vazões das instâncias agrupadas atingiram as taxas 6.71 req/s para quatro contêineres e 6.66 req/s para quatro VMs. O tempo de resposta obtido foi de 387.54 e 390.38 segundos para quatro contêineres e quatro VMs, respectivamente. No sexto caso, os agrupamentos com 6 instâncias atingiram vazões cujas taxas foram 10.00 req/s para os contêineres e 9.99 req/s para as VMs. A operação com 6 instâncias reduziu o gargalo em 99.07% em comparação ao primeiro caso, fazendo o tempo de resposta diminuir para 14.41 e 15.69 segundos com a utilização de contêineres e VMs, respectivamente. É importante destacar que a inclusão da instância número 7 não surtiu efeito na vazão, visto que a sua taxa permaneceu em 10.00 req/s. No entanto, ocorreu uma queda no gargalo da operação, reduzindo os tempos de respostas dos contêineres e das VMs para 0.94 e 0.86 segundos, respectivamente. Sendo assim, um agrupamento com 10 instâncias representaria um superdimensionamento na capacidade, gerando desperdícios na alocação dos recursos.

Os resultados do Cenário I mostraram que o tempo de chegada foi um atributo que impactou diretamente na taxa da vazão. Foi observado que uma instância atingiu a

uma vazão máxima assumindo os tempos de chegadas adotados. Todavia, o Cenário II mostrou que o tempo de resposta foi influenciado com gargalos em virtude do limite de processamento suportado pela instância. Neste cenário em questão, foi observado que um agrupamento com 10 instâncias pode alcançar uma vazão com uma taxa maior quando o tempo de chegada é reduzido de 100 ms para 10 ms. Verificou-se que as 10 instâncias atingiram vazões cujas taxas corresponderam a 16.8 req/s e 16.7 req/ para os contêineres e VMs, respectivamente. Adicionalmente, os tempos de respostas de ambas tecnologias corresponderam a 155.49 e 156.62 segundos, respectivamente. Para um agrupamento com 20 instâncias, a vazão representou uma taxa equivalente a 33.76 req/ para os contêineres e 33.33 req/ para as VMs. Ao passo que, os tempos de respostas de ambas tecnologias corresponderam a 78.03 e 78.60 segundos, respectivamente.

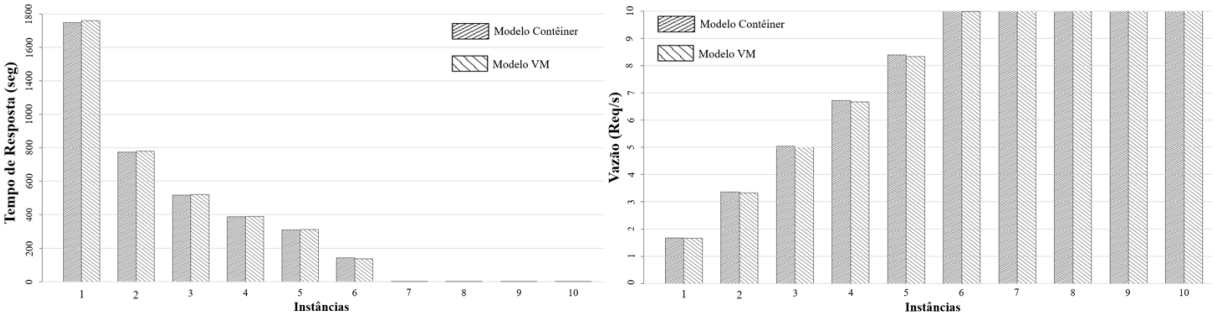

Figura 17 – a) Tempo de Resposta por instâncias - b) Vazão por instâncias

Por fim, foi analisada uma operação cujo tempo de serviço para processar uma requisição representou 0.29560 segundos para o contêiner e 0.29863 segundos para a VM. Esses valores pressupõem um *cluster* cujo poder de processamento dos nós é superior em relação aos cenários anteriores. Dessa forma, a quantidade de instâncias suportadas por 10 turmas com 20 alunos reduziu o tamanho de 6 para 3 instâncias, mantendo as mesmas taxas da vazão e os mesmos tempos de respostas. Conclui-se que a redução do número de instâncias não causaria prejuízo ao desempenho da operação. Pelo contrário, poderia trazer uma contribuição positiva na redução do consumo de energia. Além do mais, a aquisição de *clusters* cujos nós possuem mais capacidade de processamento e memória, pode ser uma opção mais vantajosa financeiramente a longo prazo, dada a economia de espaço físico e a redução na temperatura do ambiente, em virtude da diminuição na quantidade de servidores. A seção a seguir mostra um cenário que avaliar o consumo de energia e os custos dos agrupamentos de instâncias.

7.2.5 Cenário IV

Neste cenário é analisado o consumo de energia e o custo relacionado à vazão demandada pelas requisições das turmas. Nesta análise, a potência média demandada foi observada em decorrência da vazão suportada pelos agrupamentos de instâncias.

As simulações levaram em consideração um período de ausência de cargas de trabalhos destinadas aos contêineres e VMs. Para isso, o computador hospedeiro foi observado pelo medidor de consumo de energia durante um período de inatividade. Esse período de observação teve uma duração de 30 minutos e cada coleta de amostras considerou intervalos de 30 segundos. As coletas realizadas no *Watts Up Power Meter* forneceram 60 amostras referentes à potência média demandada. Em razão da inatividade do computador hospedeiro demandar um valor fixo de potência elétrica, foi avaliada separadamente as potências elétricas atribuídas às cargas de trabalhos e as instâncias agrupadas. Sendo assim, os casos avaliados demandaram uma potência média desvinculada da potência média relacionada à inatividade do computador hospedeiro. O consumo de energia atribuído apenas à inatividade representou uma potência base média de 25,04 *Watts*. Esse valor representa um custo fixo operacional apenas para manter a infraestrutura de ligado sem receber cargas de trabalhos dos clientes. Logo, o consumo de energia gerado a partir do valor 25,04 *Watts* representa de fato a potência demandada pelo processamento das requisições das turmas. Sendo assim. as análises posteriores utilizaram a potência base média (25,04 W) como valor de referência para derivar novos cálculos com base na Equação 7.1.

