Avaliação de Desempenho de Migração ao Vivo de Contêineres com Redes de Petri Estocásticas

Leonel Feitosa [1], **Paulo A. L. Rego**[2] **e Francisco Airton Silva**[1]

[1]Laboratório de Pesquisas Aplicadas a Sistemas Distribuídos (PASID)
Universidade Federal de Piauí – PI – Brasil

[2]Departamento de Computação - Universidade Federal do Ceará – CE – Brasil

{leonel,faps}@ufpi.edu.br paulo@dc.ufc.br

Resumo. *Atualmente, a maioria dos data centers em todo o mundo utiliza microsserviços e contêineres. Contêineres normalmente são executados em um único host de controle, acessando um único kernel. Uma alternativa para mitigar o problema de indisponibilidade e perda de desempenho é migrar contêineres entre hosts. Existem utilitários como o Checkpoint Restoration In Userspace (CRIU) que podem ser usados especificamente para realizar migração de contêineres, porém não é fácil escolher uma política de migração específica considerando as particularidades arquiteturais e do software que está executando de forma distribuída nos contêineres. Este artigo propõe um modelo de rede de Petri estocástica (SPN) para modelar políticas de migração de contêineres (que também pode ser aplicado no contexto de máquinas virtuais - VMs), avaliando o tempo médio de migração de tais elementos. O modelo permite também calcular a probabilidade de cada política finalizar o processo de migração em determinado espaço de tempo. Este é o primeiro trabalho com estas características no contexto de migração de elementos virtualizáveis.*

Abstract. *Currently, most data centers around the world use microservices and containers. Containers typically run on a single controlling host, accessing a single kernel. The more systems are distributed the greater the probability of failure. An alternative to mitigate the problem of unavailability and loss of performance is to migrate containers between hosts. There are utilities like Checkpoint Restoration In Userspace (CRIU) that can be used specifically to perform the migration of containers, however it is not easy to choose a specific migration policy considering the architectural particularities and the software that is running in a distributed way in the containers. This paper proposes a Stochastic Petri Net (SPN) model to model migration policies for containers (and also VMs), evaluating the mean migration time of such elements. The model also allows calculating the probability of each policy completing the migration process in a given period of time. This is the first work with these characteristics in the context of migration of virtualizable elements.*

1. Introdução

Catástrofes naturais podem causar grandes impactos em infraestruturas computacionais que não estão prontas para este tipo de problema. Muitas empresas utilizam sistemas de recuperação de desastres para minimizar o tempo de inatividade incorrido por falhas catastróficas do sistema. Os mecanismos atuais de recuperação de desastres variam de backups periódicos transportados para fora do local, a replicação síncrona contínua de dados entre locais separados geograficamente. Sobre os serviços em execução, uma atividade muito importante é a migração ao vivo de máquinas virtuais, e mais recentemente contêineres. A migração de contêineres é usada por vários motivos, sendo um deles manter a proximidade entre os serviços de computação de borda e usuários móveis [Conforti et al. 2021] ou usados em caso de ocorrência de desastres. Para realizar tal migração, deve-se usar algum utilitário de migração, ou seja, uma ferramenta responsável por mover um elemento migrável de um host para outro host sem grandes perdas de desempenho e de dados.

Para implementar a migração "ao vivo" de contêineres em sistemas operacionais Linux existe o *Checkpoint Restoration In Userspace* (CRIU). O CRIU é uma opção de utilitário usado para migração de contêineres e com uma comunidade de desenvolvimento bastante ativa. Este artigo se concentra em contêineres LXC de código aberto, com o CRIU selecionado como a ferramenta de migração dada a sua alta popularidade. O CRIU congela um aplicativo em execução (ou parte dele) e salva o estado atual do aplicativo em um ou mais arquivos. Os arquivos podem ser restaurados, permitindo que o aplicativo retome a execução do ponto congelado anterior logo antes do ponto de verificação [Maheshwari et al. 2018]. O CRIU apresenta várias políticas para migração de contêiner, incluindo StopandCopy, PreCopy, PostCopy e HybridCopy [Pickartz et al. 2016].

Cada política de migração de contêiner possui suas respectivas regras e etapas de cópia, processamento, etc. Dadas as peculiaridades de cada política, escolher qual política usar torna-se difícil por existirem vários fatores que poderão influenciar no tempo total de migração (MTT). Para minimizar o MTT, o avaliador deve considerar não só o tamanho do elemento a ser migrado, mas também a quantidade de elementos migráveis e capacidade paralela do sistema de migração. Balancear estas variáveis é uma tarefa complexa onde fazer experimentos reais pode demandar grandiosos esforços de tempo e recurso. Modelos analíticos (como redes de Petri) podem mitigar este problema realizando predições do desempenho da migração considerando os fatores mais relevantes no processo de migração.

Nos últimos anos, a comunidade científica tem se debruçado sobre a de migração de contêineres. No entanto, existem poucos trabalhos com experimentos que abranjam vários fatores que possam influenciar de fato nessa atividade. Em uma pesquisa prévia na literatura, foram encontrados alguns trabalhos que avaliaram o desempenho de migração de contêineres executando experimentos reais [Puliafito et al. 2019, Qiu et al. 2019, Maheshwari et al. 2018, Junior et al. 2020, Conforti et al. 2021, Ramanathan et al. 2021]. Porém, nenhum de tais trabalhos explorou predição de desempenho com uso de modelagem analítica. Trabalhos com esta natureza de mensuração muitas vezes são limitados em generalizar certas conclusões do estudo.

