Análise de técnicas de otimização multiobjetivo para o posicionamento de controladores em redes SDN

Ana Carolina de Oliveira Christófaro, Marcelo Menezes de Carvalho, Daniel Guerreiro e Silva

***Resumo*— O conceito de Redes Definidas por *Software* apresenta uma solução de engenharia de tráfego de crescente interesse na indústria, levantando questões associadas, por exemplo, à utilização de múltiplos controladores. Considerando que o posicionamento de controladores dentro de uma dada topologia constitui um problema de otimização multiobjectivo de forma a satisfazer certas medidas de desempenho, este trabalho apresenta um estudo comparativo entre o PSA e NSGA-II, com base no estudo inicial desenvolvido por Lange *et al.* [1]. Os resultados mostram que ambas as técnicas são capazes de prover uma boa relação entre precisão e tempo de processamento, resultando em menor utilização de recursos computacionais, com vantagem para o NSGA-II em termos de convergência e grau de exploração do espaço de busca. Tais características são desejáveis para otimização de tráfego dinâmico em tempo real e para garantia da operação confiável e resiliente da rede.**

Palavras-Chave*— *SDN, NSGA-II, PSA.

***Abstract*— Software Defined Network is a traffic engineering solution with an increasing interest by industry, which raises issues such as the use of multiple controllers. Considering that the positioning of controllers within a given topogoly constitutes a multiobjective optimization problem in order to satisfy certain performance metrics, this work presents a comparative analysis between PSA and NSGA-II, based on the initial study developed by Lange *et al.* [1]. The results yield good results for both techniques, in terms of accuracy and processing time, but with an advantage to NSGA-II algorithm, in terms of convergence and exploration degree of the search space. Such properties are interesting when one considers real time traffic optimization and a resilient network operation.**

Keywords*— *SDN, NSGA-II, PSA.

I. INTRODUÇÃO

O paradigma de redes definidas por *software* (SDN, do inglês *Software Defined Network*) introduz uma solução de engenharia de tráfego que, por meio de uma visão global da rede, apresenta potencial para aperfeiçoar o uso de recursos de rede, através de mecanismos de autoconfiguração e otimização de métricas de desempenho como latência, balanceamento de carga e resiliência. Para tanto, as redes SDN se baseiam em uma arquitetura logicamente centralizada, que separa o plano de controle do plano de dados.

Diferentemente das redes convencionais onde, para cada dispositivo de rede, protocolos e conjuntos de regras são implementados para controlar e monitorar o fluxo de dados, nas redes SDN as funções de controle dos dispositivos de rede são movidas para um controlador externo dedicado, implementado em *software* [2]. Desta forma, no plano de controle, um controlador logicamente centralizado otimiza as decisões de encaminhamento e separação de tráfego. No plano de dados, o encaminhamento de pacotes é realizado pelos dispositivos de rede de acordo com os comportamentos para manipulação de pacotes ditados pelo plano de controle. Essa separação permite a construção dinâmica e eficiente de redes mais simples através dos controladores.

A comunicação entre o plano de controle e o plano de dados é realizada por *interfaces* de programação de aplicações (API, do inglês *Application Programming Interfaces*) [3] implementadas por protocolos como o OpenFlow [4]. O OpenFlow é um protocolo aberto que permite que um controlador gerencie *switches* e dite o comportamento dos mesmos. Para prover aspectos como escalabilidade e resiliência, é possível distribuir fisicamente o plano de controle entre vários controladores com base no conceito de HyperFlow [5], que permite o particionamento do OpenFlow em múltiplos domínios. Um grande desafio desse novo paradigma envolve a avaliação do impacto da posição de cada controlador nas métricas de desempenho diante de mudanças dinâmicas nas condições da rede, e a definição da quantidade de controladores requeridos para operação confiável e resiliente da rede.

O problema de posicionamento do controlador em redes SDN foi introduzido pela primeira vez por Heller *et al.* [6], que defendem a sua relevância em um estudo quantitativo do impacto do posicionamento na latência de propagação entre os nós e o controlador. Para isso, Heller *et al.* [6] propõem uma análise exaustiva de todas as posições possíveis, a fim de analisar as características das soluções ótimas. Este trabalho foi reforçado por Guang Yao *et al*. [7] através de uma análise do impacto do posicionamento dos controladores no balanceamento de carga, de forma que a capacidade de cada controlador é limitada pela largura de banda dos seus acessos, o que resulta em um número limitado de dispositivos associado ao controlador em um determinado momento.