Os experimentos consideram um *cluster* contendo até 5 instâncias que variaram os números de contêineres e VMs de 1 a 5. Uma turma com 20 alunos gerou a carga de trabalho pelo cliente *Jmeter*. A primeira medição considerou uma instância, a segunda medição duas instâncias, a terceira medição três instâncias, assim sucessivamente até atingir cinco instâncias. Considerando os resultados obtidos, as amostras indicaram que cada instância agregada representa uma potência média correspondente a 2.06 W para os contêineres e 4.5 W para as VMs. A energia consumida foi calculada de acordo com a Equação 6.1, onde a variável PB representou a potência (W) base consumida pelo computador hospedeiro em inatividade, ao passo que a variável PD referenciou a potência demandada pelas instâncias em atividades. A constante 1000 é usada para a conversão da unidade de potência *Watt* para a unidade *kilowatt* (kW). A variável $\triangle t$ corresponde ao período de um ano relativo à energia demandada no processamento realizado pelos

escopos de instâncias definidos nas simulações. A unidade de tempo foi representada por horas, ou seja, um ano em horas correspondeu ao produto de 24 horas, 30 dias e 12 meses, totalizando 8.640 horas. No modelo, a métrica `kiloWatt_por_Tempo` (ver Tabela 5) adotou a Equação 6.1, multiplicando pelos números de instâncias em operação. Para efetuar os cálculos de custos, a Equação 7.1 foi escolhida em decorrência da variável $Tarifa$ representar o valor relativo à tarifa média nacional de R$ 0,576 por kWh.

As medições demonstraram que tanto os contêineres quanto as VMs sem processar as cargas de trabalhos não indicaram impacto na utilização CPU, independentemente da quantidade de instâncias utilizadas nos cinco cenários experimentais. A ausência de demanda destinadas aos agrupamentos de instâncias manteve o percentual de ociosidade da CPU próximo a 99,8%. Assim, a ausência de demanda no computador hospedeiro representou um cenário cujo consumo de energia anual seria de 216.35 kW/ano. Dessa forma, o custo fixo anual para mantê-lo em ligado sem demanda seria em torno de R$ 124,62. É importante destacar que o consumo de energia anual e o custo anual foram calculados pela 7.1 com base na potência base (25,04 W). Portanto, ambos são valores que não possuem cargas de trabalhos vinculadas. Por outro lado, a potência demandada corresponde apenas aos valores gerados a partir das cargas de trabalhos processadas em casos de demandas, ou seja, desconsidera a potência base consumida pela inatividade do computador hospedeiro. Consequentemente, a potência total (PT) refere-se ao montante derivado da soma entre as variáveis PB e PD. Essa segunda variável teve a potência variada conforme as quantidades de contêineres (2.06 W) e de VMs (4.5 W) em processamento.

O tempo de chegada entre as requisições foi um fator determinante no valor da taxa da vazão, pois quando o tempo de chegada foi reduzido de 100 ms para 10 ms, os agrupamentos de instâncias conseguiram atingir suas maiores taxas da vazão. Essas elevadas taxas atribuídas às capacidades dos 10 agrupamentos de instâncias permitiram a métrica `kiloWatt_por_Tempo` calcular a potência média demandada oriundas dos fluxos de requisições das turmas. No modelo, as instâncias consideradas foram atribuídas ao lugares `CT-Req_Processamento` e `VM-Req_Processamento` conforme os tempos de chegadas atrelados aos fluxos de requisições da turma. O cenário III mostrou que 100 ms entre as chegadas das requisições não foi suficiente para exaurir a capacidade do agrupamento com 10 instâncias. No entanto, a Tabela 13 mostra que 10 milissegundos conseguiram produzir uma vazão equivalente a 16.79 req/s para os 10 contêineres e 16.66

req/s para as 10 VMs. Nesse caso, todas as instâncias foram utilizadas no processamento e os tempos de respostas apresentaram gargalos relativos a 155.48 segundos para os contêineres e 156.62 segundos para as VMs.

Tabela 13 – Consumo de energia e custo anual em relação aos agrupamentos de instâncias