Nesse contexto, este artigo propõe um modelo de redes de Petri estocástica (SPN) para representar e calcular o MTT e a probabilidade de término de uma migração dado

um determinado tempo. O presente estudo validou o modelo com dados de experimentos reais e oferece um estudo de caso que serve como guia de utilização do modelo. Dessa forma, administradores de sistemas virtualizados poderão melhor planejar o processo de migração mesmo em estágios iniciais de design da arquitetura computacional a ser implantada. Em suma, as principais contribuições desse artigo são:

- Um modelo SPN que representa e avalia distintas políticas de migração de elementos virtualizáveis (contêineres e VMs) considerando principalmente parâmetros de quantidade de elementos migráveis e capacidade de migração paralela.
- Possibilitar avaliadores de sistemas virtualizados a se planejar para automatizar políticas de migração com os cálculos prévios do modelo proposto visando minimizar períodos de inatividade do sistema.
- Possibilitar a previsão da probabilidade de cada política de migração finalizar a execução em determinado espaço de tempo.

O restante deste artigo está organizado da seguinte forma: a Seção 2 descreve alguns conceitos necessários ao entendimento da proposta. A Seção 3 apresenta os trabalhos relacionados, comparando-os com nossa proposta. Na Seção 4 apresentamos o modelo SPN para modelar o sistema de migração. A Seção 5 mostra a validação do modelo. A Seção 6 apresenta os resultados obtidos no estudo de caso. Por fim, na Seção 7 encontra-se a conclusão e trabalhos futuros.

2. Referencial Teórico

Esta seção apresenta brevemente alguns conceitos necessários para o entendimento da proposta. Por limitação de páginas, não apresentamos conceitos de redes de Petri, mas o leitor pode consultar em [Cardoso and Valette 1997]. Primeiramente, focamos nas métricas que podem ser computadas pelo modelo proposto, e em seguida apresentamos as políticas de migração estudadas.

2.1. Redes de Petri Estocásticas

A grande contribuição do uso de modelos neste trabalho é quando um avaliador quer saber qual a infraestrutura necessária para atender uma determinada demanda. Isso permite o avaliador se planejar para adquirir uma infraestrutura adequada para seus interesses. O modelo permite, assim, evitar gastos desnecessários e a surpresa de ter recursos insuficientes.

A Figura 1 exibe os componentes básicos usados para modelar uma Rede de Petri Estocástica (SPN). Os lugares são representados por círculos, enquanto as transições são representadas como retângulos preenchidos (transições imediatas) ou retângulos ocos (transições temporizadas). Arcos (setas direcionadas) conectam lugares a transições e vice-versa. Tokens (pequenos círculos preenchidos) podem residir em locais que denotam o estado (ou seja, marcação) de uma SPN. O comportamento de uma SPN é definido em termos de um fluxo de tokens, o que significa que os tokens são criados e destruídos conforme os disparos de transições.

As transições imediatas representam atividades instantâneas e têm maior prioridade de disparo do que as transições temporizada. Essas transições também podem conter uma condição de guarda e um usuário pode especificar uma prioridade de disparo diferente entre outras transições imediatas. Há também expressões de guarda em

SPNs. As expressões de guarda são expressões booleanas que controlam o disparo de uma transição, declarando alguma condição referente à marcação da rede. Se a expressão de guarda de uma transição produzir um valor verdadeiro, ela consegue disparar, caso contrário, a transição é desabilitada.

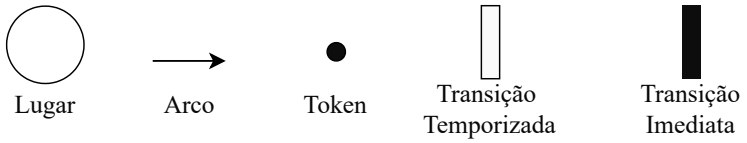

Figura 1. Componentes principais de uma SPN.

2.2. Métrica - Migration Total Time (MTT)

O Migration Total Time (MTT) é o tempo médio para terminar a migração de todos os elementos a serem migrados. MTT é o tempo esperado para chegar a um impasse de marcação. Em teoria de redes de Petri e cadeias de Markov, chamamos esse impasse de , quando um token não possui outro lugar para ser consumido. O estado absorvente pode ser calculado gerando uma Continuous-time Markov Chain (CTMC) associada ou por análise numérica [Nelson 2013]. No presente trabalho, utilizamos análise transiente, que a partir de uma SPN gera uma CTMC e faz o cálculo do estado absorvente. Usando a Figura 2 como exemplo, o MTT é baseado em um conjunto de probabilidades para quando um token sai do lugar START para o FINISH. Nesse exemplo, a migração é dividida em dois tempos divididos nas transições T1 e T2. A migração de K elementos pode ser feita em paralelo dependendo da quantidade de agentes de migração disponíveis (marcação C) [German 2000, Marsan et al. 1998].

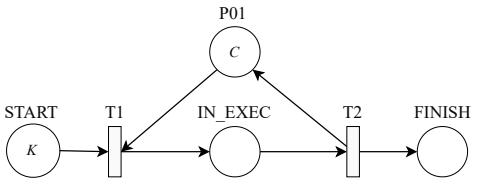

Figura 2. Exemplo de Modelo SPN.

O modelo proposto neste trabalho também calcula a, amplamente utilizada, função de distribuição cumulativa (CDF). Com a CDF é possível estimar a probabilidade (entre 0 e 1) de terminar a execução antes de um tempo específico $[P(T < t)]$ e a probabilidade de terminar a execução em um intervalo de tempo $[P(t1 < T < t2) = P(T < t2) - P(T \leq t1)]$.

