UNIVERSIDADE D
COIMBRA

Falhas em Cascata

Octávio do Nascimento Carvalho

Dissertação no âmbito do Mestrado Integrado em Engenharia Electrotécnica e de Computadores, da especialização em Telecomunicações, orientada pela Professora Doutora Lúcia Maria dos Reis Albuquerque Martins, apresentada ao Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Ciências e Tecnologia da Universidade de Coimbra

Setembro de 2019

UNIVERSIDADE D
COIMBRA

Falhas em Cascata

Orientador:

Prof. Doutora Lúcia Maria dos Reis Albuquerque Martins

Júri:

Prof. Doutora Rita Cristina Girão Coelho da Silva

Prof. Doutor Carlos Alberto Henggeler de Carvalho Antunes

Coimbra, Setembro de 2019

Agradecimentos

Agradeço à minha família, amigos, colegas e à minha namorada, Rafaela, por terem caminhado ao meu lado, pela paciência, pela força e carinho demonstrados ao longo da minha vida académica. Não poderia, de forma alguma, deixar passar em branco a ajuda e o incentivo prestados pela Dr^{a}. Lúcia Maria dos Reis Albuquerque Martins para a elaboração deste trabalho dissertação. Aqui lhe manifesto o meu grande obrigado por me ter acompanhado nesta jornada.

Este trabalho é financiado por Fundos FEDER através do Programa Operacional Regional do Centro e por Fundos Nacionais através da FCT - Fundação para a Ciência e a Tecnologia, I.P., no âmbito do projeto CENTRO-01-0145-FEDER-029312 e também por Fundos FEDER através do Programa Operacional Competitividade e Internacionalização - COMPETE 2020 e por Fundos Nacionais através da FCT - Fundação para a Ciência e a Tecnologia no âmbito do projeto SAICTPAC/0004/2015- POCI-01-0145-FEDER-016434.

Sumário

Mitigar os efeitos das falhas em cascata tem sido um desafio para a comunidade cientifica. Este tipo de falhas é originado num dado nó e propaga-se pelos nós adjacentes podendo resultar num colapso da rede. O presente trabalho foca-se neste tipo de falhas e foi motivado inicialmente pelo estudo da interdependência entre a rede elétrica e a rede de comunicação, que a controla (rede elétrica inteligente), embora o trabalho final não se foque especificamente nessa interdependência.

Na presente dissertação estudaram-se estratégias para identificar os nós que devem ser protegidos para limitar os efeitos de falhas com potencial para se propagarem em cascata em qualquer tipo de rede com topologia em árvore. Assim, existe a necessidade de identificar os nós críticos na rede, i. e. os nós com maior potencial de propagação de falhas, e torná-los mais robustos. Embora não se tenha estudado neste trabalho a forma de tornar robustos os nós, dado que isso depende do tipo de nós em questão e do tipo de rede, considerou-se que se um nó estiver protegido ele não vai falhar se ocorrer uma falha nos nós que lhe são adjacentes.

Para identificar os nós críticos foi estudada uma métrica de centralidade e foram desenvolvidos algoritmos baseados nesta métrica seguindo uma estratégia proposta em [5]. Basicamente a estratégia baseia-se em encontrar os nós de maior centralidade na rede em árvore e nas sub-árvores que vão surgindo à medida que esses nós centrais vão sendo protegidos.

Foi feita uma análise exaustiva do desempenho da estratégia proposta, usando redes de referência, e comparou-se o desempenho da rede na presença de falhas em cascata quando os nós críticos são escolhidos com essa estratégia e quando o mesmo número de nós críticos são escolhidos entre os nós de maior grau.

Concluiu-se que, de numa forma de geral, a estratégia proposta conduz a bons resultados, embora haja situações pontuais em que a métrica grau do nó conduz a melhores resultados.

Palavras - chave: Árvore abrangente mínima, centralidade, falhas em cascata e cami-

nhos mais curtos.

Abstract

Mitigating the effects of cascading failures has been a challenge for the scientic community. This type of failure is originated in a given node and can be propagated to adjacent nodes possibly resulting in a network collapse. The present work focuses on these types of failures and was initially motivated by the study of the interdependence between the power grid and the communication network that controls it (smart grid), although the final work doesn't specifically focus on this interdependence.