No entanto, em ambientes reais, as métricas de desempenho são concorrentes, de forma que não há uma solução que satisfaça simultaneamente objetivos de otimização como latência e balanceamento. Por este motivo, propõe-se que o problema do posicionamento dos controladores seja tratado como um problema de otimização combinatória multiobjetivo (MOCO, do inglês *Multi-Objective Combinatorial Optimization*), onde a solução mais apropriada seja selecionada pelo tomador de decisão a partir de um conjunto de soluções ótimas, correspondentes a compromissos entre os diferentes objetivos. Este conjunto é conhecido como conjunto ótimo de Pareto. Movendo-se de um conjunto de soluções de Pareto para outro, um objetivo é sacrificado para que se obtenha ganho em outro. Neste contexto, Hock *et al*. [8] investigam uma versão estendida do problema de posicionamento do controlador que lida com a otimização de múltiplos objetivos, como considerações de resiliência e latência entre controladores.

Em problemas combinatórios pequenos e médios, avaliar de maneira exaustiva todo o espaço de variáveis e encontrar o conjunto ótimo de Pareto pode ser possível com um esforço computacional razoável. Entretanto, para redes de larga escala, a avaliação exaustiva necessita de uma quantidade considerável de esforço computacional e uso de memória. É neste contexto que se torna necessária a busca por uma solução computacional rápida, capaz de analisar os padrões de tráfego e as demandas de largura de banda para promover o posicionamento adequado dos controladores, em termos de desempenho e tolerância a falhas.

Lange *et al.* [1] mostram que o problema de posicionamento do controlador para redes de larga escala pode ser otimizado via soluções heurísticas capazes de apresentar soluções justas e oportunas, com boa relação entre precisão e tempo de processamento. Para tanto, eles implementam a metaheurística *Pareto Simulated Annealing* (PSA), que produz resultados dentro de dezenas de segundos para instâncias de problemas cuja solução exaustiva leva dezenas de minutos, com uma média de erro de 2%. O estudo proposto ainda disponibiliza um ambiente em MATLAB que permite calcular posicionamentos ótimos de controladores por meio de técnicas exaustiva e heurística, com respeito a diferentes medidas de desempenho. Este ambiente, conhecido como POCO, apresenta uma *interface* gráfica com uma clara ilustração dos compromissos entre as métricas concorrentes, além de permitir a extensão de funcionalidades como a implementação de novos mecanismos para o cálculo e a análise do posicionamento do controlador. Em um segundo trabalho, Lange *et al.* [9] propõem a utilização do algoritmo *Pareto Capacitated k-Medoids* (PCKM), que combina algoritmos de teoria dos grafos para construir uma aproximação da fronteira de Pareto com respeito a duas funções objetivo.

Já a utilização de algoritmos genéticos foi introduzida por Jalili *et al*. [10] através da implementação e avaliação do *Nondominated Sorting Genetic Algorithm* II (NSGA-II) [11], onde concluiu-se que o NSGA-II é capaz de prover uma boa aproximação da solução ideal para o problema do posicionamento do controlador.

Dentro deste contexto, este trabalho explora o algoritmo genético multiobjetivo NSGA-II como uma heurística para o problema de posicionamento, similarmente ao realizado em [10], mas adicionalmente discute os impactos na precisão e no esforço computacional necessário, e realiza uma comparação de resultados entre esta reconhecida metaheurística evolutiva e os resultados obtidos com a utilização do algoritmo PSA na ferramenta POCO.

A estrutura deste trabalho apresenta na Seção II a fundamentação teórica para o problema do posicionamento dos controladores. Na Seção III, são apresentadas as soluções heurísticas implementadas, cujo desempenho é discutido na Seção IV. Por fim, a Seção V tece considerações finais.

II. Definição do Problema

Considere uma rede composta por n nós, tais como *switches* e controladores, conectados entre eles. Desta forma, a rede é representada pelo grafo $G = (V, E)$, onde V e E denotam o conjunto de nós (vértices) e o conjunto de conexões disponíveis (arestas), respectivamente. A menor latência entre cada par de nós compõe a matriz de distâncias D, onde d_{ij} representa a distância entre o nó i e o nó j. Os valores em D são normalizados pelo diâmetro do grafo.