2.3. Políticas de Migração Exploradas

Esta seção descreve as políticas de migração exploradas. Plataformas para gerenciar cargas de trabalho e serviços conteinerizados, como Kubernetes, podem utilizar o runC [1] para gerar e executar contêineres. runC é um software de execução de contêiner leve e portátil, iniciado pela Open Container Initiative (OCI) e utilizado por muitas soluções de virtualização baseadas em contêiner, como Docker e Containerd. Para implementar a migração ao vivo de contêiner em sistemas operacionais Linux, o runC recorre ao Checkpoint Restoration In Userspace (CRIU), que é um componente de software que oferece funcionalidade de pontos de verificação/restauração para aplicativos Linux [2]. O CRIU

[1] runC: `https://www.docker.com/blog/runc`
[2] CRIU: `https://criu.org/Main_Page`

apresenta vários métodos para migração de contêiner, incluindo StopandCopy, PreCopy, PostCopy e HybridCopy. Neste artigo modelamos dois destes métodos com redes de Petri estocásticas, o StopandCopy (Figura 3) e PreCopy (Figura 4). Também modelamos o método de migração tradicional de máquinas virtuais - VM PreCopy (Figura 5).

A Figura 3 mostra as diferentes fases da migração do contêiner com base no método StopandCopy: pontos de verificação, transferência e restauração. Durante o checkpoint, o CRIU congela o contêiner em execução no nó de origem (nó A) e coleta metadados sobre o estado da CPU, conteúdo da memória e informações sobre a árvore de processos associada ao serviço de contêiner em execução. Durante a fase de transferência, as informações de metadados coletadas são transferidas para o nó de destino (nó B). A fase de restauração retoma o serviço de contêiner do ponto congelado com os metadados transferidos no nó de destino. O tempo de migração do contêiner pode ser expresso como:

$$T_{cm} = T_c + T_t + T_r \tag{1}$$

onde:

- T_{cm} é o tempo total de migração do contêiner,
- T_c é o tempo do ponto de verificação,
- T_t é o tempo de transferência de metadados da origem para o nó de destino e
- T_r é o tempo de restauração.

Para o método StopandCopy, T_{cm} também corresponde ao tempo de inatividade do serviço porque o serviço de contêiner em execução está interrompido no início da fase de checkpoint, e o serviço é restaurado somente após a restauração bem sucedida.

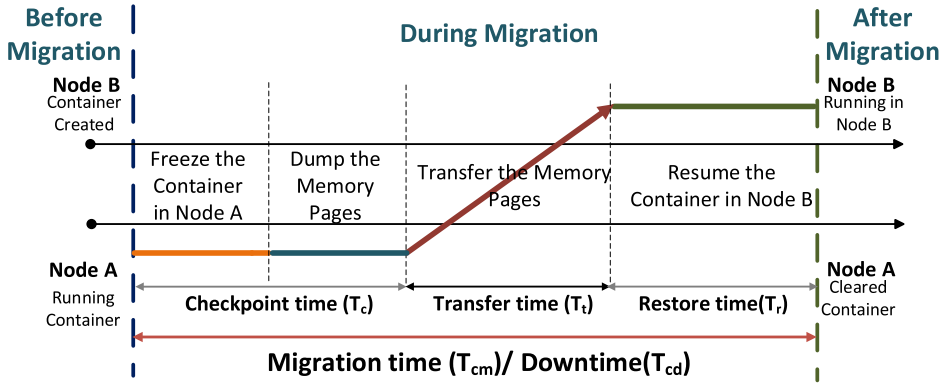

Figura 3. Método de migração Container StopandCopy [Ramanathan et al. 2021].

O método PreCopy mostrado na Figura 4 é outra abordagem migratória de contêiner, consistindo em fases de pre-dump, dump, transferência de dados pré-dump/dump e restauração. O ponto de verificação do aplicativo no nó de origem (nó A) é dividido em fases de pré-dump e dump. A fase de pré-dump coleta todas as informações de estado e memória do contêiner, e a fase de dump coleta apenas as informações das páginas de memória modificadas. Na fase de transferência, os metadados coletados são transferidos para o nó de destino (nó B), e na fase de restauração o serviço de aplicativo é restaurado.

O tempo de migração do contêiner na abordagem PreCopy pode ser expresso como:

$$T_{pcm} = T_{pd} + T_{pdt} + T_d + T_{dt} + T_r \tag{2}$$

onde:

- T_{pcm} é a migração de contêiner total baseada em PreCopy,
- T_{pd} é o tempo de pré-dump,
- T_{pdt} é o tempo de transferência de dados pré-dump,
- T_d é o tempo de dump, e
- T_{dt} é o tempo de transferência de dados de dump.

A vantagem do método PreCopy é que o serviço permanece responsivo durante a fase de pré-dump de coleta do ID do processo, das páginas de memória e do estado de execução.

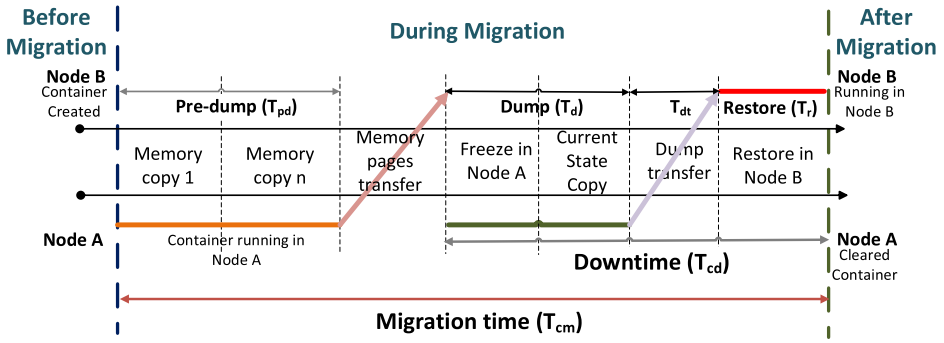

Figura 4. Método de migração Container PreCopy [Ramanathan et al. 2021].