In the present dissertation, strategies to limit the effects of failures with potential to propagate in cascade in any network with tree topology were studied. Thus, there is a need to identify the critical nodes in the network, i. e. nodes with the greatest potential for failure propagation, and make them more robust. Although we didn't study how to make the nodes robust, as this depends on the type of nodes in question and the type of network, it was considered that if a node is protected it will not fail if the adjacent nodes fail.

To identify critical nodes a centrality metric was studied and algorithms based on this metric were developed following a strategy proposed in [5]. Basically the strategy is based on finding the most central nodes in the tree network and in the subtrees that appear successively after the most central nodes are protected.

An extensive analysis of the performance of the proposed strategy using reference networks was performed and the performance of the network in the presence of cascading failures was compared when critical nodes are chosen with this strategy and when the same number of critical nodes are chosen among the nodes with highest degree.

It was concluded that in general the new centrality metric leads to good results, although there are occasional situations where the node degree metric leads to better results.

Keywords: Minimum spanning tree, centrality, cascading failures and shortest paths.

Conteúdo

Capítulo 1

Introdução

Atualmente, há cada vez mais estudos sobre eficiência energética desenvolvidos para dar resposta a um consumo de energia crescente, principalmente devido à emergente utilização de veículos elétricos. É necessário melhorar os processos de medição, monitorização, gestão e controlo da rede elétrica e isso é feito com a ajuda da rede de comunicação [2]. É necessário ter em atenção a interdependência entre a rede elétrica e a de comunicação, que é fundamental para o bom funcionamento do sistema global (a rede elétrica inteligente), mas em contrapartida esta interdependência pode ser uma fonte de problemas, caso o controlo inteligente seja ineficaz. Um dos problemas associado à interdependência destas duas redes são as falhas em cascata, inicialmente ocorridas em um dado ponto de uma das redes e que desencadeiam falhas sucessivas em ambas as redes, pondo em causa o funcionamento global. Para combater este problema, necessitamos de criar uma rede inteligente robusta e fiável apostando fortemente na sua resiliência.

É de notar, que na maior parte das vezes a rede elétrica não falha por um router falhar. Esta rede é regida pelas leis de Kirchoff, e quando está bem dimensionada, as suas linhas de transmissão têm capacidade para se manterem funcionais apesar da falha na rede de comunicação. Ao falharem as comunicações falha a capacidade para atuar sobre a rede elétrica o que poderá dar origem posteriormente a problemas nesta rede. Por conseguinte, o aumento da resiliência nas redes de comunicação é indispensável para um bom controlo e monitorização das redes elétricas inteligentes e existem práticas comuns que diminuem a possibilidade de falha nestas redes aumentando assim a sua resiliência. Por exemplo, o uso de baterias assegura por tempo limitado o funcionamento de um router no caso de

falha da rede elétrica. Outra estratégia comum é utilizar mais do que uma ligação entre os nós da rede para assegurar a comunicação no caso de falha de uma das ligações. Existem outras estratégias para aumentar a resiliência das redes de comunicação, a maioria das quais saem fora do âmbito deste estudo.

1.1 Objetivo da dissertação

Nesta dissertação estudou-se uma forma de identificar os nós a proteger de modo a limitar a propagação de falhas em cascata em redes com topologia em árvore baseada numa métrica de centralidade apresentada em [4] e utilizada em [5] e [6]. Nos artigos mencionados é usado um modelo conhecido de propagação de falhas em cascata, o "Susceptible-Infected(SI) Spreading Model", que também foi usado no presente trabalho. Assim, o objetivo desta dissertação foi avaliar, da forma mais exaustiva possível, o efeito de se usar a métrica de centralidade referida para selecionar e proteger os nós de uma rede, com topologia em árvore, de modo a mitigar o efeito de uma eventual falha em cascata.

Apesar das redes elétricas inteligentes terem servido de motivação inicial para este estudo, a estratégia estudada nesta dissertação para limitar o efeito das falhas em cascata, pode aplicar-se a qualquer tipo de rede suscetível a este tipo de falhas. Existem no entanto outro tipo de aplicações como é o caso da identificação de rumores em redes sociais.