Dado um número de controladores k, a tarefa de otimização está em determinar um conjunto de posições $\mathcal{P}$ para os k controladores, tal que $\mathcal{P}_k = \{\mathcal{P} \in 2^V | \, |\mathcal{P}| = k\}$, de forma que as funções objetivo $f_i \; (i \in \{1, \ldots, J\})$ sejam minimizadas.

As funções objetivo mapeiam objetivos individuais, como latência e balanceamento de carga, em valores numéricos. Neste trabalho, foram consideradas as funções objetivo implementadas por [1] para o caso livre de falha, i.e. não se considera a hipótese de um dos controladores parar de funcionar. As funções objetivo definidas para avaliação da latência entre os nós tem a finalidade de prover informação sobre a conectividade (*i*) entre cada nó e o controlador ao qual ele está associado e (*ii*) entre os controladores.

Dada a matriz de distância D, a latência entre os nós e o controlador pode ser mensurada pelo cálculo da média $\pi^{avg\ lat\ N2C}$ através das latências de todos os nós para um controlador p na posição examinada ou selecionando o valor máximo $\pi^{max\ lat\ N2C}$ do pior caso, conforme as Equações (1) e (2).

$$\pi^{avg\ lat\ N2C}(\mathcal{P}) = \frac{1}{|V|} \sum_{v \in V} \left(min_{p \in \mathcal{P}} d_{v,p} \right), \quad (1)$$

$$\pi^{max\ lat\ N2C}(\mathcal{P}) = max_{v \in V} min_{p \in \mathcal{P}} d_{v,p}. \quad (2)$$

A latência entre os controladores, por sua vez, pode ser mensurada calculando-se a média das latências entre todos os pares p_1, p_2 de controladores $\pi^{avg\ lat\ C2C}$ ou selecionando o valor máximo do pior caso $\pi^{max\ lat\ C2C}$, conforme as Equações (3) e (4).

$$\pi^{avg\ lat\ C2C}(\mathcal{P}) = \frac{1}{\binom{|\mathcal{P}|}{2}} \sum_{p_1, p_2 \in \mathcal{P}} d_{p_1,p_2}, \quad (3)$$

$$\pi^{max\ lat\ C2C}(\mathcal{P}) = max_{p_1, p_2 \in \mathcal{P}} d_{p_1,p_2}. \quad (4)$$

A avaliação do balanceamento de carga entre os controladores é definida de forma a minimizar a diferença entre o número de nós associados a cada controlador, tal que n_p indica o número total de nós associados ao controlador p em uma dada posição em $\mathcal{P}$. Para tanto, $\pi^{balanceamento}$ indica a diferença em valores de n_p entre os controladores com menor e maior número de nós associados, ou seja,

$$\pi^{balanceamento}(\mathcal{P}) = max_{p \in \mathcal{P}} n_p - min_{p \in \mathcal{P}} n_p. \quad (5)$$

Um dado conjunto de posições $\mathcal{P}$ é dito pertencer à fronteira de Pareto se não existir nenhum outro conjunto $\mathcal{P}'$ factível, capaz de melhorar um dos objetivos do problema (em relação a $\mathcal{P}$) sem simultaneamente piorar pelo menos um dos demais. Todas as soluções pertencentes à fronteira de Pareto são ditas não dominadas. Se todas as funções objetivo são para minimização, pelo conceito de dominância entende-se que uma dada solução $\mathcal{P}'$ domina uma solução $\mathcal{P}$ se, e somente se, $\forall i \; f_i(\mathcal{P}') \leq f_i(\mathcal{P})$ para todo i = 1, 2, .., J e $f_i(\mathcal{P}') < f_i(\mathcal{P})$ para pelo menos um i [12].

III. Soluções Heurísticas

Nesta Seção são apresentadas as abordagens heurísticas PSA e NSGA-II, com suas devidas adaptações para o problema multiobjetivo de posicionamento do controlador.

A. Pareto Simulated Annealing

O *Pareto Simulated Anneling* [13] é um mecanismo derivado da conhecida técnica de otimização *Simulated Annealing*. Ele permite explorar apenas um subconjunto do

espaço de busca no âmbito da otimização combinatória multiobjetivo, a fim de encontrar a fronteira de Pareto com soluções ótimas.

