Localização de Controladores em Redes Definidas por Software com Orientação Financeira

Alexander D. de Sousa[1], **Luiz F. M. Vieira**[1], **Marcos A. M. Vieira**[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Minas Gerais
31270-901 - Belo Horizonte - MG - Brasil

{alexanderdecker,lfvieira,mmvieira}@dcc.ufmg.br

Abstract. *Controller placement problems seek to spatially position the controllers of a software-defined network (SDN) and choose the links that will serve them. These problems are NP-Hard. In this work, we propose a new formulation focused on long distance networks based on minimizing the financial costs of network maintenance. Our modeling still limits latency within the network and avoids overloading the controllers. We proved that the decision version of our formulation is NP-Complete. We created an exact algorithm for the offline structuring of the network and a set of heuristics for the adaptation of the topology at runtime. Our heuristic results reach about 5% surplus over the optimal cost within simulated test cases. Therefore, our solution can lead to financial savings for the maintenance of SDN long distance networks.*

Resumo. *Os problemas de localização de controladores buscam posicionar espacialmente os controladores de uma rede definida por software (SDN) e escolher os enlaces que atenderão os mesmos. Esses problemas são NP-Difícil. Neste trabalho, é proposta uma nova formulação focada em redes de longa distância baseada na minimização dos custos financeiros de manutenção da rede. Nossa modelagem ainda limita a latência dentro da rede e evita a sobrecarga dos controladores. É provada a versão de decisão da nossa formulação como NP-Completa. Foi criado um algoritmo exato para a estruturação offline da rede e um conjunto de heurísticas para a adaptação da topologia em tempo de execução. Os resultados das heurísticas alcançam cerca de 5% de excedente em relação ao custo ótimo dentro dos casos de teste simulados. Portanto, a solução proposta pode levar a uma economia financeira para a manutenção de redes SDN de longa distância.*

1. Introdução

O paradigma de Redes Definidas por *Software*, ou *SDN* (do inglês, *Software Defined Networks*), surgiu nos últimos anos e tem se mostrado muito promissor. Dentre suas várias vantagens, destaca-se, por exemplo, seu potencial para a resolução da chamada *ossificação da Internet*, termo que se refere ao fato de novas tecnologias e protocolos de rede serem dificilmente inseríveis no mundo real, devido ao perigo de interrupção da rede e consequente dano a atividades às quais a *Internet* já se tornou ferramenta essencial [Guedes et al. 2012].

O princípio fundamental que rege as *SDN* consiste na definição da operação de elementos comutadores da rede, como roteadores e *switches*, através de comandos enviados em tempo de execução. As variações de *SDN* são diversas, porém todas mantém uma estrutura básica em três camadas [Guedes et al. 2012]. A camada mais inferior consiste no plano de dados e é responsável pelo encaminhamento de mensagens propriamente dito. Por demandar alto desempenho, esta camada é mantida simples e genérica, sendo baseada na leitura de tabelas programáveis [McKeown et al. 2008] ou na execução de microcódigos alteráveis em tempo de execução [Jouet and Pezaros 2017]. A camada intermediária consiste no plano de controle, constituído de dispositivos ou *softwares* controladores. Os controladores atuam efetivamente na reprogramação dos elementos de rede, oferecendo recursos para facilitar o controle por parte das aplicações como, por exemplo, uma visão logicamente centralizada da rede [Guedes et al. 2012]. Finalmente, na terceira camada rodam as aplicações de rede em si, que atuam através dos controladores.

A possibilidade de se redefinir facilmente grande parte das operações dos elementos de rede permite uma maior flexibilização das aplicações, a criação de aplicações otimizadas para cada contexto e facilidade no que diz repeito à atualização ou troca de protocolos ou padrões. Por estas e outras várias vantagens, a definição por *software* ganhou espaço dentre diversos tipos de redes, desde as locais e interiores a *datacenters* até as de longa distância, ou *WANs* (do inglês *Wide Area Networks*). Estas últimas, em especial, sofrem com algumas dificuldades devido principalmente à longa extensão de seus enlaces e ao seu geral grande volume de tráfego. De fato, a localização e interligação dos controladores *SDN* deve ser tal que o *round trip time* dos pares de requisição e resposta entre o comutador e seu controlador não rompa um limite máximo aceitável, ao mesmo tempo em que a carga sobre cada controlador não exceda sua capacidade de processamento. As chamadas *sdWANs* estão ganhando espaço nos cenários acadêmico e mercadológico [Jain et al. 2013] atuais, aplicando aspectos de definição por *software* a nível de *intranet* nos centros de dados ou mesmo em *backbones*, mesmo que estes últimos ainda sejam utilizados para propósitos acadêmicos e experimentais fora de infra-estruturas essenciais para a *Internet*.

Sendo assim, a elaboração inteligente da topologia da rede, considerando as interligações entre os diversos elementos de *hardware* e o posicionamento dos mesmos, pode ser complexa em demasia. O posicionamento geográfico de controladores em redes *sdWAN* é NP-difícil e se trata de uma variação do problema de localização de facilidades (*Facility Location Problem*). Os problemas que tratam do posicionamento ou da escolha dos enlaces que ligam os controladores aos comutadores ou mesmo entre os próprios controladores são chamados de Problemas de Localização de Controladores (*Controler Placement Problems*) [Heller et al. 2012].

Há diversos trabalhos abordando o referido problema na literatura, como será melhor explorado na seção 2. Em geral, é buscada uma configuração de controladores, enlaces e comutadores que otimize determinado aspecto de rede, como por exemplo a latência total entre pontos extremos da rede ou a confiabilidade da mesma. Todavia, ainda não foi devidamente explorada uma abordagem voltada à economia de recursos financeiros, em especial reduzindo os custos de manutenção. A otimização do custo financeiro da energia em *sdWANS*, em particular, é bastante aplicável pelo fato dos nós se localizarem em cidades ou mesmo províncias muitas vezes distantes, o que pode fazer com que o preço

da energia varie muito de um nó para outro. Dessa forma, um esquema de balanceamento de carga apropriado pode ser imperativo para uma boa economia de recursos financeiros. Afinal, por se tratar de um sistema de computação de larga escala, os custos energéticos se mostram muito notáveis. Estima-se que em 2010 de 1,1% a 1,5% dos gastos de energia ao redor do mundo foram empregados no abastecimento de *datacenters* [Koomey 2011], ao passo em que no mesmo ano de 1,7% a 2,2% do consumo de energia dos Estados Unidos foi empregado com esse mesmo fim. Além disso, tal consumo mostra-se crescente ao longo dos anos, tendo aumentado cerca de 56% de 2005 a 2010.