1.2 Conteúdo da dissertação

Esta dissertação é constituída pelos 5 capítulos seguintes:

Cap.1: Introdução - Capítulo introdutório que descreve o tema abordado neste trabalho de dissertação, dando uma visão global do mesmo.

Cap.2: Conceitos Básicos - Aqui são descritos os conceitos básicos relativos à representação de redes e em particular à representação de redes com topologia em árvore.

Cap.3: Falhas em Cascata - Neste capítulo é apresentada uma métrica de centralidade adequada às falhas em cascata, bem como os algoritmos baseados no uso desta métrica para a identificação ordenada dos nós mais críticos do ponto de vista da pro-

pagação deste tipo de falhas.

Cap.4: Análise de Resultados - Este capítulo contém a análise dos resultados obtidos com os algoritmos desenvolvidos assim como a comparação entre o uso da métrica de centralidade e a métrica grau do nó.

Cap.5: Conclusão - Por fim, são apresentadas as conclusões deste trabalho de dissertação bem como os possíveis trabalhos futuros que poderão ser desenvolvidos tendo como base a mesma métrica de centralidade.

Capítulo 2

Conceitos Básicos

2.1 Introdução

Neste capítulo são apresentados alguns conceitos e algoritmos que servem de base aos capítulos seguintes [1].

2.2 Representação da rede

Uma rede pode ser representada através de um grafo $G(V, E)$, onde $V = \{v_1, v_2, \ldots, v_N\}$ é o conjunto de N nós existentes na rede e E é o conjunto de arcos $E = \{e_{i,j} : i, j = 1, \ldots, N \wedge i \neq j\}$ que ligam o nó v_i ao nó v_j, com $v_i, v_j \in V$.

Nó (ou vértice) corresponde a um ponto de interconexão numa rede, com capacidade de enviar e receber informação ou disponibilizar serviços. A função do nó pode variar consoante o tipo de rede em análise. No caso das redes de comunicação, pode representar um *router*, um servidor, um *switch*, entre outros.

Arco (ou aresta) $e_{i,j}$ representa a ligação entre dois nós v_i e v_j. Quando o fluxo da informação tem um único sentido diz-se que o arco é dirigido. No caso de entre dois nós existir um arco em que o fluxo de informação circula em ambos os sentidos, diz-se que o arco é não dirigido. Neste caso é indiferente referir o arco por $e_{i,j}$ ou por $e_{j,i}$. Na figura 2.1 estão representadas duas redes, uma em que todos os arcos são dirigidos e outra com

arcos não dirigidos.

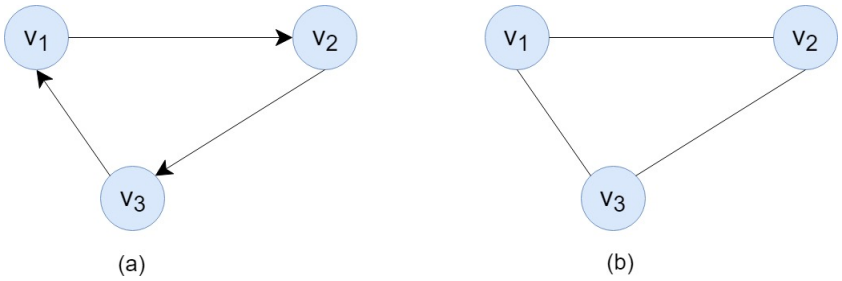

Figura 2.1: Redes com arcos dirigidos (a) e não dirigidos (b).

Os arcos podem ter um valor numérico associado dizendo-se nesse caso que são ponderados, caso contrário diz-se que são não ponderados. Esse valor numérico $c_{i,j}$ (associado ao arco $e_{i,j}$), pode corresponder ao custo, à distância, à capacidade, entre outros.

Grau do nó é uma medida inerente ao nó v_i é o grau d_i, que corresponde ao número de nós que lhe são adjacentes, i. e. ao número de nós a que está ligado através de arcos incidentes ou divergentes. No caso de uma rede não dirigida o grau do nó v_i é dado pela cardinalidade do conjunto $\{e_{i,j} : j = 1, \ldots, N \ \wedge \ e_{i,j} \in E\}$.