O *Simulated Anneling* original é um método de Monte Carlo que permite movimentos de busca para soluções piores que a atual, com o intuito de evitar ótimos locais. Para determinar a probabilidade de se aceitar tal movimento, este método se baseia em um parâmetro de controle conhecido como "temperatura", que incorpora níveis de dispersão durante a análise dos subconjuntos. A probabilidade de movimento para uma solução pior reduz gradualmente com o decaimento da temperatura ao longo das iterações. Aceitando soluções bastante ruins no início, devido à temperatura elevada, têm-se uma cobertura mais ampla do espaço de pesquisa, enquanto a menor probabilidade de aceitação no final, devido à redução da temperatura, leva à convergência.

A implementação do PSA desenvolvida no POCO por Lange *et al*. [1] é baseada em [13] e apresentada no Algoritmo 1. Definem-se como entradas a topologia G, o número de controladores k e os parâmetros do PSA, que incluem a temperatura inicial T_0, a taxa de redução da temperatura ρ, o número de iterações por nível de temperatura m e o número de arranjos de posições a serem avaliadas durante cada iteração s. Define-se também um conjunto S contendo s arranjos de k posições escolhidas aleatoriamente. Para cada arranjo, é associado um vetor de pesos $\Lambda = [\lambda_1, \lambda_2, \dots, \lambda_M]$ aleatório e independente, onde M indica o número de objetivos responsável por ativar dispersão sobre todas as regiões do conjunto não dominado através da troca de informações sobre os arranjos. A fronteira de Pareto atualizada à cada iteração é definida por F, sendo inicialmente composta por S.

Algoritmo 1: Pareto Simulated Annealing

1: **entradas**: $G = (V,E)$, k, s, m, T_0, ρ
2: $n = |V|$
3: S = geraPosiçõesRandomicamente(n,s,k)
4: Λ = geraPesosRandomicamente(S)
5: F = fronteiraPareto(avaliaPosições(S))
6: $T = T_0$
7: **enquanto** $T > 1$ **faça**
8: Y = defineVizinhos$\left(S, n, \left\lceil \frac{kT}{2T_0} \right\rceil\right)$
9: atualizaFronteiraPareto (F,Y)
10: Λ = atualizaPesos(S)
11: S:= aceita $y \in Y$ com probabilidade $P(S,Y,T,\Lambda)$
12: **se** m iterações forem efetuadas em T então
13: $T = T_0$
14: **fim se**
15: **fim enquanto**
16: retorna F e as posições correspondente

O processamento do algoritmo inicia a uma temperatura $T = T_0$, e ao final de m iterações a temperatura T é decrementada por um fator ρ, até que seja menor ou igual a 1. A cada iteração, são gerados novos arranjos de posições vizinhos aos contidos em S, que são armazenados em Y. Para que os arranjos sejam considerados vizinhos, eles podem diferenciar em até $\left\lceil \frac{k}{2} \right\rceil$ posições. Após cada geração, os arranjos vizinhos são integrados à F e os vetores de pesos Λ são recalculados, de forma que o vetor de pesos associado ao arranjo x seja modificado para aumentar a probabilidade de afastar-se do seu vizinho mais próximo y, aumentando-se os pesos dos objetivos para os quais x é melhor que y e diminuindo-se os pesos para os objetivos para os quais x é pior que y. Um elemento de S é substituído pelo seu vizinho em Y com probabilidade $P(S,Y,T,\Lambda)$, tal que esta probabilidade é igual a 1 quando o arranjo vizinho constitui uma melhoria e decrementado para indicar uma piora em função da quantidade de deterioração (determinada pela diferença entre S e Y), da temperatura T e da matriz de pesos Λ em cada iteração.

B. NSGA-II

O NSGA-II [11] é um algoritmo evolutivo rápido e elitista que permite explorar apenas um subconjunto do espaço de busca no âmbito da otimização combinatória multiobjetivo, a fim de encontrar uma fronteira de Pareto mais próxima da ideal, sendo amplamente utilizado em problemas mais complexos. Para tanto, o NSGA-II se baseia no conceito de população, e utiliza os operadores de seleção por torneio, recombinação e mutação para criar novas populações capazes de gerar soluções melhores a cada iteração.