Quantidade de Instâncias	Projeção Anual dos Ambientes baseados em Contêineres					Projeção Anual dos Ambientes baseados em VMs					Aumento % da VM
	Vazão	kW/ano	Total-kW/ano	Custo/ano	Total-Custo/ano)	Vazão	kW/ano	Total-kW/ano	Custo/ano	Total-Custo/ano	
01-Instância	1.68	17.80	234.14	R$ 10.25	R$ 134.87	1.67	38.88	255.23	R$ 22.39	R$ 147.01	9.00%
02-Instâncias	3.36	35.60	251.94	R$ 20.50	R$ 145.12	3.33	77.76	294.11	R$ 44.79	R$ 169.40	16.74%
03-Instâncias	5.04	53.40	269.74	R$ 30.76	R$ 155.37	5.00	116.64	332.99	R$ 67.18	R$ 191.80	23.45%
04-Instâncias	6.71	71.19	287.54	R$ 41.01	R$ 165.62	6.67	155.52	371.87	R$ 89.58	R$ 214.19	29.33%
05-Instâncias	8.39	88.99	305.34	R$ 51.26	R$ 175.87	8.33	194.40	410.75	R$ 111.97	R$ 236.59	34.52%
06-Instâncias	10.07	106.79	323.14	R$ 61.51	R$ 186.13	10.00	233.28	449.63	R$ 134.37	R$ 258.98	39.14%
07-Instâncias	11.75	124.59	340.93	R$ 71.76	R$ 196.38	11.66	272.16	488.51	R$ 156.76	R$ 281.38	43.28%
08-Instâncias	13.43	142.39	358.73	R$ 82.02	R$ 206.63	13.33	311.04	527.39	R$ 179.16	R$ 303.77	47.01%
09-Instâncias	15.11	160.19	376.53	R$ 92.27	R$ 216.88	15.00	349.92	566.27	R$ 201.55	R$ 326.17	50.39%
10-Instâncias	16.79	177.98	394.33	R$ 102.52	R$ 227.13	16.66	388.80	605.15	R$ 223.95	R$ 348.56	53.46%

De acordo com as projeções anuais apresentadas na Tabela 13, a energia demandada por um contêiner e uma VM correspondeu a 17.80 kW/ano e a 38.88 kW/ano, respectivamente. Em decorrência da soma entre a potência demandada com a potência base referente a 216.35 kW/ano, os valores totais das duas operações corresponderam a 234,14 kW/ano e a 255.2 kW/ano, respectivamente. Os custos do caso 01 corresponderam a R$ 134.87 para os contêineres e R$ 147.01 para as VMs. Em relação à potência base, o contêiner e a VM do caso 01 representaram um aumento percentual de 8.23% e 17.97%, respectivamente. Já em relação ao contêiner, a VM apresentou uma diferença de 21.09 kW/ano no consumo de energia total e de R$ 12.14 no custo total, resultando no aumento percentual de 9.00% em relação ao contêiner. Considerando as três instâncias agrupadas no caso 03, a potência elétrica demandada pelos contêineres e VMs corresponderam a 53.40 kW/ano e a 116.64 kW/ano, respectivamente. Os valores totais das duas operações corresponderam a 269.74 kW/ano e a 332.99 kW/ano, respectivamente. Na perspectiva financeira do caso 03, ambas tecnologias gerou custos totais referentes a R$ 155.37 e a R$ 191.81, respectivamente. Em relação à PB, as instâncias alocadas no caso 03 representaram aumentos percentuais de 24.68% e de 53.91%, respectivamente. Nesse caso, as VMs apresentaram diferenças de 63.24 kW/ano no consumo de energia total e de R$ 36.43 no custo total, resultando no acréscimo percentual de 23.45 % em relação aos contêineres. Considerando as 06 instâncias agrupadas no caso 06, a potência elétrica demandada pelos contêineres e VMs corresponderam a 106.79 kW/ano e a 233.28 kW/ano, respectivamente. Os valores totais das duas operações corresponderam a 323.14 kW/ano e a 449.93 kW/ano, respectivamente. Na perspectiva financeira do caso 06, ambas tecnologias

geraram custos totais referentes a R$ 186.13 e a R$ 258.98, respectivamente. Em relação à PB, as instâncias alocadas no caso 06 representaram aumentos percentuais de 49.36% e de 107.83%, respectivamente. Nesse caso, as VMs apresentaram diferenças de 126.49 kW/ano no consumo de energia e de R$ 72.86 no custo total, resultando no acréscimo percentual de 39,14% em relação aos contêineres. As demais análises estão expostas na Tabela 13, as quais apresentaram projeções mais favoráveis aos contêineres acerca do consumo de energia e custo.

Todos dos resultados indicaram que as operações baseadas em VMs apresentaram um consumo de energia mais acentuados do que contêineres. Sobretudo quando os escopos aumentaram de capacidade à medida que as instâncias foram agregadas. Por exemplo, tendo como base o valor de referência, a última análise do caso 10 mostrou que o agrupamento com 10 instâncias geraram um aumento percentual de 82.27% para os contêineres e 179.71% para as VMs. Quando a comparação ocorreu entre contêineres e VMs cujos casos tiveram os mesmos tamanhos, observou-se que a operação da VM do caso 01 demandou 9.00% a mais de energia elétrica do que o contêiner do caso 01. Adicionalmente, a operações das VMs caso 10 demandaram 53,46% a mais de energia elétrica do que os contêineres do caso 10. Na perspectiva do melhor aproveitamento dos recursos computacionais, um computador hospedeiro suportaria uma maior quantidade de instâncias baseadas em contêineres, visto que as VMs tendem a ocupar maiores parcelas dos níveis de utilização da CPU. Do ponto de vista da redução do custo, uma elevada quantidade de operação baseadas em VMs migradas para contêineres poderia representar uma redução considerável no custo do consumo de energia atribuído à operação do *data center*. Por isso, independentemente do valor da tarifa cobrada pela concessionária de energia elétrica, a diferença na energia demandada pelas VMs refletiria de médio a logo prazo no custo operacional da infraestrutura. Portanto, a adoção de ambientes baseados em contêineres torna-se uma alternativa viável na minimização do custo operacional do *data center*. Além de contribuir diretamente na redução da emissão de gases de efeitos estufa. Muito embora, a proposta desta dissertação não é sugerir preferencialmente uma tecnologia em detrimento de outra, uma vez que ambas podem ser complementares na composição de arquiteturas híbridas para diversas aplicações.