Trajeto é uma sequência finita e alternada de nós e arcos. No entanto, para definir um trajeto não é obrigatório mencionar os nós e os arcos. Basta usar só um deles, como se mostra a seguir.

Trajeto dirigido é um tipo de trajeto que só pode conter arcos dirigidos, como representado na figura 2.2.

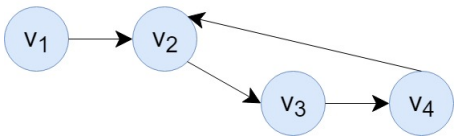

Figura 2.2: Trajeto dirigido.

O trajeto pode então ser definido de três formas diferentes, tal como referido anteriormente:

- $v_1 - e_{1,2} - v_2 - e_{2,3} - v_3 - e_{3,4} - v_4 - e_{4,2} - v_2$;

- $v_1 - v_2 - v_3 - v_4 - v_2$;
- $e_{1,2} - e_{2,3} - e_{3,4} - e_{4,2}$;

Caminho (dirigido ou não dirigido) é definido como um trajeto sem repetição de nós.

Ciclo é uma sequência de nós e arcos que define um caminho que é iniciado num certo nó mais o arco que liga o último nó do caminho ao primeiro. Podemos utilizar a seguinte notação para definir um ciclo: $v_1 - v_2 - ... - v_{i-1} - v_i - v_1$. Ao observar a figura 2.2, podemos identificar um ciclo na sequência: $v_2 - v_3 - v_4 - v_2$.

Tipos de grafos

Um grafo pode ter as seguintes características:

- dirigido ou não dirigido, dependendo do tipo de arco que o constitui;
- cíclico ou acíclico, conforme exista pelo menos um ciclo ou não;
- ponderado ou não ponderado, caso os seus arcos tenham um valor numérico associado ou não;
- conexo ou desconexo, quando todos os nós estão interligados entre si (isto é, para qualquer par de nós existe pelo menos um caminho) ou não, respetivamente.

Nas figuras 2.3 e 2.4 são dados dois exemplos de grafos.

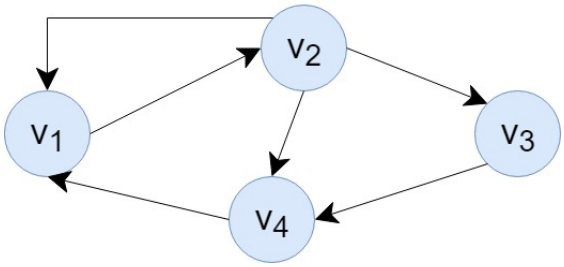

Figura 2.3: Grafo dirigido, cíclico e não ponderado.

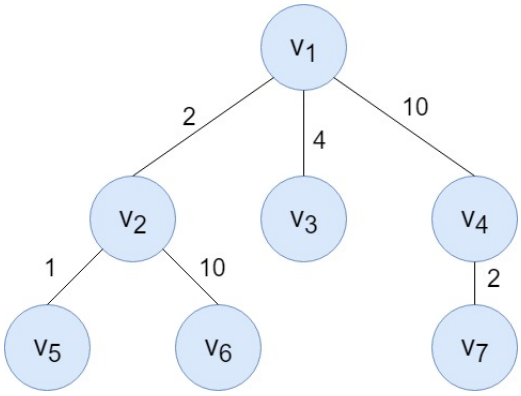

Figura 2.4: Grafo não dirigido, acíclico e ponderado.

Lista de adjacência é usada para representar um grafo G e é representada por A em que $A = \cup_{i=1}^{N} A(v_i)$ sendo $A(v_i)$ o conjunto de arcos adjacentes ao nó v_i dado por $A(v_i) = \{e_{i,j} : j = 1, \ldots, N \ \wedge \ e_{i,j} \in E\}$. Na figura 2.5, podemos observar uma ilustração da lista de adjacência do grafo da figura 2.4. É de notar que A é um vetor de listas ligadas, em que cada lista representa os arcos adjacentes ao nó v_i e os custos $c_{i,j}$ de cada arco.