A implementação do NSGA-II desenvolvida neste trabalho é baseada no algoritmo proposto em [14] e apresentada no Algoritmo 2. Define-se como entradas do algoritmo a topologia G, número de controladores k, o tamanho da população N, o número máximo de iterações $MaxIt$, a taxa de recombinação t_c e a taxa de mutação t_m. Denomina-se por indivíduo um arranjo de posições que compõe a população. Para cada indivíduo, são calculados os valores das funções objetivo. Por fim, F define as fronteiras de Pareto obtidas a cada iteração.

Algoritmo 2: NSGA-II

1: **entradas**: $G = (V,E)$, k, N, t_c, t_m,$MaxIt$
2: $n = |V|$, $n_c = t_c . N$, $n_m = t_m . N$
3: P = geraPosiçõesRandomicamente(n,N,k)
4: $[P, F]$ = ordenaPorNãoDominância(P)
5: P = calcDistanciaAglomeração(P, F)
6: $[P, F]$ = ordenaPopulação(P)
7: **para** $it = 1 : MaxIt$
8: **para** $i = 1 : 2 : n_c$
9: p_1 = selecionaTorneio(P)
10: p_2 = selecionaTorneio(P)
11: $[PC_i, PC_{i+1}]$ = recombina(p_1, p_2)
12: **fim para**
13: **para** $i = 1 : n_m$
14: m = selecionaTorneio(P)
15: PM_i = muta(m)
16: **fim para**
17: $P = [P, PC, PM]$
18: $[P, F]$ = ordenaPorNãoDominância(P)
19: P = calcDistanciaAglomeração(P, F)
20: P = ordenaPopulação(P)
21: $P = P(1:N)$
22: $[P, F]$ = ordenaPorNãoDominância(P)
23: P = calcDistanciaAglomeração(P, F)
24: $[P, F]$ = ordenaPopulação(P)
25: $F1 = P(F\{1\})$
26: **fim para**
27: retorna $F1$ e as posições correspondentes

A partir da população inicial P_0, composta por N arranjos de k posições escolhidas aleatoriamente, cada arranjo é comparado aos demais a fim de verificar a relação de dominância entre eles e assim ordená-los em níveis de não dominância. Este procedimento constitui o conceito de *fast non-dominated sorting*, cujo algoritmo é definido em [11].

Para manter uma boa dispersão entre os arranjos obtidos, o NSGA-II utiliza um critério conhecido como "distância de aglomeração", que permite avaliar o espalhamento das soluções classificadas em um mesmo nível de não dominância: ela é medida para cada objetivo, de forma que as soluções situadas nos extremos (soluções com maior e menor valor) têm a distância de aglomeração igual a ∞, enquanto que para as soluções intermediárias este valor é definido pela diferença normalizada absoluta dos valores do objetivo de duas soluções adjacentes. O valor geral da distância de aglomeração de uma

solução é a soma dos valores de distâncias individuais correspondentes a cada objetivo, de forma que um valor geral pequeno indica que uma dada solução se encontra em uma região menos densa.

A medida da distância de aglomeração é definida pelo perímetro normalizado do cuboide envolvendo a solução i, representado pela Fig. 1, cujos vértices identificam as soluções adjacentes a i. Desta forma, as soluções localizadas ao extremo do nível de não dominância terão sua distância igual a ∞.

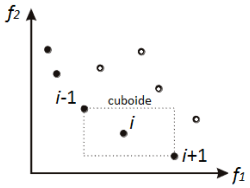

Fig. 1. Cálculo da distância de aglomeração. Os círculos sólidos representam membros de um mesmo nível de não dominância, onde a distância de aglomeração da *i*-ésima solução é o perímetro do cuboide mostrado na caixa tracejada [11].

A seleção dos indivíduos da população a serem utilizados no processo de recombinação é realizada através da técnica de torneio ternário com base na aptidão de cada arranjo, tal que a aptidão de um determinado arranjo i é avaliada de acordo com o nível de não dominância no qual foi classificado ($i^{classificação}$) e a distância de aglomeração ($i^{distância}$), por meio do operador de comparação de aglomeração ($\prec_n$):

$$i \prec_n j, se\ i_{classificação} < j_{classificação} \quad (6)$$

$$ou\ ((i_{classificação} = j_{classificação})\ e\ (i_{distância} > j_{distância}))$$

Desta forma, o NSGA-II opta por soluções mais bem classificadas com relação à não-dominância e, se a classificação for a mesma, prefere-se aquela localizada em uma região com menor aglomeração, para se manter dispersão na população gerada.