Neste artigo é detalhado um sub-problema de otimização voltado à resolução do *Problema de Localização de Controladores* com enfoque na redução dos custos financeiros de manutenção de *WANs* definidas por *software*. Além da redução do consumo bruto de energia, o modelo também considera o custo da energia em cada localidade e oferece garantias com respeito à latência e ao balanceamento de carga. Dentre as demais contribuições deste trabalho destacam-se a prova de NP-completude da versão de decisão do problema aqui proposto, a elaboração de uma solução exata para o mesmo e ainda a proposição de heurísticas que podem ser utilizadas para o controle de topologia em tempo de execução[1]. O resto deste texto está organizado da seguinte maneira: a seção 2 apresenta os trabalhos relacionados, a seção 3 define o problema a ser resolvido, a seção 4 apresenta a modelagem analítica do problema, a seção 5 descreve a prova de NP-Completude do problema aqui enunciado, a seção 6 descreve os algoritmos propostos, a seção 7 mostra os resultados experimentais e a seção 8 mostra as conclusões.

2. Trabalhos Relacionados

A literatura relacionada aos Problemas de Localização de Controladores é normalmente agrupada de acordo com o objetivo da otimização. Um dos objetivos mais frequentes da literatura é a minimização da latência, que inclui a minimização do tempo de propagação das mensagens, do tempo em que as mensagens ficam armazenadas em filas e do tempo de processamento dos controladores. Esta abordagem também envolve a escolha dos enlaces de forma a fazer a distribuição balanceada de carga dentre os controladores. A maioria dos trabalhos aplica algoritmos populares de resolução do Problema das K-Medianas Mínimas [Wang et al. 2017], visto que esta versão do problema se assemelha muito com o Problema de Localização de Facilidades.

Outro objetivo comum é o aumento da confiabilidade e da resiliência. A confiabilidade, no caso, diz respeito ao inverso da probabilidade de falha de um enlace e a resiliência é a capacidade da manutenção do funcionamento da rede mesmo na presença de nós ou enlaces errantes. Assim como a minimização da latência, este problema se assemelha à localização de facilidades e, muitas vezes, os trabalhos na literatura também fazem uso de algoritmos aplicáveis ao Problema das K-Medianas Mínimas. Em [Hu et al. 2014], por exemplo, é utilizada a meta-heurística de Recozimento Simulado (*Simulated Annealing*) para uma solução não exata do problema. Outros trabalhos objetivam a redução do consumo bruto de energia a partir da definição *online* de quais dispositivos ligar ou desligar [Ruiz-Rivera et al. 2015]. Há ainda uma abordagem pouco explorada [Wang et al. 2017], que consiste na otimização multi-objetivo. [Zhang et al. 2016] é um dos poucos trabalhos a contemplar esta abordagem, buscando otimizar a confiabili-

[1] todos os códigos estão disponíveis em github.com/AlexDecker/eCPPSolver

dade da rede, balancear a carga entre os controladores e minimizar o pior caso da latência.

Este trabalho, ao contrário dos supracitados, busca a redução dos custos financeiros com energia como objetivo único. Todavia, o modelo de otimização incorpora diversas restrições relacionadas com aspectos de rede, como latência e distribuição de carga, o que permite a otimização multi-objetivo através do ajuste de tais restrições. Assim como [Ruiz-Rivera et al. 2015], as heurísticas aqui propostas podem ser adaptadas facilmente para o contexto distribuído, escolhendo controladores e enlaces em tempo de execução de acordo com o tráfego em cada comutador (vide subseção 7.2).

3. Definição do Problema

O problema aqui tratado segue uma abordagem diferente dos demais presentes na literatura. Ao invés de buscar a otimização de aspectos diretamente relacionados a parâmetros operacionais da rede, como latência, vazão ou balanceamento de carga, o objetivo é minimizar os custos financeiros de manutenção da rede, visto que esta é uma das principais preocupações sob uma visão empresarial. Sendo assim, o foco principal é a minimização dos custos com energia, que em geral é uma das principais despesas em empreendimentos relacionados com processamento massivo de dados e enlaces de longa distância. Não obstante, restrições relacionadas à latência dentro da rede e divisão de carga são também incluídas, de forma a obter um sistema financeiramente compensativo e que garanta qualidade de serviço.

Seja uma sdWAN formada por um conjunto de localidades e enlaces de controle pré-definido. Cada localidade possui um preço de energia próprio, detém um comutador e pode receber um controlador. O custo de transmissão e o tempo de propagação dentro de uma mesma localidade são aproximados em zero e os enlaces de controle podem ser utilizados apenas para fins de comunicação interna do plano de controle ou para a interface *southbound*. Deseja-se selecionar localidades nas quais serão instalados controladores e quais enlaces entre controladores e comutadores serão estabelecidos de forma a minimizar o custo financeiro dos gastos de energia. Simultaneamente, deseja-se limitar a soma de todas as latências nas conexões de controle e respeitar a quantidade máxima de requisições que cada controlador posicionado pode receber.

Pode-se ainda definir o problema de decisão inerente ao de otimização, que consiste em determinar se há algum subconjunto de localidades e de enlaces de controle que sirva para a instalação dos controladores de forma que a demanda de cada comutador seja satisfeita, a capacidade de todos os controladores seja respeitada, a soma das latências dos enlaces de controle utilizados não ultrapasse uma constante pré-definida e os custos totais com energia também sejam limitados superiormente por uma constante.

4. Modelagem do Problema

Esta seção destina-se à descrição analítica do sub-problema do *Problema de Localização de Controladores* proposto na seção 3, além da elaboração do modelo de otimização associado. Para melhor legibilidade das equações, todas as variáveis marcadas com ^{s} referem-se aos comutadores, ao passo em que todas as variáveis marcadas com ^{c} referem-se aos controladores. Os símbolos utilizados nessa seção estão sumarizados na tabela 1.