7.3 Considerações Finais

Este capítulo abordou as discussões sobre as análises realizadas nesta pesquisa acerca da avaliação de desempenho, consumo de energia e custo de ambientes provisionados por contêineres e VMs. Neste estudo, foram adotadas duas arquitetura experimentais destinadas a dois estudos de casos, onde o primeiro foi formulado com o intuito de fazer avaliações e comparação relacionadas ao desempenho, ao consumo de energia e ao custo, ao passo que o segundo foi concebido como intuito de ampliar a abrangência da pesquisa com a adotação de modelos SPNs. O Estudo de Caso I abordou uma série de análises e demostração submetidas a cenários que observaram a vazão em Mbit/s, a utilização de CPU, o consumo de energia e o custo. O Estudo de Caso II apresentou uma série de análises realizadas em cenários gerados pelos modelos SPNs que permitiram encontrar configurações mais otimizadas em relação ao desempenho, ao consumo de energia e custo.

8 Conclusões

A transformação digital são iniciativas adotadas pelos negócios com o intuito de buscar oportunidades de crescimento e melhoria nas operacionais, com adoção de tecnologias garantam mais qualidade e agilidade aos serviços de TI vinculados aos negócios. Essas iniciativas têm o propósito de impulsionar a inovação nas organizações, com o auxílio de práticas que visem modernizar as aplicações, os processos de negócios e as infraestruturas computacionais. Além disso, com o passar dos anos, os usuários finais têm sido mais dependentes de recursos tecnológicos, devido ao aumento da popularidade do acesso à internet e a propagação dos computadores pessoais, dispositivos móveis de quaisquer natureza, *Smart* TVs, dispositivos com internet das coisas, entre outras. Devido ao distanciamento social em virtude da pandemia do coronavírus, o processo de transformação digital foi acelerado, em decorrência do crescimento das demandas de serviços online, como, por exemplo, teletrabalho, telemedicina, jogos online, serviços de *streaming* aplicados a entretenimento e educação à distância. Essas tendências exigiram a adoção de *data centers* modernos com capacidade para suportar ambientes escaláveis e ajustáveis, visando processar grandes volumes de cargas de trabalhos.

Os *data centers* representam infraestruturas computacionais construídas para processar grandes volumes de dados com características diversas. Em regra, eles concentram uma série de nós que podem ser combinados para formar um ou mais *clusters*, com o propósito de aumentar a capacidade da arquitetura. Essa fusão de nós permite compartilhar os recursos entre várias pilhas de serviços ou aplicações que executam tarefas distribuídas entre muitas réplicas. Dessa forma, as capacidades dos ambientes podem escalar dinamicamente conforme as demandas geradas pelas cargas de trabalhos. Devido ao poder computacional desses agrupamentos, a utilização dos recursos e o consumo de energia tendem ser elevados e, consequentemente, o custo associado a energia demandada.

O mecanismo de auto-escalonamento consiste em uma técnica de elasticidade de capacidade de réplicas, implementado na maioria dos sistemas de orquestração de *clusters*. Isso permite ajustar a capacidade da operação em relação ao volume de requisições em processamento, de modo a reduzir os gargalos na operação ou evitar desperdícios de recursos alocados. Essa característica pode trazer benefícios como a otimização na utilização dos recursos e na redução do consumo de energia. Contudo, não é uma tarefa simples definir

as melhores configurações de ambientes para atender as demandas para variados cenários concretos. Normalmente, ocorre dos limites de recursos serem definidos sem critérios precisos, baseados apenas em experiências empíricas. Todavia, as técnicas de modelagens podem auxiliar na identificação de valores aproximados para todos os parâmetros de configuração dos ambientes. Com a adoção de modelos analíticos é possível predizer os limites mínimos e máximos das instâncias adequadas às políticas de auto-escalonamento.

Nesse sentido, esta dissertação propôs uma abordagem integrada de experimentos e de modelos para o planejamento de capacidade de operações provisionadas por abordagens denominadas de conteinerização e virtualização. O estudo levou em consideração análises sobre avaliação de desempenho, consumo de energia e custo submetidas a ambientes com variadas configurações de capacidade. Para a condução da pesquisa, foram concebidos estudos casos compostos por vários cenários experimentais que permitiram realizar uma série análises e demonstrações acerca de instâncias baseadas em contêineres e VMs. Os estudos de casos foram separados em duas fases complementares onde a primeira adotou uma arquitetura experimental cujos recursos foram provisionados para implantar ambientes baseados em contêineres e VMs. No estudo de caso I, foram analisados os comportamentos da vazão na rede, da utilização da CPU e do custo do consumo de energia sobre operações implantadas por contêineres do *Docker Engine* e VMs do *Hypervisor KVM*. Nesse contexto, os ambientes instanciados pelas duas tecnologias foram observados através de experimentos executados via ferramentas de medições e coletas. Posteriormente, as amostras derivadas dos experimentos foram analisadas e apresentados considerandos as métricas selecionadas em etapas anteriores.

Mais adiante, um estudo de caso II foi concebido com o intuito de complementar o primeiro, visando consolidar todas as etapas executadas por ambos na metodologia proposta nesta dissertação. Dessa forma, uma nova arquitetura base foi criada com pequenas modificações nas configurações dos recursos e na organização dos componentes para que os ambientes baseados em contêineres e VMs fossem provisionados com implantações da aplicação Moodle. Nesse caso, os ambientes Moodle em operação foram observados através de experimentos executados por um conjunto de requisições com funcionalidades frequentemente utilizadas pelos alunos. Para esse propósito, foram observados os comportamentos da vazão das requisições por segundos, do tempo de resposta, da potência elétrica e dos custos associados à operação. As amostras das requisições coletadas foram

utilizadas como insumos para as etapas de geração e de validação dos modelos SPNs. Os resultados dessas etapas deram consequência na geração de novos cenários com variações nos parâmetros de capacidade não suportados pela infraestrutura real. As análises dos modelos abordaram cenários experimentais acerca da avaliação de desempenho, consumo de energia e custo de operações baseadas em contêineres e VMs.