Após selecionados os indivíduos, a recombinação é realizada mantendo-se as posições de controladores unânimes entre os indivíduos selecionados e escolhendo aleatoriamente as demais posições dentre as restantes contidas nestes. A mutação, por sua vez, consiste em escolher posições aleatórias, dentro de um arranjo de posições também selecionado por torneio ternário, para serem substituídas aleatoriamente.

A cada iteração, $t_c.N$ recombinações e $t_m.N$ mutações são geradas. Os indivíduos (arranjos) gerados são concatenados à população atual, a população resultante é reclassificada em níveis de não-dominância e a distância de aglomeração é recalculada. De acordo com a ordem de classificação das melhores soluções, os N primeiros indivíduos são selecionados para serem a população da geração (iteração) seguinte e são organizados em fronteiras de acordo com o nível de classificação de não-dominância. A fronteira de Pareto será determinada pela fronteira cuja classificação de não-dominância for igual a 1, ao final de todas as iterações.

IV. SIMULAÇÃO E ANÁLISE

Nesta Seção, são apresentados os resultados de desempenho obtidos pelos algoritmos propostos nas soluções heurísticas PSA e NSGA-II, a primeira já implementada no POCO e a segunda cuja implementação foi proposta neste trabalho. O caso de estudo utilizado consiste da topologia de rede da *Internet2 OS3E,* disponibilizada pelo *Internet Topology Zoo* [15], que contém n = 34 nós. Objetiva-se encontrar o posicionamento ótimo para k = 6 controladores, de forma que as funções objetivo descritas na Seção II sejam minimizadas. Para tanto, os algoritmos PSA e NSGA-II foram configurados com os parâmetros de entrada a seguir:

- PSA: m = 60, T_o = 50, ρ = 0.9 e s = 10;
- NSGA-II: $MaxIt$ = 600, t_c= 0.8, t_m=0.4 e N = 10.

O critério definido para escolha dos parâmetros de entrada foi de que ambos os algoritmos trabalhassem com uma população de 10 arranjos de posições e que os demais parâmetros fossem capazes de explorar os operadores de recombinação e mutação no NSGA-II e do controle de temperatura no PSA.

A Fig. 2 ilustra a evolução da fronteira de Pareto estimada para as funções objetivo de balanceamento e de máxima latência nó-controlador durante o processamento dos algoritmos propostos. É possível verificar que ambos os algoritmos alcançam a fronteira de Pareto ideal, obtida pela solução exaustiva.

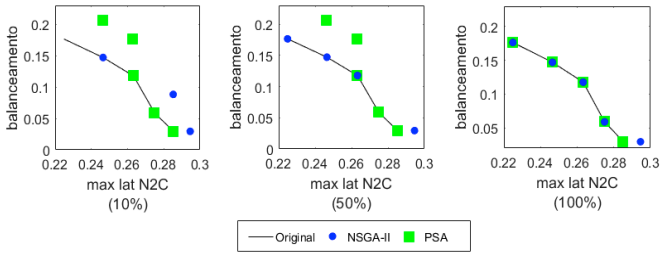

Fig. 2. Evolução da fronteira de Pareto estimada com 10% (esquerda), 50% (centro) e 100% (direita) de execução decorrida dos algoritmos .

Uma métrica de desempenho comumente utilizada para medir qualidade de algoritmos multiobjetivo é a distância geracional [16]. Dado um conjunto ótimo de Pareto candidato obtido por uma solução heurística $A = \{a_1, \dots, a_{\mathcal{M}}\}$ e o conjunto ótimo de Pareto obtido pela solução exaustiva $B = \{y_1, \dots, y_{\mathcal{M}}\}$, a distância geracional é definida por $DG(A) = \frac{1}{\mathcal{M}}\left(\sum_{i=1}^{N} d_i^p\right)^{\frac{1}{p}}$, onde d_i denota a distância Euclidiana entre a solução a_i e a solução mais próxima em B. Um algoritmo com um pequeno valor de DG é melhor.