Seja a rede definida por $G = (V, A)$, em que os vértices V representem as localizações de comutadores e A os enlaces da rede que podem ser dedicados a funções de

Tabela 1. Sumário dos símbolos utilizados nessa seção e seus significados

Símbolo	Significado
V	Conjunto de vértices da rede
A	Matriz de adjacência dos possíveis enlaces de controle da rede
K_i	Preço da energia na localidade i (\$/J)
P_i	Indica a existência de um controlador ativo na localidade i
C	Matriz de adjacência dos enlaces de controle (API *southbound*)
${}^sF_i$	Frequência de envio de requisições do comutador (s^{-1})
${}^cF$	Capacidade de processamento de requisições do controlador (s^{-1})
ρ_j	Probabilidade de resposta do controlador j
E_{ij}	Custo de transmissão de uma mensagem no enlace i, j (J)
${}^cE$	Custo do processamento de uma resposta no controlador (J)
${}^sE$	Custo do processamento de uma requisição no comutador (J)
${}^cW$	Potência do controlador enquanto ocioso (W)
${}^sW_i$	Potência do comutador enquanto ocioso mais custo de encaminhamento de mensagens no plano de dados (W)
T_{ij}	RTT no enlace i, j (s)
$\hat{T}$	Soma máxima dos RTTs (s)

controle, mais precisamente à API *southbound*. Os enlaces A são representados como uma matriz binária de adjacência. A energia demandada para enviar uma mensagem pelo enlace ij aparece em E_{ij} e o ***RTT*** esperado para um par de requisição e resposta entre os elementos ij aparece em T_{ij}. O vetor K_i contém o custo da energia na localidade i, o vetor ${}^sF_i$ contém a frequência média de requisições do comutador da localidade i, o vetor ${}^sE_i$ contém a quantidade de energia gasta por requisição no comutador da localidade i e o vetor ${}^sW_i$ contém a potência gasta pelo comutador da localidade i quando ocioso somada ao gasto com o processamento e encaminhamento de mensagens do plano de dados.

O número de controladores não é definido *a priori*, porém admite-se que todos sigam o mesmo modelo, suportando uma frequência máxima de requisições de ${}^cF$, gastando uma quantidade de energia ${}^cE$ para processar cada mensagem que chega e gastando ${}^cW$ de potência quando ocioso. Admite-se ainda que a probabilidade de uma requisição gerar uma resposta no controlador j seja de ρ_j. Deseja-se determinar quais comutadores terão controladores em suas localidades e quais enlaces da interface *southbound* serão utilizados, de forma a minimizar o custo total, definido por $K_T = {}^cW \sum_{i=1}^{|V|} K_i P_i + \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} K_j C_{ij} {}^sF_i \cdot (\rho_j E_{ji} + {}^cE) + \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} K_i C_{ij} {}^sF_i \cdot (E_{ij} + {}^sE_i) + \sum_{i=1}^{|V|} {}^sW_i K_i$. P, no caso, consiste no vetor binário de posicionamento dos controladores e indica se há um controlador na localidade i. C, por sua vez, consiste na matriz de conexões de controle. Cada posição de P se torna uma variável binária em um problema de programação inteira, assim como cada posição válida de C. Uma posição de C só é válida se a mesma posição estiver assinalada na matriz A.

Nota-se que, como os comutadores são distribuídos de antemão, os únicos gastos influenciáveis pelas variáveis de decisão oriundos dos mesmos são os custos com a transmissão de requisições aos controladores. Sendo assim, os custos relacionados a comutadores ociosos, ao encaminhamento de mensagens no plano de dados e ao processamento de requisições representam constantes na função objetivo e, portanto, podem ser omitidos, gerando a função de custo $K_T = {}^cW \sum_{i=1}^{|V|} K_i P_i + \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} K_j C_{ij} {}^sF_i \cdot (\rho E_{ji} + {}^cE) + \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} K_i C_{ij} {}^sF_i \cdot E_{ij}$.

O problema envolve um conjunto de restrições de ordem linear em relação a $|V|$, sumarizadas nos itens que se seguem.

- **Restrições de conectividade.** Cada comutador se conecta a exatamente um controlador. Essa restrição pode ser implementada fazendo $\sum_{j=1}^{|V|} C_{ij} = 1 \quad \forall i$.
- **Restrições contra sobrecarga.** Sendo ${}^{c}F$ a frequência máxima de requisições suportada por cada controlador, essa restrição pode ser implementada como $\sum_{i=1}^{|V|} C_{ij} {}^{s}F_i \leq {}^{c}F \quad \forall j$.
- **Restrição de latência.** Limita superiormente a soma dos RTTs de todas as conexões de controle utilizadas a um parâmetro $\hat{T}$. Pode ser implementada como $\sum_{i=1}^{|V|} \sum_{j=1}^{|V|} C_{ij} T_{ij} \leq \hat{T}$.
- **Restrições de acoplamento.** Responsável por impedir inconsistências entre P e C. Sendo assim, uma coluna em C pode apenas ter algum valor não nulo se a posição correspondente em P estiver assinalada. Esse conjunto de restrições pode ser implementado como $\sum_{i=1}^{|V|} C_{ij} - |V| P_j \leq 0 \quad \forall j$.

5. NP-Completude

O problema de decisão é facilmente provável como NP-Completo. Em primeiro lugar, o problema está em NP, visto que um certificado (P, C) pode ser verificado aplicando as inequações das restrições e em seguida verificando se $K_T \leq K_0$ para alguma constante pré-definida K_0, o que possui complexidade total limitada a $O(|V|^2)$. Além disso, o problema é difícil, visto que o problema do *conjunto dominante de tamanho até p* é polinomialmente redutível ao mesmo, como demonstrado a seguir.

Lema 1. Seja $< G, p >$ uma instância qualquer do problema do conjunto dominante, onde G corresponde ao grafo e p corresponde ao número máximo de nós no conjunto dominante. Seja a instância do problema de localização de controladores $< G, {}^{c}W, K, E, \rho, {}^{c}E, {}^{s}F, {}^{c}F, T, \hat{T}, K_0 >$, onde ${}^{c}W_i = \rho_i = K_i = 1 \quad \forall i$, $E_{ij} = 0 \quad \forall (i,j)$, ${}^{c}E = 0$, ${}^{s}F_i = 0 \quad \forall i$, ${}^{c}F \geq 0$, $T_{ij} = 0 \quad \forall (i,j)$, $\hat{T} \geq 0$ e $K_0 = p$. Existe um conjunto dominante de tamanho até p em G se e somente se existe uma configuração válida para o problema do localização de controladores dados os parâmetros especificados.