Apesar deste trabalho focar em contêineres, também modelamos uma migração de máquinas virtuais para mostrar que o modelo é adaptável para este tipo de elemento. Assim, o último método de migração modelado é o VM PreCopy (Figura 5). Durante a migração da VM, o estado da CPU, o estado da memória, as interfaces de rede e a imagem do disco de toda a VM são migrados da origem para o nó de destino. Durante o processo de cópia de páginas de memória, as páginas sujas (ou seja, páginas de memória modificadas) são transferidas iterativamente (referidas como fase push) enquanto a VM ainda está em execução no nó de origem. Quando a contagem máxima de iterações é atingida, a VM é temporariamente interrompida no nó de origem, todas as páginas da memória principal são copiadas para o destino e, finalmente, a VM é retomada no nó de destino. Este processo de cópia de página de memória é referido como método PreCopy.

3. Trabalhos Relacionados

Alguns trabalhos focaram em avaliação de desempenho de migração de contêineres [Puliafito et al. 2019, Qiu et al. 2019, Maheshwari et al. 2018, Junior et al. 2020, Conforti et al. 2021, Ramanathan et al. 2021]. [Puliafito et al. 2019] executaram uma extensa avaliação de desempenho com técnicas de migração com computação em névoa.

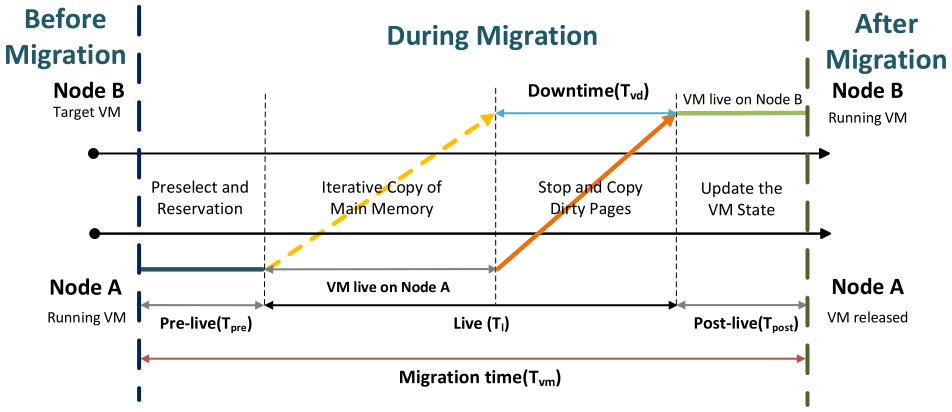

Figura 5. Método de migração VM PreCopy [Ramanathan et al. 2021].

Os resultados obtidos lançam luz sobre a migração de contêineres em ambientes de computação em névoa, esclarecendo, em geral, qual técnica de migração pode ser a mais apropriada sob determinadas condições de rede e serviço. [Qiu et al. 2019] adotaram contêineres LXC que são usados como plataformas Cloudlet e utiliza CRIU (Checkpoint/Restore in the userspace) para migração de contêiner entre Cloudlets. Os resultados demonstraram que a abordagem proposta pode aumentar o desempenho e a confiabilidade do processo de migração em comparação com o uso de máquinas virtuais (VMs).

Ainda sobre contêineres, [Maheshwari et al. 2018] propuseram uma abordagem de migração de contêiner com reconhecimento de tráfego com uma implementação de sistema de ponta a ponta usando o hipervisor LXD (Linux Container Hypervisor). A avaliação do sistema considera as principais métricas associadas à qualidade da experiência (QoE) do aplicativo e à eficiência da rede, como o tempo médio de resposta do sistema e o custo de migração para diferentes combinações de carga, recursos de computação, largura de banda entre bordas da nuvem, rede e latência do usuário. [Junior et al. 2020] Propuseram explorar a estrutura em camadas fornecida pelo sistema de arquivos OverlayFS para obter uma visão geral dos volumes e transferi-los antes da migração real do contêiner. Eles implementaram esse mecanismo dentro do Kubernetes e fizeram avaliações com base em um banco de teste de computação de névoa real mostrando que as técnicas reduziram o tempo de inatividade do contêiner durante a migração por um fator 4 em comparação com uma linha de base sem ponto de verificação de volume. [Conforti et al. 2021] estenderam uma plataforma chamada QUIC para oferecer suporte à migração de conexão do lado do servidor quando um contêiner é migrado entre hosts. Eles avaliaram a solução testando as estratégias propostas usando diferentes técnicas de migração de contêiner e em um cenário sem migração. [Ramanathan et al. 2021] propuseram um framework que permite a migração de componentes EPC virtuais conteinerizados usando uma solução de migração de código aberto que ainda não suporta totalmente a pilha de protocolos de rede móvel. [Ramanathan et al. 2021] também fizeram uma análise abrangente baseada em experimentos de migração ao vivo em duas tecnologias de virtualização — VM e contêiner — com exame adicional sobre a abordagem de migração de contêiner. Nosso trabalho se

baseou nos seus experimentos e arquitetura para confeccionar o modelo SPN proposto.

Estes trabalhos descritos acima abordaram desempenho de migração de contêineres executando experimentos reais. Dentre os trabalhos elencados, nenhum deles utilizou modelagem analítica para predição de desempenho. Nosso trabalho permite predizer o comportamento da migração, algo que a mensuração não permite. O modelo SPN proposto neste trabalho compara três estratégias de migração de contêineres e uma de máquinas virtuais. Tal comparação é explorada variando o número de elementos a serem migrados e variando a capacidade de migração paralela do sistema. O modelo permitiu também observar o resultado da CDF, que permite determinar a probabilidade de toda a migração finalizar em determinada janela de tempo.