A fim de analisar o desempenho dos algoritmos, foi calculada a distância geracional média para 100 simulações, cujos resultados numéricos são apresentados na Fig. 3. A comparação da distância geracional média ao longo das iterações permite mostrar que ambas as soluções heurísticas são capazes de convergir para a fronteira de Pareto ideal, com uma vantagem, em termos de distância média final, para o NSGA-II.

Além disso, foi avaliado o número de gerações vs. distância geracional em uma única simulação, cujos resultados são apresentados na Fig. 4. Por gerações entende-se o total de soluções-candidatas geradas desde a inicialização. É possível observar que no decorrer das gerações aumenta-se a probabilidade de gerar soluções melhores por meio dos operadores de recombinação e mutação no NSGA-II e por meio do controle de temperatura no PSA. Além de definir a forma como cada heurística opera, tais operadores influem também na extensão do espaço de busca que irão explorar.

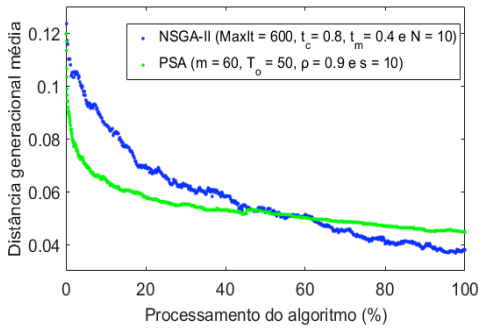

Fig. 3. Distância geracional média ao longo do percentual de execução decorrido dos algoritmos NSGA-II e PSA.

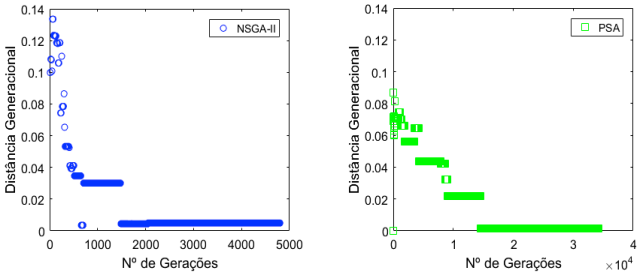

Fig. 4. Resultados numéricos do NSGA-II e do PSA medidos pela distância geracional.

O espaço de busca explorado no método exaustivo é definido pelo número de combinações de k controladores em uma rede contendo n nós, ou seja $\binom{n}{k}$. No NSGA-II, este número é definido por $MaxIt.N.(p_m + p_c)$, e no PSA é definido por $\left[\frac{-\log T_0}{\log \rho}\right].m.s$ [1]. Para o cenário estudado, tem-se $\binom{n=34}{k=6}$ = 1.368.000 combinações de controladores possíveis. Observa-se, também na Fig. 4, que através das soluções metaheurísticas é possível alcançar uma fronteira de Pareto estimada muito próxima da ideal com apenas 0.15% e 2% do espaço de busca total com o NSGA-II e o PSA, respectivamente.

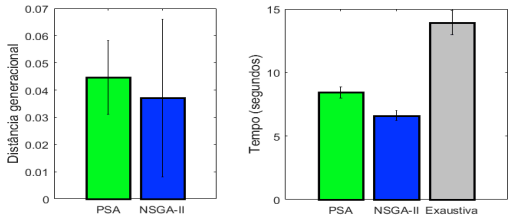

Fig. 5. Resultados numéricos do NSGA-II e do PSA medidos pela média e desvio padrão da distância geracional e do tempo de processamento para 100 simulações.

Apesar do cenário estudado apresentar um espaço de busca relativamente pequeno para a utilização de metaheurísticas, ele retrata bem uma rede real. Uma breve comparação da execução dos algoritmos para 100 simulações é apresentada pelos gráficos de média e desvio padrão representados na Fig. 5, cuja análise permite concluir que ambas as soluções metaheurísticas são capazes de alcançar soluções ótimas em tempo de processamento inferior à solução exaustiva, característica desejável para otimização de tráfego dinâmico em tempo real e para garantia da operação confiável e resiliente da rede. As soluções obtidas pelo NSGA-II, apesar de apresentarem uma média inferior às soluções obtidas com o PSA, tiveram um desvio padrão mais elevado, o que pode ser explicado pelo menor espaço de busca utilizado pelo primeiro.