Prova. *Qualquer escolha de enlaces de controle é viável do ponto de vista da restrição de latência, assim como qualquer configuração de (C, P) é viável em relação às restrições contra sobrecarga de controladores, visto que todas as latências e demandas de comutadores são nulas. Sendo assim, sem entrar no mérito do custo de energia, o posicionamento dos controladores em qualquer conjunto dominante de G gera uma configuração viável da versão de otimização do problema do localização de controladores, visto que a única restrição não trivial passa a ser a de que cada comutador deve ser conectado a exatamente um controlador. Cada controlador instalado gera um custo constante unitário de energia, ou seja, o custo total de energia é numericamente igual ao número de controladores. Portanto, se existir um conjunto dominante de tamanho até p em G, haverá uma configuração viável para o problema do localização de controladores com até $p = K_0$ controladores, ou seja, com custo até K_0. Da mesma forma, se existir uma solução para o problema de localização de controladores com os parâmetros definidos no lema 1, as no máximo $K_0 = p$ localidades com controladores serão um conjunto dominante de G. Como as únicas operações necessárias para a elaboração da instância do problema de posicionamento de controladores a partir da instância do problema do conjunto dominante são definições de variáveis de até duas dimensões, a transformação pode ser dita polinomial.* ■

Como o problema está em NP e em NP-Difícil, concluímos que está em NP-Completo.

Algoritmo 1: POSICIONADOR DE CONTROLADORES RELAXADO

Entrada: $< G, {}^cW, K, {}^cE, {}^sF, {}^cF >$
início
  $P = \emptyset$
  $C = \emptyset$
  $\zeta = \{{}^cF \quad$ **para cada** $\quad u \in V\}$
  enquanto existirem nós não dominados
    $\hat{u} = \max_{u \in V} \epsilon(u)$
    se $\hat{u}$ não possuir um controlador
      $P = P \cup \hat{u}$
    fim
    para cada u em $N(\hat{u})_{1..\hat{i}}$
      Seja e a aresta entre u e $\hat{u}$, faça $C = C \cup e$
      considere u dominado
      $\zeta_{\hat{u}} = \zeta_{\hat{u}} - {}^sF_u$
    fim
    se $\hat{u}$ ainda não estiver dominado
      $\zeta_{\hat{u}} = \zeta_{\hat{u}} - {}^sF_{\hat{u}}$
      considere $\hat{u}$ dominado
      Seja e a aresta de *auto-loop* de $\hat{u}$, faça $C = C \cup e$
    fim
  fim
  retorne C,P
fim

Algoritmo 2: HEURÍSTICA HGCL

Entrada: $< G, {}^cW, K, {}^cE, {}^sF, {}^cF, T, \hat{T} >$
início
  enquanto verdadeiro
    C,P = Posicionador de controladores relaxado
    $T' = 0$
    para cada aresta $e_{ij} \in C$
      $T' = T' + T_{i,j}$
    fim
    se $T' \leq \hat{T}$
      retorne C,P
    senão
      escolha aleatoriamente uma aresta problemática e
      $A = A - e$
    fim
  fim
fim

6. Algoritmos Propostos

O algoritmo exato consiste basicamente na aplicação do algoritmo de *Branch and Bound* contido no pacote *GNU Linear Programming Kit* (*GLPK*) [GNU 2012] à modelagem apresentada na seção 4. Todas as três heurísticas aqui propostas são adaptações e extensões do algoritmo aproximativo de razão $ln(|V|)$ para o problema do conjunto dominante capacitado descrito em [Kao et al. 2015]. Tal algoritmo admite demandas inseparáveis dentre os nós e busca a minimização da soma dos pesos dos nós escolhidos para o conjunto dominante, sendo provado como $O(|V|^3)$ em tempo. A primeira heurística aqui descrita basicamente aplica o algoritmo aproximativo fazendo pequenas alterações de forma a melhor compreender o problema tratado. A principal alteração consiste em um método a partir do qual a solução garantidamente obedece a restrição de latência. Tal método é baseado no princípio de que a remoção de todas as arestas de latência maior que $\hat{T}/(|V|+1)$ garante que qualquer solução seja viável no que diz respeito à limitação da latência. Isso se deve ao fato de qualquer solução ter no máximo $|V|+1$ arestas de controle, o que ocorre no caso em que há apenas um controlador servindo toda a rede. Nesse caso, a latência total quando as arestas de latência superior a $\hat{T}/(|V|+1)$ são eliminadas é limitada superiormente a $(|V|+1) \cdot \hat{T}/(|V|+1) = \hat{T}$. Soluções com um número de arestas $|C| \leq |V|+1$ terão, portanto, latência total $|C| \cdot \hat{T}/(|V|+1) \leq \hat{T}$. A segunda heurística, por sua vez, altera um pouco mais o algoritmo de obtenção do conjunto dominante, enquanto a terceira heurística modifica a segunda em relação ao método para a obtenção de uma solução que respeite a restrição da latência. As três heurísticas aparecem descritas nas subseções que

se seguem.

6.1. HGCL – Heurística Gulosa voltada ao Consumo Local

Seja $N(u)$ a lista de adjacência ainda não dominada do nó u em ordem decrescente de demandas, ou seja, de frequências de requisição. A eficiência ϵ de u é dada pela equação $\epsilon(u) = \max_{1 \leq i \leq |N(u)|} \frac{i \cdot x(u,i)}{w(u)}$. Aqui, $w(u)$ é dado pelo custo da localidade u permanecer com um controlador e servir o próprio comutador, isto é, $w(u) = K_u({}^cW + \rho \cdot {}^sF_u{}^cE)$. A função $x(u,i)$, por sua vez, retorna 1 se a capacidade residual de processamento de um eventual controlador em u suportar as demandas dos i primeiros elementos de $N(u)$ e a soma de tais demandas for maior que zero, retornando $-\infty$ caso contrário.