4. Modelo SPN

Este trabalho adotou a estratégia de modelagem ao invés de mensuração. Ao contrário do uso de modelagem, a técnica de mensuração é atrelada totalmente ao ambiente de execução e aos parâmetros adotados. Por exemplo, caso o avaliador execute a migração de dois elementos, não é possível com este experimento predizer como será o comportamento da migração de cem elementos paralelos sem ter uma infraestrutura compatível para executar tão atividade. Por outro lado, a modelagem permite isso.

Portanto, esta seção apresenta um modelo SPN para representar e computar características de desempenho de uma rede com diferentes estratégias de migração de contêiner e máquinas virtuais, conforme os diagramas de fluxo apresentados na seção anterior. O objetivo do modelo proposto é auxiliar administradores de sistemas baseados nessa arquitetura na tarefa complexa de ajustar vários parâmetros adequadamente para alcançar níveis de desempenho desejáveis. Logo, o modelo deve ser útil para checar o efeito de mudanças no sistema, antes mesmo que elas sejam implementadas.

A Figura 6 mostra o modelo SPN proposto. Nas seções anteriores, já foi explicado o funcionamento de cada política de migração, bem como apresentado o detalhamento de cada componente e suas funcionalidades. Escolhemos as mesmas políticas utilizadas no artigo base ([Ramanathan et al. 2021]), que possuía dados detalhados de experimentos que adotamos como entrada. Assim, esta seção destaca o funcionamento do modelo de forma geral. Para o processo de migração, os tokens representam uma VM ou contêiner a ser migrado. Existe uma quantidade de elementos migráveis correspondentes à marcação NME no lugar Node_A. Assim, os tokens se encontram inicialmente no Node_A e serão movidos para o Node_B. A migração somente ocorrerá caso tenha capacidade de migração. Tal capacidade é dada pela quantidade de agentes de migração paralela, dado pela marcação PMC no lugar P11. Esses agentes de migração podem ser por exemplo threads do utilitário CRIU que são executadas em um terceiro host somente responsável por executar a migração. O lugar Node_B é do tipo absorvente, como explicado na seção anterior. Quando os tokens chegam ao destino Node_B, a migração está finalizada. O cálculo é feito a partir da probabilidade de todos os tokens presentes no Node_A chegarem ao Node_B. A métrica CDF é calculada com base no lugar Node_B.

A escolha de qual caminho o token vai seguir se dá por condições de guarda presentes nas seguintes transições: Checkpoint_T, Pre-Dump_T e Pre-live. As transições são ativadas conforme a variação da variável POLICY. Se a variável POLICY for igual a 1, ela seguirá pela política de migração Container-StopandCopy. Se for a variável POLICY for

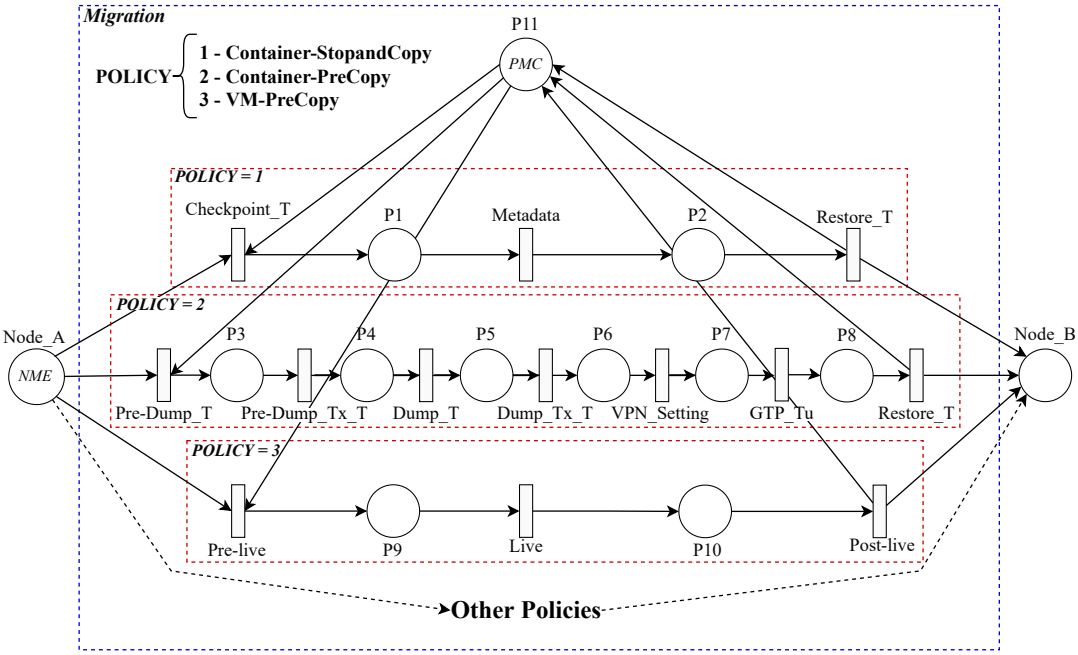

Figura 6. Modelo SPN para calcular o tempo total de migração por diferentes políticas de migração.

igual 2, o token seguirá pela política Conteiner-PreCopy. Por fim, se POLICY for igual a 3, ela irá pelo caminho referente a política VM-PreCopy. Vale ressaltar que pode existir N tipos de políticas de migração. Utilizamos apenas três políticas para o modelo proposto pois havíamos dados reais de experimentação de apenas tais políticas. Todas as transições possuem distribuição de probabilidade exponencial que foi condizente com o experimento real de validação. Todas as transições possuem semântica *infinite server*, indicando que as transições podem ser disparadas com múltiplos tokens em paralelo [Marsan et al. 1998]. A Tabela 1 apresenta os principais elementos do modelo.