Considerando o estudo de caso II, as amostras foram coletadas nos experimentos realizados em cinco turmas do *Moodle* com quantidades de alunos variadas. A vazão média de cada turma apresentou um intervalo de confiança com 95% que forneceu limites inferiores e superiores para verificar as conformidades entre as médias obtidas nas análises do modelo. As equivalências entre as médias do sistema real e do modelo foram validadas dentro da margem de erro predefinida. Dessa forma, o modelo apresentou resultados precisos e consistentes em comparação ao desempenho do sistema real. Uma vez validada a corretude do modelo, novos cenários foram analisados com perspectivas de escopos distintos. O primeiro cenário observou o comportamento da vazão de requisições por segundos em relação aos tempos de chegadas com variações de 100 milissegundos até 10 segundos, totalizando 19 escopos experimentais. O cenário II avaliou os gargalos na operação provenientes dos aumentos nas demandas das turmas. O cenário III forneceu estimativas de escopos com instâncias suficientes para dar vazão às demandas das turmas, bem como, reduzir os gargalos causados pelos atrasos no tempo de resposta. Por fim, o cenário IV observou o consumo de energia e o custo relacionado à vazão demandada pelas requisições das turmas. Nessa análise, a potência elétrica demandada foi observada em decorrência da vazão suportada nas operações dos contêineres e VMs.

O custo da energia demandada levou em consideração a tarifa média nacional para realizar projeções custos anuais. Apesar das duas tecnologias coexistirem na maioria das arquiteturas de sistemas, os resultados indicaram que uma provável migração gradual de um ambiente baseado em VMs para um de contêineres pode preservar a infraestrutura existente e ainda teria uma redução considerável no custo do consumo de energia atribuído à operação do *data center*. Um agrupamento com 10 instâncias, ocupado na sua vazão máxima, gerou uma redução de 46,54% na demanda de energia favor dos contêineres. Além disso, a iniciativa poderia contribuir diretamente com a redução da emissão de gases de efeitos estufa. Todavia, é importante destacar que este trabalho não descarta a adoção de ambiente Mooble provisionados exclusivamente por VMs ou híbridos entre as duas

tecnologias, porém sugere a adoção desta metodologia proposta para que a capacidade da operação seja otimizada na perspectiva de desempenho, do consumo de energia e do custo.

8.1 Contribuições

Através dos pontos abordados nesta dissertação, foi possível destacar as seguintes contribuições:

- Proposição de uma metodologia: os passos realizados nesta pesquisa foram mapeados em 09 etapas detalhadas com o propósito de auxiliar os pesquisadores na condução de outras pesquisas.
- Proposição de arquiteturas experimentais: muitas vezes ocorrem indefinições na preparação dos ambientes de experimentação, gerando dificuldades para escolher as tecnologias e ferramentas mais apropriadas para determinadas pesquisas. Essas duas arquiteturas propostas podem servir de referência para outras pesquisas relacionados com a desta dissertação.
- Elaboração de modelos de infraestrutura para ambientes de aplicações AVA Moodle provisionados por contêineres e VMs: os modelos propostos têm como intuito fornecer informações para apoiar na tomada de decisão de quaisquer partes interessadas, como administradores, especialistas, projetistas e gestores. Foram desenvolvidos e validados modelos que permitiram identificar cenários otimizados em relação ao tempo de resposta, à vazão, ao consumo de energia e ao custo.
- Avaliação de ambientes AVA Moodle para diferentes cenários de turmas: por meio dos estudos de casos, foram apresentados cenários com diferentes variações de parâmetros, como tempos de chegadas das requisições, quantidades de instâncias, tempos de serviços e potência elétrica. As análises demonstraram factíveis aplicabilidades dos modelos em cenários concretos de plataformas AVA Moodle.
- Identificação de gargalos em relação à demanda: os gargalos excessivos podem afetar negativamente a qualidade dos serviços entregues e a experiência do usuário. Esse problema pode estar atrelado a ajustes incorretos na capacidade dos ambientes. Os modelos SPNs e validados modelos as partes interessadas no planejamento de capacidade de *clusters* baseados em contêineres ou VMs.
- Definição de limiares de auto-escalonamento: foram analisados cenários que

podem estimar limites mínimos e máximos das instâncias adequadas às políticas de auto-escalonamento. Esse mecanismo leva em consideração parâmetros de desempenho que disparam gatilhos para escalar os contêineres ou VMs. As análises indicaram cenários com quantidades de instâncias necessárias para suportar determinadas cargas de trabalhos atreladas a tempos de chegadas específicos.

Além das contribuições citadas acima, foram escritos dois artigos a partir dos resultados apresentados nesta dissertação, sendo o primeiro já publicado e o segundo aceito com publicação a ser realizada pela revista.

- Cleyton Gonçalves, Ermeson Andrade, Gustavo Callou, Bruno Nogueira. *"Avaliação de Desempenho, Consumo de Energia e Custo para Ambientes Baseados em Contêineres e Máquinas Virtuais"*.
- Cleyton Gonçalves, Ermeson Andrade, Júlio Mendonça, Gustavo Callou. "*Modelos Estocásticos para o Planejamento de Ambientes AVA Moodle Baseados em Contêineres e Máquinas Virtuais"*.