A heurística para encontrar uma solução para o problema do localização de controladores com relaxação na restrição de latência é sumarizada no algoritmo 1. A variável $\hat{i}$, no caso, corresponde ao valor de i para o qual a razão $i/(w(u) \cdot x(u,i))$ foi maximizada para o nó de maior eficiência $\hat{u}$. O algoritmo 1 garante que as localidades retornadas sejam um conjunto dominante e que as capacidades dos controladores sejam respeitadas, restando, portanto, apenas a restrição da latência sem garantias de satisfação. O algoritmo 2, por sua vez, garante que a solução encontrada seja viável em todos os aspectos. Ciente de que a remoção das arestas *problemáticas*, ou seja, de latência maior que $\hat{T}/(|V|+1)$, garante a satisfação da restrição da latência, o algoritmo remove uma aresta problemática qualquer por vez até encontrar uma solução viável, o que acontecerá no pior caso após todas as arestas problemáticas serem removidas. Esse algoritmo necessariamente retornará uma solução viável desde que ${}^sF_i \leq {}^cF \quad \forall i$, visto que nesse caso um conjunto dominante $P = V$ é trivialmente viável para posicionamento dos controladores. De fato, só existirão soluções viáveis caso essa condição seja satisfeita, visto que, caso contrário, nenhum controlador poderá servir o comutador com demanda superior a ${}^cF$.

A complexidade do *Posicionador de Controladores Relaxado* é $O(|V|^3)$ assim como o algoritmo aproximativo utilizado de base. A verificação da condição da latência, por sua vez, pode ser feita em $O(|A|)$. A HGCL chamará o *Posicionador de Controladores Relaxado* $O(|A|)$ vezes, o que gera uma complexidade total de $O(|A|(|V|^3+|A|))$. Como $|A| = O(|V|^2)$, é possível ignorar o custo assintótico da verificação da condição de latência, o que faz a complexidade ser equivalente a $O(|A||V|^3)$.

6.2. HGCG – Heurística Gulosa voltada ao Consumo Geral

A algoritmo da HGCG é essencialmente o mesmo do algoritmo da HGCL. Todavia, a função de eficiência passa considerar a soma das demandas a serem satisfeitas ao invés do número de nós a serem dominados, ou seja, $\epsilon(u) = \max_{1 \leq i \leq |N(u)|} x(u,i)/w(u,i) \sum_{j=1}^{i} {}^sF_{[N(u)_j]}$. Além disso, o peso dos nós passa a ser o custo energético total a ser agregado tanto com o posicionamento do controlador na localidade quanto pelo estabelecimento dos enlaces de controle. Dessa forma,
$w(u,i) = K_u{}^cW + \sum_{j=1}^{i} {}^sF_{[N(u)_j]} \left(K_u(\rho E_{[u,N(u)_j]} + {}^cE) + K_{[N(u)_j]} E_{[N(u)_j,u]}\right)$.

Analogamente à HGCL, uma solução gerada pela HGCG possui garantia de viabilidade desde que ${}^sF_i \leq {}^cF \quad \forall i$. A complexidade desse algoritmo também é a mesma da HGCL, ou seja, $O(|A||V|^3)$, visto que, mesmo tendo aspecto linear, o cálculo de $w(u,i)$ pode ser realizado com complexidade constante a partir de $w(u,i-1)$.

6.3. HGCG.2 – Heurística Gulosa voltada ao Consumo Geral (versão alternativa)

A HGCG.2 se difere da HGCG pelo método utilizado para assegurar a viabilidade no que diz respeito à restrição da latência. Ao invés de escolher uma aresta problemática qualquer, como fazem as heurísticas 1 e 2, a HGCG.2 escolhe sempre a aresta problemática de maior latência. Isso não agrega assintoticamente nenhum custo extra, o que faz com que esse algoritmo também rode em $O(|A||V|^3)$.

7. Experimentos

Foram conduzidos experimentos seguindo duas abordagens distintas, discriminadas abaixo. Na subseção 7.1, as heurísticas são avaliadas com diversas redes geradas aleatoriamente sem compromissos rígidos com redes reais, o que é utilizado para avaliar o comportamento das mesmas em casos variados, diversos e muitas vezes extremos. Para tanto, o algoritmo exato é utilizado como referencial para a qualidade das soluções. Na subseção 7.2, por outro lado, a WAN *Internet2*, largamente utilizada em trabalhos de posicionamento de controladores na literatura, é utilizada como caso de uso e simulada sobre a plataforma *ns3* [GNU 2006]. Uma versão distribuída das heurísticas HGCL e HGCG.2 foi implementada de forma a organizar a topologia em tempo de execução.

7.1. Redes geradas aleatoriamente

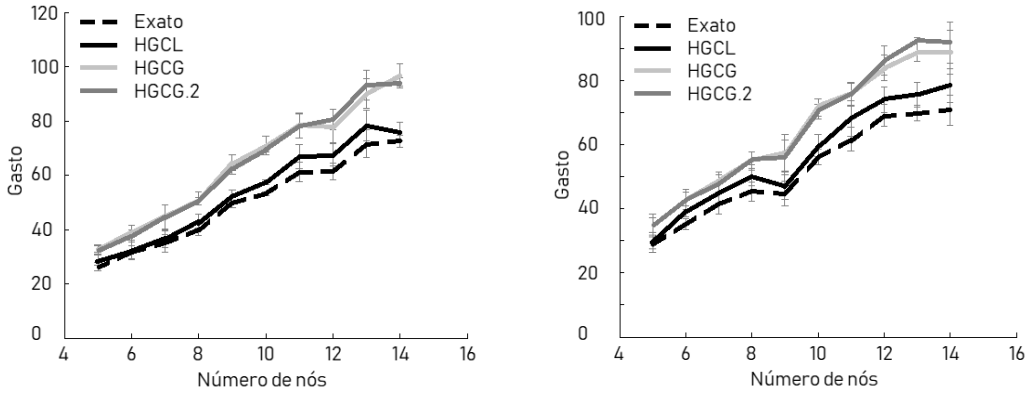
Para facilitar a inserção de parâmetros de novas redes, foi elaborado um *parser* de *XML* que transforma os parâmetros de controladores, comutadores, conexões de controle e outros mais nas matrizes necessárias para a definição da instância do problema. Tal *parser* de *XML* também admite parâmetros coringa e, nesse caso, gera uma instância aleatória que garanta a existência de ao menos uma solução viável. Nesse caso, é necessário informar apenas os limites dentro dos quais cada parâmetro deverá se manter e os números de arestas e de vértices desejados. Tal recurso foi empregado para gerar instâncias aleatórias enquanto cada parâmetro de interesse era avaliado controladamente. A tabela 2 mostra os parâmetros utilizados nos experimentos. Os valores entre colchetes correspondem aos limites utilizados para as gerações de instâncias aleatórias. A frequência máxima de requisições de um controlador é calculada para cada instância de forma a garantir viabilidade. Para tanto, é gerado um conjunto de pareamentos aleatórios entre comutadores e controladores e a frequência mínima viável para tais configurações é considerada.