5. Validação do Modelo SPN

Para utilizar um modelo analítico (e.g., PN, CTMC, redes de filas, etc.) é importante validar tal modelo. A validação se trata de comparar os resultados gerados pelo modelo com os resultados advindos de uma simulação ou mensuração real [Jain 1990]. Esta seção mostra uma comparação dos resultados de um experimento real com os resultados obtidos com o modelo SPN proposto. Os dados do experimento foram extraídos de [Ramanathan et al. 2021], em que os autores testaram as três políticas detalhadas na Seção 2 (Container StopandCopy, Container PreCopy e VM PreCopy). O trabalho usou EPC (Evolved Packet Core) as a Service (EPCaaS), para o qual a migração foi testada. Três componentes de rede denominados foram migrados dentro de contêineres ou VMs, são eles: *Home Subscriber Server* (HSS), *Mobility Management Entity* (MME), e *Serving and Packet Gateway* (SPGW). Portanto, no trabalho de [Ramanathan et al. 2021], cada migração de contêineres ou VMs tinha implantado um desses três componentes de rede.

A Tabela 2 mostra os parâmetros de entrada utilizados para alimentar o modelo advindos do trabalho de [Ramanathan et al. 2021]. Assim, a Tabela 3 apresenta o resultado da validação. O modelo SPN pode ser solucionado de duas formas, por análise esta-

Tabela 1. Descrição dos elementos principais do modelo.

Tipo	Elemento	Descrição
Lugares	Node_A	Local onde se encontra o elemento de migração a ser migrado
	Node_B	Local para onde será migrado o elemento de migração
	P11	Capacidade associada ao processamento de uma migração
Transições Temporizadas	Checkpoint_T	Tempo de criação do ponto de restauração
	Metadata	Tempo para transferir os metadados do elemento a ser migrado
	Restore_T	Tempo para a restauração
	Pre-Dump_T	Tempo de verificação do aplicativo no nó de origem
	Pre-Dump_Tx_T	Tempo de transferência da verificação do aplicativo no nó de origem
	Dump_T	Tempo da coleta de informações do estado da memória
	Dump_Tx_T	Tempo de transferência das informações do estado da memória
	VPN_Setting	Tempo para configurar a conectividade OpenVPN
	GTP_Tu	Tempo para atualização do túnel GTP que se aplica apenas na política SPGW
	Restore T	Tempo para restauração
	Pre-live	Tempo para preparar a migração
	Live	Tempo para migrar
	Post-live	Tempo associado após a migração
Marcações dos Lugares	NME	Quantidade associada aos elementos a serem migrados
	PMC	Capacidade do agente de migração

cionária ou simulação estacionária. O modelo gerou os resultados por meio de simulação, condicionados a uma margem de erro de 0.1. O intervalo de confiança de todos os resultados do experimento está dentro do intervalo de confiança dos resultados do modelo. Pode-se afirmar, portanto, com 95% de confiança que há evidências que o modelo representa minimamente o sistema real de migração.

Tabela 2. Parâmetros de configuração de tempo usados no estudo de caso [Ramanathan et al. 2021].

Tipo	Transição	Aplicação		
		HSS	MME	SPGW
Tempo	Checkpoint_T	5.98	2.19	3.581
	Metadata	2.00	1.16	1.61
	Restore_T	7.03	2.39	3.65
	Pre-Dump_T	0.32	0.173	0.237
	Pre-Dump_Tx_T	2.35	2.053	1.723
	Dump_T	0.215	0.224	0.183
	Dump_Tx_T	0.549	0.504	0.557
	VPN_Setting	0.28	0.491	0.581
	GTP_Tu	NA	NA	0.449
	Restore T	0.879	0.641	0.735
	Pre-live	2.9	2.83	2.97
	Live	10.0	10.0	10.14
	Post-live	4.0	4.0	4,.0

6. Estudo de Caso

Esta seção apresenta uma análise numérica com uso do modelo SPN proposto. O objetivo desta seção é mostrar a amplitude de utilização do modelo. Focamos em dois parâmetros em particular: a quantidade de elementos a serem migrados e a capacidade do agente de migração. O estudo de caso é útil tanto para obter novas descobertas sobre o funcionamento do modelo e consequentemente do sistema real, como também ilustrar como

Tabela 3. Comparação dos resultados da mensuração com o modelo proposto.

Conteiner StopandCopy				
	Experimento		***Modelo***	
Aplicação	***Mean***	***CI 95%***	***Mean***	***CI 95%***
HSS	15.01	[14.58 - 15.44]	15.16	[14.3 - 15.9]
MME	5.74	[5.56 - 5.92]	5.75	[5.0 - 6.4]
SPGW	8.85	[8.43 - 9.27]	8.62	[7.6 - 9.5]
Container PreCopy				
	Experimento		***Modelo***	
Aplicação	***Mean***	***CI 95%***	***Mean***	***CI 95%***
HSS	4.84	[4.58 - 5.1]	4.81	[4.3 - 5.2]
MME	3.09	[2.96 - 3.22]	3.09	[2.8 - 3.3]
SPGW	4.47	[4.34 - 4.6]	4.52	[4.1 - 4.9]
VM PreCopy				
	Experimento		***Modelo***	
Aplicação	***Mean***	***CI 95%***	***Mean***	***CI 95%***
HSS	16.9	[16.5 - 17.3]	16.86	[16.1 - 17.5]
MME	16.83	[16.55 - 17.11]	16.75	[14.6 - 18.8]
SPGW	17.11	[16.48 - 17.74]	17.13	[14.9 - 19.3]

o modelo pode ser explorado. Os tempos utilizados nas transições do modelo foram os mesmos utilizadas para a aplicação SPGW [Ramanathan et al. 2021]. A representação do modelo e a computação dos resultados da análise numérica foram obtidos com a ferramenta Mercury [Maciel et al. 2017]. Nos estudos de caso, utilizamos a mesma quantidade de políticas de migração representadas no modelo apresentado na Seção 4. Os parâmetros foram alimentados baseados em tempos colhidos no processo de validação, que estão presentes na Tabela 2.