8.2 Limitações

Essa pesquisa apresenta algumas possibilidades de extensão conforme as limitações encontradas abaixo:

- Adoção de outros conjuntos de requisições: o modelo conseguiu representar um conjunto de 13 requisições utilizadas com mais frequência na plataforma. Todavia, o Moodle oferece uma gama de funcionalidades vinculadas a diversos recursos que podem ser objetos de modelagens. Isso permitiria ampliar o leque de análises com a criação de novos cenários e outras variações nos parâmetros do modelo.
- Avaliação de conteúdos multimídia: os *streamings* de áudio e vídeo consomem mais recursos computacionais e também demandam uma parcela superior a 80% do tráfego da Internet. Dentro desse montante estão incluídas as plataformas de aprendizagens, as quais lançam mão dos recursos audiovisuais para auxiliar no processo de aprendizagem. No entanto, a criação de estudos de casos exclusivos para conteúdos de áudio e vídeo tornaria o escopo da pesquisa muito extenso, dado o aumento na curva de aprendizagem do tema e no esforço requerido para executar uma série de etapas da metodologia proposta.

- Adoção de outras aplicações AVA: apesar do Moodle representar o AVA *opensource* com maior popularidade para aprendizagens online, o mercado oferece outras alternativas de AVA com bastante adesão nos meios acadêmicos. Dessa forma, a arquitetura adotada nesta dissertação suportaria a implantação de novos ambientes com dois ou mais AVAs. Nesse caso, as cargas de trabalhos seriam realizadas com base em requisições de funcionalidades comuns a todas aplicações para garantir uma maior imparcialidade nas análises.
- Análise de Dependabilidade: as aplicações distribuídas estão sujeitas a falhas por diversas situações provocadas por agentes externos, erros de configurações, tempo de vida útil dos equipamentos, entre outras. Dessa forma, a análise de Dependabilidade, que envolve um conjunto de técnicas para assegurar a tolerância a falhas dos sistema computacionais, poderia ser aplicado a fim de garantir uma maior confiabilidade e disponibilidade dos serviços providos pelos ambientes adotados.
- Avaliação de ambientes implantados por sistemas de orquestração: o sistema de orquestração permite construir *Clusters Multi Nodes*, onde as instâncias podem ser distribuídas entre dois ou mais nós. Esses sistemas garantem maior resiliência das aplicações em virtude dos *Clusters* suportarem mecanismos de redundância, de balanceamento de carga e de disponibilidade. Nesta pesquisa, apenas um computador hospedeiro foi alocado para criar os ambientes baseados em contêineres e VMs.

8.3 Trabalhos Futuros

Em trabalhos futuros, almejamos avaliar outros ambientes AVAs, como o Claroline ou o TelEduc. Almejamos também avaliar a persistência da aplicação através de sistema de gerenciamento de banco de dados não relacionais, os quais tendem a ser mais aderentes às arquiteturas de microsserviços, na perspectiva de desempenho e escalabilidade. Esses bancos são conhecidos pelo termo NoSQL *(Not only Structure Query Language)*. Outro possível trabalho futuro é criar SPNs para representar o comportamento de falha e reparo dos ambientes baseados em contêineres e VMs. Além disso, pretendemos adotar um ou dois sistemas de orquestração de contêineres para construir ambientes distribuídos em Clusters Multi Nodes.

Referências

AMDOCS. **New Streamer 2021 Report**. 2021. <https://www.amdocs.com/sites/default/files/New-Streamer-2021-Report-08Feb21.pdf>. [Online; acessado em: 23 de outubro de 2021].

ANATEL. **Com maior uso da internet durante pandemia, número de reclamações aumenta**. 2020. <https://g1.globo.com/economia/tecnologia/noticia/2020/06/11/com-maior-uso-da-internet-durante-pandemia-numero-de-reclamacoes-aumenta-especialistas-apontam-problemas-mais-comuns.ghtml>. [Online; acessado em: 23 de Outubro de 2021].

ANDRAE, A.; EDLER, T. On global electricity usage of communication technology: Trends to 2030. **Challenges**, v. 6, p. 117–157, 04 2015.

ANEEL. **Agência Nacional de Energia Elétrica**. 2020. <http://www.aneel.gov.br/ranking-das-tarifas>. [Online; acessado em: 11 de Dezembro de 2020].

APACHE. **Apache HTTP Server Documentation**. 2021. <https://httpd.apache.org/docs-project/>. [Online; acessado em: 25 de Dezembro de 2021].

BALBO, G. Introduction to generalized stochastic petri nets. In: . Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 83–131. ISBN 978-3-540-72482-7.

Barik, R. K.; Lenka, R. K.; Rao, K. R.; Ghose, D. Performance analysis of virtual machines and containers in cloud computing. In: **2016 International Conference on Computing, Communication and Automation (ICCCA)**. [S.l.: s.n.], 2016. p. 1204–1210.

BELKHIR, L.; ELMELIGI, A. Assessing ict global emissions footprint: Trends to 2040 & recommendations. **Journal of Cleaner Production**, v. 177, p. 448–463, 2018.

Bhimani, J.; Yang, Z.; Leeser, M.; Mi, N. Accelerating big data applications using lightweight virtualization framework on enterprise cloud. In: **2017 IEEE High Performance Extreme Computing Conference (HPEC)**. [S.l.: s.n.], 2017. p. 1–7.

BOS, H.; TANENBAUM, A. **Sistemas Operacionais Modernos**. PEARSON BRASIL, 2015. 325-341 p. ISBN 9788543005676. Disponível em: <https://books.google.com.br/books?id=Zu9BtAEACAAJ>.