Tabela 2. Parâmetros utilizados

Parâmetro	Intervalo de Valores	Unidade
Variação de longitude	10, 100	graus
Variação de latitude	10, 100	graus
Velocidade de Propagação	[150000,170000]	Km/s
Tempo de proc. por requisição (Controlador)	[0.001,0.003]	s
Tempo de proc. por requisição (Comutador)	[0.0005,0.001]	s
Energia gasta em transmissões	[0.00001,0.0001]	$J/(bit \cdot Km)$
Tamanho médio de uma mensagem	12000	bits
Frequência de requisições de um comutador	[416667,833333]	Hz
Custo	[0.0007,0.0017]	1/J
Potência do controlador quando ocioso	[400, 600]	W
Potência do comutador quando ocioso	[200, 300]	W
Energia gasta por mensagem (Controlador)	[0.005, 0.007]	J
Energia gasta por mensagem (Comutador)	[0.001, 0.003]	J

A figura 1 compara os algoritmos no que diz respeito à qualidade de suas respostas, considerando a média dos dados de cinco execuções de instâncias aleatórias em cada

experimento. No caso, são avaliadas instâncias baseadas em grafos completos e cujos nós estão espalhados ao longo de uma área de variações de latitude e de longitude iguais a 10º cada. As latências de cada aresta foram compostas com base no tempo de propagação da luz em cada enlace e no tempo de processamento das mensagens de controle em ambos os dispositivos. As distâncias consideradas foram obtidas pela aplicação da fórmula de *Haversine* [Robusto 1957] considerando, portanto, a curvatura terrestre. Os gastos aqui representados incluem constantes como os gastos dos comutadores enquanto ociosos e os gastos dos comutadores com o processamento de mensagens de controle. Não é definida uma unidade de medida simplesmente pelo fato de nenhuma moeda em específico ter sido utilizada. A figura 1(a) mostra os resultados utilizando grafos completos, enquanto a figura 1(b) utiliza apenas metade das arestas, sem considerar os *auto-loops*. As topologias foram geradas aleatoriamente a partir de um número arestas desejado. É possível perceber que em ambos os casos avaliados a HGCL se mostrou superior às demais.



(a) Grafos completos (b) Grafos com metade das arestas

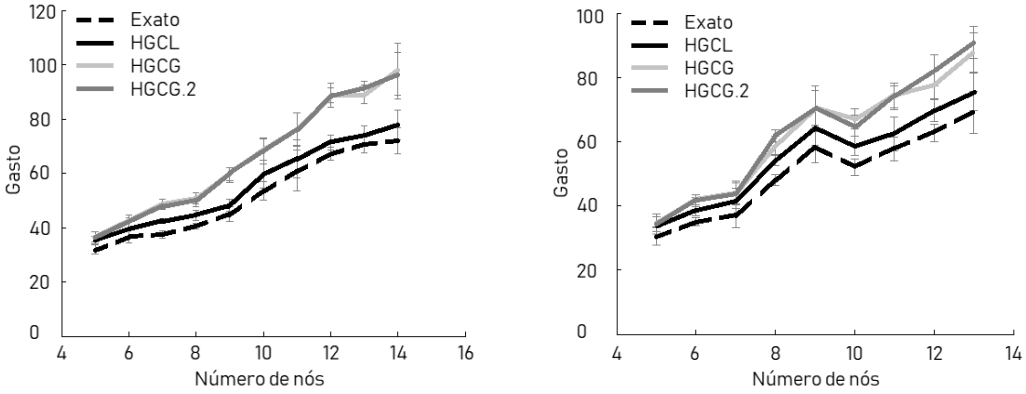
Figura 1. Gastos de energia para redes espalhadas ao longo de áreas de 10ºx10º

A figura 2, por sua vez, considera um caso muito extremo em termos de área, variando o posicionamento dos nós ao longo de uma área de variações de latitude e de longitude iguais a 100º cada. Todavia, não há alterações significativas no que diz respeito à superioridade da HGCL em relação às demais, que mantém um excedente de 5% a 10% sobre o custo ótimo.

7.2. Internet2

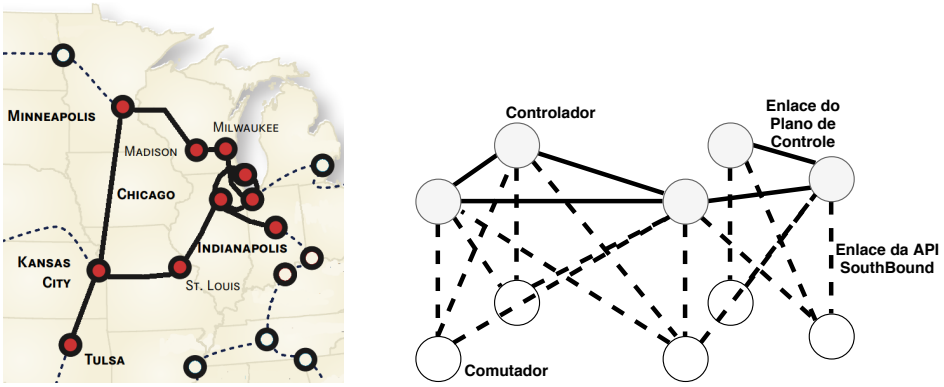
A fim de avaliar os algoritmos sob um crivo mais prático, as heurísticas HGCL e HGCG.2 foram implementadas utilizando o simulador de eventos discretos *ns3*. A topologia e os parâmetros empregados nos experimentos são oriundos do fragmento da infra-estrutura da rede acadêmica *Internet2* ilustrada na figura 3(a). Cada localidade do mapa foi traduzida em termos de abstrações do *ns3* como dois *Nodes* – um para o controlador e outro para o comutador, aqui considerados dispositivos independentes – e um enlace *PointToPoint* – representando o enlace da interface *southbound* interno à localidade. Os enlaces entre os controladores seguem as linhas contíguas demarcadas no mapa e cada enlace do plano de controle implica na existência de dois enlaces *southbound*, que ligam de forma cruzada os contro-ladores e comutadores de ambas as localidades, como mostrado na figura 3(b).