A Figura 7 apresenta o resultado para o MTT em função da variação da quantidade de elementos a serem migrados. Foram testadas as seguintes quantidades de elementos migráveis: NME = [10, 15, 20, 25, 50, 100]. Ao contrário da validação, aqui foi configurado o uso de dois agentes de migração paralela (marcação PMC = 2). O tempo de migração aumenta proporcionalmente à quantidade de elementos migrados pois há um limite paralelo de migrações igual a 2. A política VM-PreCopy teve o maior de todos MTT pois como foi mencionado anteriormente, VMs são mais sofisticadas e pesadas para serem migradas do que contêineres. Conteiner-PreCopy teve um MTT menor do que Conteiner-StopandCopy. Interessante que apesar de Conteiner-PreCopy possuir mais etapas que obviamente o tempo de cada transição também influencia o resultado, no fim, ela possui um desempenho superior considerando o MTT. Comparando as três políticas vemos que com menos elementos a serem migrados, os MTTs são semelhantes e à medida que a quantidade aumenta, estas linhas se distanciam. A política que possui o pior desempenho, tem um ângulo de piora maior do que os demais. Por exemplo, comparando o intervalo de NME=[50-100], temos que o Container-PreCopy possui um aumento de MTT=150s enquanto que o VM-PreCopy possui um aumento de MTT=450s. Portanto, para grandes quantidades de elementos a serem migrados a escolha da política se torna cada vez mais criteriosa pois isso impacta de forma expressiva no valor final de MTT.

A Figura 8 apresenta o MTT em função do aumento da capacidade de migrações paralelas (PMC). O PMC foi variado com os seguintes valores: 1, 2, 3, 4, 5, 6, 7, 8, 9 e 10. A análise foi feita considerando 10 elementos a serem migrados (NME=10). Novamente, foram comparadas as três políticas. Durante o começo da análise, as três políticas de migração começam com um tempo consideravelmente alto, onde a VM-PreCopy tem o pior desempenho. Desde o início da variação de PMC até o final, o tempo MTT segue di-

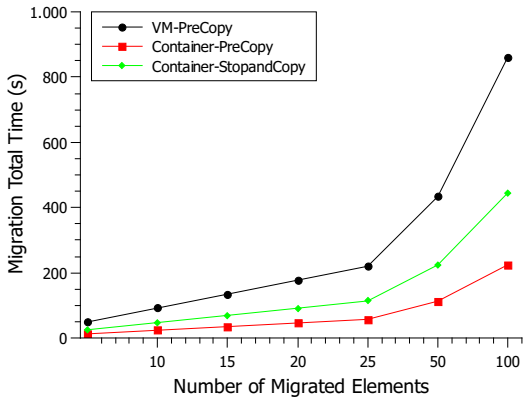

Figura 7. MTT em função do número de elementos a serem migrados (NME).

minuindo até se estabilizar. A estabilização acontece para as três políticas para PMC >= 5. Isso implica que há uma baixa alteração a partir desse ponto, pois o MTT permanece o mesmo. Apesar das linhas se aproximarem com o aumento do PMC, observa-se que elas não se tocam (pelo menos não até este valor de PMC=10). No gráfico anterior, para a variação do NME, no início temos que o MTT para as duas estratégias de contêineres é praticamente igual. Aqui, para a variação de PMC, essa igualdade nunca existe. Portanto, o impacto da variação de PMC foi maior do que o impacto da variação de NME. A principal contribuição deste estudo de caso é mostrar que o avaliador pode ter uma grande economia de recursos conhecendo o limiar de PMC que fará com que não melhore mais o desempenho mesmo o aumentando.

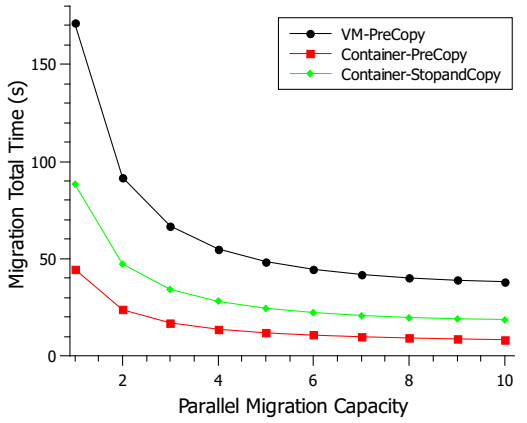

Figura 8. MTT em função da capacidade de migrações paralelas (PMC).