V. CONCLUSÕES

Este trabalho apresentou a utilização do POCO como ferramenta para computar o posicionamento ótimo de controladores. Além de validar os resultados obtidos por Lange *et al.* [1] através da implementação da heurística PSA, foi também proposta a implementação da heurística NSGA-II. Os resultados permitiram concluir que ambas as soluções heurísticas são capazes de convergir para a fronteira de Pareto ideal ao longo de seu processamento, com grande vantagem de tempo em relação à solução exaustiva. O NSGA-II, por sua vez, apresentou vantagem em termo de convergência e grau de exploração do espaço de busca.

REFERÊNCIAS

[1] S. Lange, S. Gebert, T. Zinner, P. Tran-Gia, D. Hock, M. Jarschel, and M. Hoffmann, "Heuristic Approaches to the Controller Placement Problem in Large Scale SDN Networks," *IEEE Transactions on Network and Service Management*, 2015.

[2] B. Nunes, Marc Mendonca, X. Nguyen, Katia Obraczka, Thierry Turletti, *A survey of software-defined networking: Past present and future of programmable networks*, pp. 1-18, 2014.

[3] M. Jarschel, T. Zinner, T. Hoßfeld, P. Tran-Gia, W. Kellerer, "Interfaces attributes and use cases: A compass for SDN", IEEE Commun. Mag., vol. 52, no. 6, pp. 210-217, Jun. 2014.

[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks", *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69-44, Apr. 2008.

[5] A. Tootoonchian, Y. Ganjali, "HyperFlow: A distributed control plane for OpenFlow", Proc. INM/WREN, pp. 1-6, 2010.

[6] B. Heller, R. Sherwood, N. McKeown, "The controller placement problem", *Proc. HotSDN*, pp. 7-12, 2012.

[7] G. Yao, J. Bi, Y. Li, L. Guo, "On the Capacitated Controller Placement Problem in Software Defined Networks", IEEE Commun. Letters, vol. 18, no. 8, pp. 1339-1342, Aug. 2014.

[8] D. Hock, M. Hartmann, S. Gebert, M. Jarschel, T. Zinner, and P. Tran-Gia, "Pareto-optimal resilient controller placement in SDN-based core networks", *Proc. 25th ITC*, pp. 1-9, 2013.

[9] S. Lange et al., "Specialized heuristics for the controller placement problem in large scale SDN networks", *Proc. ITC*, pp. 210-218, 2015.

[10] A. Jalili, V. Ahmadi, M. Keshtgari, M. Kazemi, "Controller placement in software-defined wan using multi objective genetic algorithm", Knowledge-Based Engineering and Innovation, 2015.

[11] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: NSGA-II", *IEEE Transactions on Evolutionary Computation*, *6*(2), 182-197, 2002.

[12] A. Konak, D. Coit, A. Smith, "Multi-objective optimization using genetic algorithms: A tutorial", Rel. Eng. & Sys. Safety 91 (9): 992-1007, 2006.

[13] P. Czyżżak, A. Jaszkiewicz, "Pareto simulated annealing—A metaheuristic technique for multiple-objective combinatorial optimization", J. Multi-Criteria Decision Anal., vol. 7, no. 1, pp. 34-47, Jan. 1998.

[14] Project Title: Non-dominated Sorting Genetic Algorithm II (NSGA-II) Yarpiz (www.yarpiz.com).

[15] S. Knight, H. Nguyen, N. Falkner, R. Bowden, M. Roughan, "The internet topology zoo", *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765-1775, Oct. 2011.

[16] Veldhuizen, D. A. V. e Lamont, G. B., "On measuring multiobjective evolutionary algorithm performance", *Evolutionary Computation. Proceedings of the 2000 Congress on*, volume 1, páginas 204–211 vol.1, 2000.

Ana Carolina de Oliveira Christófaro, Marcelo Menezes de Carvalho e Daniel Guerreiro e Silva, Faculdade de Tecnologia, Universidade de Brasília (UNB), Brasília-DF, Brasil, E-mails: ana.christofaro@gmail.com, mmcarvalho@ene.unb.br, danielgs@ene.unb.br.