Todas as mensagens, incluindo pacotes transmitidos ao longo do plano de dados, requisições aos controladores e respostas aos comutadores, foram definidas com tamanho



(a) Grafos completos (b) Grafos com metade das arestas

Figura 2. Gastos de energia para redes espalhadas sobre áreas de 100º x 100º



(a) Mapa da sub-rede da Internet2 (enlaces contíguos) utilizada.

(b) Abstrações empregadas na simulação.

Figura 3. Ilustrações da rede simulada.

fixo de 1 Kb. As taxas de transmissão foram definidas como de 1 Gbps e as latências em cada conexão ponto-a-ponto foram definidas pela soma do tempo de transmissão e do tempo de propagação considerando cabos de fibra-ótica e as distâncias rodoviárias entre as cidades. Os custos de transmissão de cada mensagem foram também calculados a partir das distâncias rodoviárias, utilizando como base o consumo de $10^{-5}\ J/(Km \cdot bit)$. O custo energético de processamento de cada requisição no controlador foi estipulado como metade do custo de transmissão médio da rede e os dispositivos envolvidos consomem 600 W por simplesmente permanecerem ligados. Foi utilizado um gerador de tráfego sobre cada comutador de forma que se passasse uma quantidade de pacotes de dados média proporcional à densidade demográfica do estado em que cada localidade se encontra. O processamento de cada pacote gera uma requisição ao controlador com uma probabilidade variável, utilizada para modelar aplicações de diferentes naturezas. Se aplicável, a requisição é enviada ao controlador adjacente assinalado como pai do comutador em questão e respondida com probabilidade 50%. O custo do comutador por permanecer ligado, por processar cada mensagem e gerar as cabíveis requisições não foi incorporado no custo total por não depender diretamente dos resultados dos algoritmos de localização de controladores.

Cada heurística apresentada na seção 6 depende de aspectos gerais de consumo energético, da latência em cada enlace, do grafo que representa a rede e do tráfego em cada comutador. Detre todos estes parâmetros, apenas os dados de tráfego possuem uma liquidez significativamente alta, visto que os demais são diretamente relacionados com a instalação física da infra-estrutura de rede. Sendo assim, a primeira parte de ambos os algoritmos distribuídos consiste em um mecanismo de *broadcast* dos dados de tráfego, que são tratados como as demandas de cada nó pelas heurísticas. Inicialmente, todos os controladores estão ligados e servem ao comutador de sua mesma localidade. Cada controlador, então, envia seus dados de tráfego para todos os demais controladores em sua vizinhança. Em seguida, até que todos os dados sejam obtidos, a cada mensagem recebida o controlador avalia se o dado é novo ou não. Se for, o mesmo é retransmitido para todos os demais exceto seu remetente. Os enlaces de controle admitem entrega confiável, o que permite que essa fase seja concluída em $O(d(G))$, onde $d(G)$ corresponde ao diâmetro do grafo. Todavia, nós mais centrais terminarão mais cedo do que nós periféricos, o que requer cuidados especiais, como será tratado mais adiante. De posse de todos os parâmetros necessários, uma das heurísticas consideradas – HGCL ou HGCG.2 – é chamada em cada controlador. Após o término da execução, cada controlador é responsável por notificar seus filhos, que por sua vez passam a enviar as requisições para seu novo pai. Caso algum comutador fique sem pai – por, no caso, o mesmo ser eventualmente periférico e ainda não ter terminado o algoritmo de controle de topologia –, as requisições por *default* continuam sendo transmitidas para o controlador interno à localidade, que possivelmente é o próprio pai do comutador ou seria desativado. O primeiro caso trivialmente não se mostra problemático. O segundo, por sua vez, pode aumentar o consumo momentaneamente por prolongar o tempo em que o controlador permanece ativo, porém não por muito tempo. Todavia, há um caso incomum na prática, porém possível, em que o controlador permanece ativo mas não é alocado para servir ao comutador de sua localidade. Isso pode ocorrer, por exemplo, se um vizinho consistir em um vértice pendente com uma demanda muito alta e um preço de energia alto o bastante para que seja compensatório encaminhar todas as requisições para a localidade vizinha, cuja capacidade é suficiente apenas para servir este nó. Nesse caso, o controlador pode antecipar ao comutador de sua localidade qual será o seu verdadeiro pai, visto que o mesmo conhece a topologia final da rede. Além disso, pode informar ao nó todos os conjuntos de pais e filhos, de forma que, caso o verdadeiro pai ainda esteja servindo indevidamente o comutador de sua própria localidade, o mesmo possa ser redirecionado também para seu verdadeiro pai com uma cópia da topologia final e assim em diante.

Foram realizados dois conjuntos de experimentos, um limitando a soma das latências a 0,05 s e outro com o dobro desse limite. Em cada experimento, foi medido o custo financeiro em um intervalo de 100 s, variando a probabilidade de uma mensagem gerar uma requisição. Os resultados aparecem nos gráficos da figura 4. A *Baseline* referida nos gráficos consistiu em manter todos os controladores ativos. É facilmente observável que ambas as heurísticas testadas funcionam melhor quando há um baixo tráfego de controle. De fato, elas permitem que menos controladores permaneçam ativos ao longo da rede, o que gera economias no que diz respeito ao consumo fixo dos equipamentos. O aumento do tráfego de controle faz com que os gastos relacionados diretamente com as transmissões sejam mais expressivos, o que aproxima a topologia gerada pelas heurísticas da *Baseline*. A HGCL, em especial, pode gerar resultados muito desfavoráveis simplesmente por não