A Figura 9 apresenta a CDF, que permite determinar a probabilidade de toda a migração finalizar em determinada janela de tempo. Essa janela de tempo refere-se ao MTT. A análise (do tipo transiente) foi feita considerando 10 elementos a serem migrados (NME=10) e com capacidade paralela de migração igual a 2 (PMC=2). A janela total de MTT variou entre 0 e 150s. No geral, a probabilidade de finalização da migração é maior para Container-PreCopy, que é maior do que Container-StopandCopy, que é maior do que VM-PreCopy. A partir daqui iremos considerar a notação P[política] para denotar a probabilidade de uma política específica. P[Container-PreCopy] = P[VM-PreCopy] = 0 para os MTTs iguais a 25s e 65s, respectivamente. Um ponto interessante é analisar o

MTT = 50s, onde temos que P[VM-PreCopy] = 0%, P[Container-StopandCopy] = 65% e P[Container-PreCopy] = 100%. Consideremos para este exemplo que MTT = 50s é um requisito importante de Acordo de Nível de Serviço a ser atendido. Se o analista não quiser assumir riscos, obviamente irá adotar a política Container-PreCopy, porém, se este requisito não for tão crítico, ele pode usar o Container-StopandCopy assumindo o risco que há apenas 65% de chance de a migração terminar nesse requerido tempo. Finalmente, vale ressaltar que no tempo MTT=125s, a probabilidade de término da migração é de 100%. Novamente, se o requisito demanda que se tenha um MTT igual a 125s, qualquer uma das três políticas irá atendê-lo.

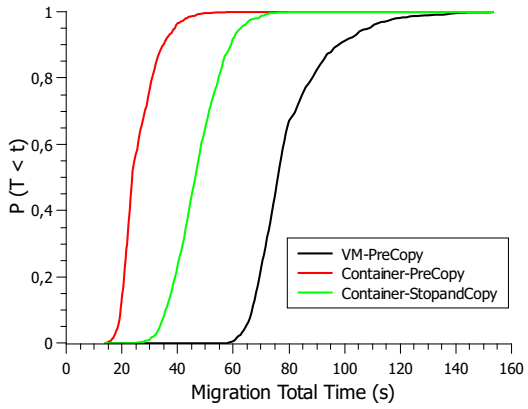

Figura 9. CDF considerando o tempo total de migração.

7. Conclusão

Este artigo apresentou um modelo SPN para comparar políticas de migração de elementos virtualizáveis (contêineres ou VMs). O modelo calcula a métrica MTT, e explora a variação do número de elementos a serem migrados além da capacidade de migração paralela do sistema. O modelo permitiu observar também a probabilidade de toda a migração finalizar em determinada janela de tempo. O modelo poderá auxiliar analistas de sistemas na tarefa de prever o tempo total de migração em diversos cenários. Essa previsão muitas vezes só poderá ser feita com o modelo aqui proposto pois pode ocorrer do avaliador não possuir a respectiva infraestrutura para testes reais. Neste trabalho, afirmamos que a política de Container-PreCopy é a mais performática considerando o tempo total de migração. Não entramos no mérito de analisar outras questões. O avaliador pode, por exemplo, considerar também se fará duas etapas de dump (Container-PreCopy) ou apenas uma (StopandCopy). Uma limitação deste artigo é que não testamos outras políticas, mas o modelo é facilmente adaptável para incluí-las. Trabalhos futuros incluem calcular gasto energético demandado na migração bem como o custo monetário de infraestrutura a ser utilizada no processo.

Referências

Cardoso, J. and Valette, R. (1997). *Redes de petri*. Editora da UFSC Florianópolis.

Conforti, L., Virdis, A., Puliafito, C., and Mingozzi, E. (2021). Extending the quic protocol to support live container migration at the edge. In *2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 61–70. IEEE.

German, R. (2000). *Performance analysis of communication systems with non-Markovian stochastic Petri nets*. John Wiley & Sons, Inc.

Jain, R. (1990). *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. john wiley & sons.

Junior, P. S., Miorandi, D., and Pierre, G. (2020). Stateful container migration in geo-distributed environments. In *2020 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 49–56. IEEE.

Maciel, P., Matos, R., Silva, B., Figueiredo, J., Oliveira, D., Fé, I., Maciel, R., and Dantas, J. (2017). Mercury: Performance and dependability evaluation of systems with exponential, expolynomial, and general distributions. In *2017 IEEE 22nd Pacific Rim international symposium on dependable computing (PRDC)*, pages 50–57. IEEE.

Maheshwari, S., Choudhury, S., Seskar, I., and Raychaudhuri, D. (2018). Traffic-aware dynamic container migration for real-time support in mobile edge clouds. In *2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, pages 1–6. IEEE.

Marsan, M. A., Balbo, G., Conte, G., Donatelli, S., and Franceschinis, G. (1998). Modelling with generalized stochastic petri nets. *ACM SIGMETRICS performance evaluation review*, 26(2):2.

Nelson, R. (2013). *Probability, stochastic processes, and queueing theory: the mathematics of computer performance modeling*. Springer Science & Business Media.

Pickartz, S., Eiling, N., Lankes, S., Razik, L., and Monti, A. (2016). Migrating linux containers using criu. In *High Performance Computing: ISC High Performance 2016 International Workshops, ExaComm, E-MuCoCoS, HPC-IODC, IXPUG, IWOPH, P^ 3MA, VHPC, WOPSSS, Frankfurt, Germany, June 19–23, 2016, Revised Selected Papers 31*, pages 674–684. Springer.

Puliafito, C., Vallati, C., Mingozzi, E., Merlino, G., Longo, F., and Puliafito, A. (2019). Container migration in the fog: A performance evaluation. *Sensors*, 19(7):1488.

Qiu, Y., Lung, C.-H., Ajila, S., and Srivastava, P. (2019). Experimental evaluation of lxc container migration for cloudlets using multipath tcp. *Computer Networks*, 164:106900.

Ramanathan, S., Kondepu, K., Razo, M., Tacca, M., Valcarenghi, L., and Fumagalli, A. (2021). Live migration of virtual machine and container based mobile core network components: A comprehensive study. *IEEE Access*, 9:105082–105100.