Brondolin, R.; Sardelli, T.; Santambrogio, M. D. Deep-mon: Dynamic and energy efficient power monitoring for container-based infrastructures. In: **2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)**. [S.l.: s.n.], 2018. p. 676–684.

Chen, F.; Zhou, X.; Shi, C. The container deployment strategy based on stable matching. In: **2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)**. [S.l.: s.n.], 2019. p. 215–221.

CISCO. **Cisco Predicts More IP Traffic in the Next Five Years**. 2021. <https://www.cisco.com/c/dam/m/en_us/network-intelligence/service-provider/digital-transformation/knowledge-network-webinars/pdfs/1213-business-services-ckn.pdf>. [Online; acessado em: 18 de Junho de 2021].

COSTA, L. H. M. K. **O que é virtualização?** 2021. <https://www.gta.ufrj.br/ensino/eel879/trabalhos_v1_2017_2/kvm/>. [Online; acessado em: 09 de Novembro de 2021].

Cuadrado-Cordero, I.; Orgerie, A.; Menaud, J. Comparative experimental analysis of the quality-of-service and energy-efficiency of vms and containers' consolidation for cloud applications. In: **2017 25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)**. [S.l.: s.n.], 2017. p. 1–6. ISSN 1847-358X.

DEHON, P.; SILVA, A.; INOCêNCIO, A.; CASTRO, C.; COSTA, H.; AFONSO, P. Cvchatbot: Um chatbot para o aplicativo facebook messenger integrado ao ava moodle. In: . [S.l.: s.n.], 2018. p. 1623–1632.

FIENI, G.; ROUVOY, R.; SEINTURIER, L. Smartwatts: Self-calibrating software-defined power meter for containers. In: **2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)**. [S.l.: s.n.], 2020. p. 479–488.

GERMAN, R. Performance analysis of communication systems - modelling with non-markovian stochastic petri nets. In: **Wiley-Interscience series in systems and optimization**. [S.l.: s.n.], 2000.

GONCALVES, C.; ANDRADE, E.; CALLOU, G.; NOGUEIRA, B. Avaliação de desempenho, consumo de energia e custo para ambientes baseados em contêineres e máquinas virtuais. **Revista Brasileira de Computação Aplicada**, v. 13, n. 1, p. 11–26, nov. 2020. Disponível em: <http://seer.upf.br/index.php/rbca/article/view/10827>.

GRIMA, P.; ALMARGO, L.; LLABRES, X. **Industrial Statistics with Minitab**. [S.l.: s.n.], 2012. ISBN 978-0-470-9727-5-5.

IFSTAT. **Ifstat Network Monitor**. 2020. <https://linux.die.net/man/1/ifstat>. [Online; acessado em: 02 de Novembro de 2020].

Ivanov, K. **KVM Virtualization Cookbook**. 1st. ed. [S.l.]: Packt Publishing, 2017. 40 p.

JAIN, R. **The art of computer systems performance analysis - techniques for experimental design, measurement, simulation, and modeling.** [S.l.]: Wiley, 1991. I-XXVII, 1-685 p. (Wiley professional computing). ISBN 978-0-471-50336-1.

JMETER. **Apache JMeter**. 2020. <http://jmeter.apache.org>. [Online; acessado em: 02 de Novembro de 2020].

JONES, N. How to stop data centres from gobbling up the world's electricity. **Springer Nature Limited**, v. 561, p. 163–166, September 2018. ISSN 1476-4687. Disponível em: <https://www.nature.com/magazine-assets/d41586-018-06610-y/d41586-018-06610-y.pdf>.

JR, J. C. C. Utilização do ava moodle e suas contribuições no processo de ensino-aprendizagem: um relato de experiência da plataforma em uma disciplina de ciências humanas voltada à saúde. In: . [S.l.: s.n.], 2019. v. 13, p. 6–26.

KENETT, R. S.; ZACKS, S.; AMBERTI, D. Modern industrial statistics: with applications in r, minitab and jmp. In: . [S.l.: s.n.], 2014.

LIMA, C. J. d.; VINICIUS, A. V. G. d. S.; CARNEIRO, E. C. d. A.; CALLOU, G. R. d. A. Performance and power consumption evaluation of the moodle environment. **Revista Eletrônica de Gestão Organizacionalt**, v. 17, n. 5, p. 120–133, May 2019. Disponível em: <https://rsdjournal.org/index.php/rsd/article/view/15191>.

LIN, C.-C.; CHEN, J.-J.; LIU, P.; WU, J.-J. Energy-efficient core allocation and deployment for container-based virtualization. In: **2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)**. [S.l.: s.n.], 2018. p. 93–101. ISBN 978-1-5386-7308-9.

LITTLE, J. D. C. A Proof for the Queuing Formula: L = (lambda) W. **Operations Research**, v. 9, n. 3, p. 383–387, June 1961. Disponível em: <https://ideas.repec.org/a/inm/oropre/v9y1961i3p383-387.html>.

MANN, H. B.; WHITNEY, D. R. On a test of whether one of two random variables is stochastically larger than the other. **Ann. Math. Statist.**, The Institute of Mathematical Statistics, v. 18, n. 1, p. 50–60, 03 1947. Disponível em: <https://doi.org/10.1214/aoms/1177730491>.

MARSAN, M. A.; BALBO, G.; CONTE, G.; DONATELLI, S.; FRANCESCHINIS, G. **Modelling with Generalized Stochastic Petri Nets**. 1st. ed. USA: John Wiley & Sons, Inc., 1994. ISBN 0471930598.

MOODLE. **Moodle statistics**. 2021. <https://moodle.net/stats/>. [Online; acessado em: 31 de Março de 2021].