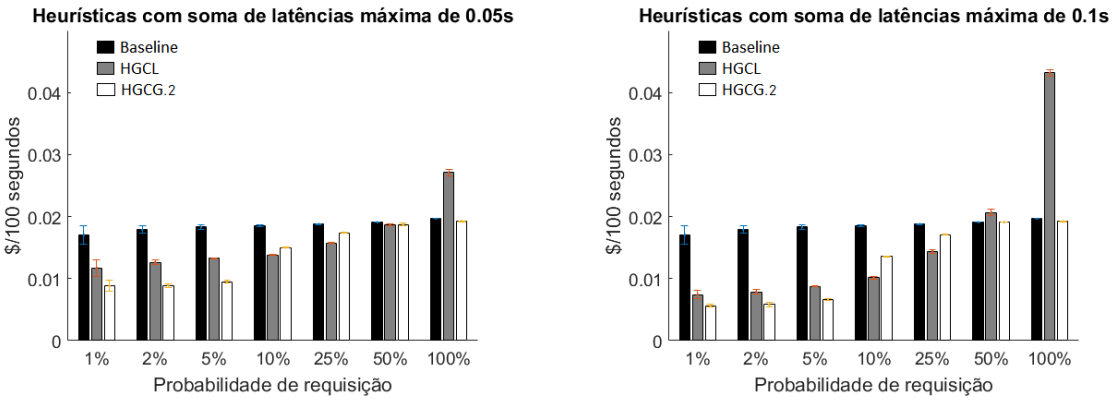


Figura 4. Gastos de energia na sub-rede da *Internet2* com diferentes limitações.

incluir a avaliação dos enlaces em sua escolha gulosa, ao contrário do HGCG.2.

Com relação à latência, o aumento do limite superior de fato causa um melhor desempenho do algoritmo, visto que uma quantidade menor de enlaces é descartada. Todavia, observa-se que esse padrão não se faz presente quando o HGCL é empregado com probabilidades de requisição de 50-100%. Isso se deve ao fato de enlaces de maior tempo de propagação também possuírem maior gasto energético de transmissão. Ao descartá-los, o HGCL deixa de escolher enlaces que poderiam gerar-lhe piores resultados.

8. Conclusões

Foi apresentada uma nova visão para o *Problema de Localização de Controladores*, visando otimizar parâmetros financeiros com garantias de qualidade de infra-estrutura de rede. A versão de decisão do problema foi provada como NP-Completa e, portanto, três heurísticas foram desenvolvidas. Dentre estas, a HGCL foi a mais eficaz em todos os cenários avaliados no primeiro conjunto de experimentos, que consideravam redes e parâmetros gerados aleatoriamente, alcançando entre 5% e 10% excedente em relação ao ótimo. As três heurísticas se mostraram equivalentes em termos assintóticos, todas rodando em $O(|A||V|^3)$. Apesar de claramente ser mais lenta do que as heurísticas, a solução exata pode ser empregada em casos com uma quantidade de nós condizente com muitas das redes que inspiraram a elaboração deste problema. Como espera-se que a execução do algoritmo seja necessária apenas no ato do projeto da rede, a utilização do algoritmo exato é altamente aconselhável, visto que haverá dinheiro envolvido. As heurísticas, por sua vez, podem ser úteis no caso em que o número de controladores é propositalmente superdimensionado de forma a adaptar a rede de forma dinâmica. Dessa forma, utilizam-se estatísticas de tráfego obtidas em tempo de execução para decidir quais enlaces e dispositivos devem permanecer ligados ou desligados a fim de gerar menos gastos financeiros. Como mostrado no segundo conjunto de experimentos, a utilização da versão distribuída das heurísticas pode gerar uma economia de até 67,2% sobre os gastos com os controladores e com a transmissão pela interface *southbound*. Nesses cenários, a heurística HGCG.2 se mostrou superior, principalmente pelo fato de evitar a escolha de enlaces muito custosos durante a construção da topologia, característica que não é diretamente avaliada pela HGCL.

Referências

[GNU 2006] GNU (2006). Ns3. www.nsnam.org. [Online; acessada 22-12-2018].

[GNU 2012] GNU (2012). Glpk: Gnu linear programming kit. www.gnu.org/software/glpk. [Online; accessada 23-05-2018].

[Guedes et al. 2012] Guedes, D., Vieira, L., Vieira, M., Rodrigues, H., and Nunes, R. V. (2012). Redes definidas por software: uma abordagem sistêmica para o desenvolvimento de pesquisas em redes de computadores. *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC 2012*, 30(4):160–210.

[Heller et al. 2012] Heller, B., Sherwood, R., and McKeown, N. (2012). The controller placement problem. In *Proceedings of the first workshop on Hot topics in software defined networks*, pages 7–12. ACM.

[Hu et al. 2014] Hu, Y., Wang, W., Gong, X., Que, X., and Cheng, S. (2014). On reliability-optimized controller placement for software-defined networks. *China Communications*, 11(2):38–54.

[Jain et al. 2013] Jain, S., Kumar, A., Mandal, S., Ong, J., Poutievski, L., Singh, A., Venkata, S., Wanderer, J., Zhou, J., Zhu, M., et al. (2013). B4: Experience with a globally-deployed software defined wan. In *ACM SIGCOMM Computer Communication Review*, volume 43, pages 3–14. ACM.

[Jouet and Pezaros 2017] Jouet, S. and Pezaros, D. P. (2017). Bpfabric: Data plane programmability for software defined networks. In *Proceedings of the Symposium on Architectures for Networking and Communications Systems*, pages 38–48. IEEE Press.

[Kao et al. 2015] Kao, M.-J., Chen, H.-L., and Lee, D.-T. (2015). Capacitated domination: Problem complexity and approximation algorithms. *Algorithmica*, 72(1):1–43.

[Koomey 2011] Koomey, J. (2011). Growth in data center electricity use 2005 to 2010. www.analyticspress.com/datacenters.html. Analytics Press. July.

[McKeown et al. 2008] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). Openflow: enabling innovation in campus networks. *ACM SIGCOMM*, 38(2):69–74.

[Robusto 1957] Robusto, C. C. (1957). The cosine-haversine formula. *The American Mathematical Monthly*, 64(1):38–40.

[Ruiz-Rivera et al. 2015] Ruiz-Rivera, A., Chin, K.-W., and Soh, S. (2015). Greco: An energy aware controller association algorithm for software defined networks. *IEEE communications letters*, 19(4):541–544.

[Wang et al. 2017] Wang, G., Zhao, Y., Huang, J., and Wang, W. (2017). The controller placement problem in software defined networking: a survey. *IEEE Network*, 31(5):21–27.

[Zhang et al. 2016] Zhang, B., Wang, X., Ma, L., and Huang, M. (2016). Optimal controller placement problem in internet-oriented software defined network. In *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2016 International Conference on*, pages 481–488. IEEE.