Mukhedkar, P.; Vettathu, A. **Mastering KVM Virtualization**. 1st. ed. [S.l.]: Packt Publishing, 2016. 01–10 p.

MURATA, T. Petri nets: Properties, analysis and applications. **Proceedings of the IEEE**, v. 77, n. 4, p. 541–580, April 1989.

MYSQL. **MySQL Documentation**. 2021. <https://dev.mysql.com/doc/>. [Online; acessado em: 25 de Dezembro de 2021].

Nickoloff, S. K. J.; Fisher, B. **Docker in Action**. 1st. ed. [S.l.]: Manning books, 2019. 30–40 p.

NMON. **NMON for Linux**. 2020. <http://nmon.sourceforge.net/pmwiki.php>. [Online; acessado em: 02 de Novembro de 2020].

PETRI, C. **Kommunikation mit Automaten**. Rheinisch-Westfälisches Institut f. instrumentelle Mathematik an d. Univ., 1962. (Schriften des Rheinisch-Westfälischen Institutes für Instrumentelle Mathematik an der Universität Bonn). Disponível em: <https://books.google.com.br/books?id=NCZMvAEACAAJ>.

PHP. **PHP Documentation**. 2021. <https://www.php.net/docs.php>. [Online; acessado em: 25 de Dezembro de 2021].

PHUNG, J.; LEE, Y. C.; ZOMAYA, A. Y. Lightweight power monitoring framework for virtualized computing environments. **IEEE Transactions on Computers**, v. 69, n. 1, p. 14–25, 2020.

R. **An Introduction to R**. 2021. <https://cran.r-project.org/doc/manuals/r-release/R-intro.html>. [Online; acessado em: 14 de Novembro de 2021].

RAHO, M.; SPYRIDAKIS, A.; PAOLINO, M.; RAHO, D. Kvm, xen and docker: A performance analysis for arm based nfv and cloud computing. In: **2015 IEEE 3rd Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE)**. [S.l.: s.n.], 2015. p. 1–8.

REDHAT. **O que é virtualização?** 2020. <https://www.redhat.com/pt-br/topics/virtualization/what-is-virtualization>. [Online; acessado em: 23 de Dezembro de 2020].

REISIG, W. Understanding petri nets modeling techniques, analysis methods, case studies. **Bull. EATCS**, v. 112, 2014.

RYU Project Team. **RYU SDN Framework**. 2018. <https://osrg.github.io/ryu-book/en/Ryubook.pdf>. [acessado em: 02-Novembro-2020].

Salah, T.; Zemerly, M. J.; Yeun, C. Y.; Al-Qutayri, M.; Al-Hammadi, Y. Performance comparison between container-based and vm-based services. In: **2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)**. [S.l.: s.n.], 2017. p. 185–190. ISSN 2472-8144.

SALEKHOVA, L. L.; GRIGORIEVA, K. S.; ZINNUROV, T. A. Using lms moodle in teaching clil: A case study. In: **2019 12th International Conference on Developments in eSystems Engineering (DeSE)**. [S.l.: s.n.], 2019. p. 393–395.

SHAPIRO, S. S.; WILK, M. B. An analysis of variance test for normality (complete samples)†. **Biometrika**, v. 52, n. 3-4, p. 591–611, 12 1965. ISSN 0006-3444. Disponível em: <https://doi.org/10.1093/biomet/52.3-4.591>.

SILVA, B.; MACIEL, P.; ZIMMERMANN, A. Performability models for designing disaster tolerant infrastructure-as-a-service cloud computing systems. In: **8th International Conference for Internet Technology and Secured Transactions (ICITST-2013)**. [S.l.: s.n.], 2013. p. 647–652.

SILVA, B.; MATOS, R.; CALLOU, G.; FIGUEIREDO, J.; OLIVEIRA, D.; FERREIRA, J.; DANTAS, J.; JúNIOR, A. L.; ALVES, V.; MACIEL, P. Mercury: An integrated environment for performance and dependability evaluation of general systems. In: . [S.l.: s.n.], 2015.

Tadesse, S. S.; Malandrino, F.; Chiasserini, C. Energy consumption measurements in docker. In: **2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)**. [S.l.: s.n.], 2017. v. 2, p. 272–273. ISSN 0730-3157.

Turnbull, J. **The Docker Boook - Containerization is the new virtualization**. 2016. <https://github.com/eduleboss/the-best-docker-books/blob/master/books/The%20Docker%20Book%20-%20James%20Turnbull%20-%20v17.03.0.pdf>. [Online; acessado em: 07-Maio-2019].

VMWARE. **Virtualização**. 2020. <https://www.vmware.com/br/solutions/virtualization.html>. [Online; acessado em: 24 de Dezembro de 2020].

WATTSUP. **Watts Up Power Meter**. 2020. <https://arcb.csc.ncsu.edu/~mueller/cluster/arc/wattsup/usb/watts-up-meters-manual.pdf>. [Online; acessado em: 02 de Novembro de 2020].

Yadav, R. R.; Sousa, E. T. G.; Callou, G. R. A. Performance comparison between virtual machines and docker containers. **IEEE Latin America Transactions**, v. 16, n. 8, p. 2282–2288, Aug 2018. ISSN 1548-0992.

ZHENG, R.; WANG, H.; JIN, H.; XU, D.; CHEN, Y.; LI, X.; RAO, Y.; ZHANG, Z. Energy saving strategy of power system cluster based on container virtualization. In: **2020 Asia Energy and Electrical Engineering Symposium (AEEES)**. [S.l.: s.n.], 2020. p. 351–355.