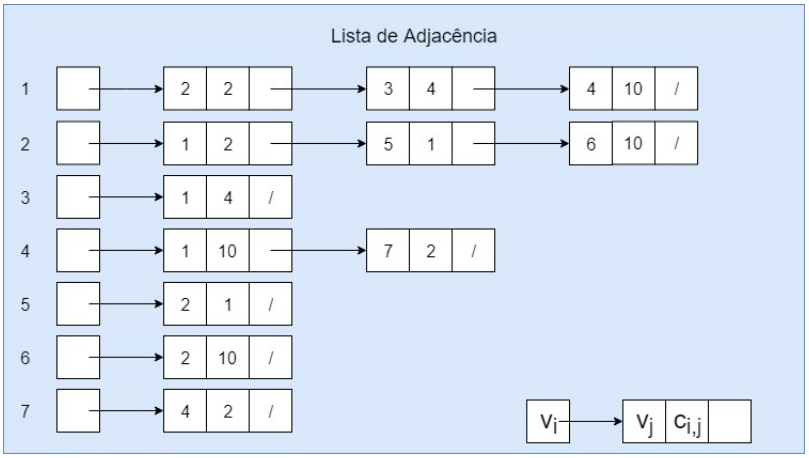

Figura 2.5: Lista de adjacências correspondente à rede na figura 2.4

Árvore T é um grafo acíclico e conexo, com M nós e $M-1$ arcos e, entre cada dois dos seus nós constituintes existe somente um caminho. No contexto deste trabalho as árvores são sempre grafos não dirigidos. Um exemplo de uma árvore é o grafo representado na figura 2.4.

Árvore abrangente é uma árvore que contém todos os nós que constituem de um dado grafo G interligados pelo menor número possível de arcos. A árvore abrangente mínima, pode ser obtida por exemplo pelo algoritmo de Prim [1], em que o somatório dos pesos dos seus arcos é o menor possível.

Árvores com raiz são árvores que têm um nó particular que é designado por nó raiz. A raiz é o único nó de uma árvore que não tem predecesso. As suas ligações são somente com os seus sucessores através de um único caminho. Na figura 2.4, podemos observar que o nó v_1 é raiz sendo predecessor dos nós v_2, v_3 e v_4, e estes são consequentemente sucessores do nó v_1 (também designados por $child(v_1) = \{v_2, v_3, v_4\}$). Já na ligação entre o nó v_4 e v_7, o nó v_4 é predecessor do v_7, o v_7 é sucessor do v_4, e o v_4 e v_7 são descendentes do nó v_1. Os nós descendentes do nó v_1 são os seus sucessores, e assim sucessivamente.

Sub-grafo $G' = (V', E')$ é um sub-grafo de $G = (V, E)$ se $V' \subseteq V$ e $E' \subseteq E$.

Sub-árvore é um sub-grafo de um grafo com topologia em árvore. No caso de ser retirado um nó, o número de sub-árvores criadas é igual ao número de nós adjacentes a esse mesmo nó. Ao considerar que foi retirado o arco $e_{i,j}$ entre os nós adjacentes $v_i, v_j \in T$ são geradas duas sub-árvores. Para que seja possível diferenciar as diversas sub-árvores que possam surgir foi utilizada a notação de [4], em que T_x^y é uma sub-árvore com raiz em v_x, proveniente da árvore T com raiz em v_y. O tamanho de T_x^y, ou seja o número de nós de T_x^y, é representado por t_x^y.

Para ilustrar esta notação, podemos observar um simples exemplo na figura 2.6. É de notar que $t_2^1 = 3$ porque existem 3 nós na sub-árvore, o nó v_2 é raiz de T_2^1 e o nó v_1 é raiz de T. Já $t_7^1 = 1$, porque só existe um único nó na sub-árvore T_7^1 que tem como raiz v_7 e é gerada a partir de T com raiz em v_1.

Nós folha são nós que têm grau 1 e não têm sucessores.

Árvore binária é uma árvore onde cada nó pode conter somente 0, 1 ou 2 sucessores. Dentro das árvores binárias existem as árvores estritamente binárias em que todos os nós que não são folhas têm dois sucessores, esquerdo e direito. A profundidade de um nó

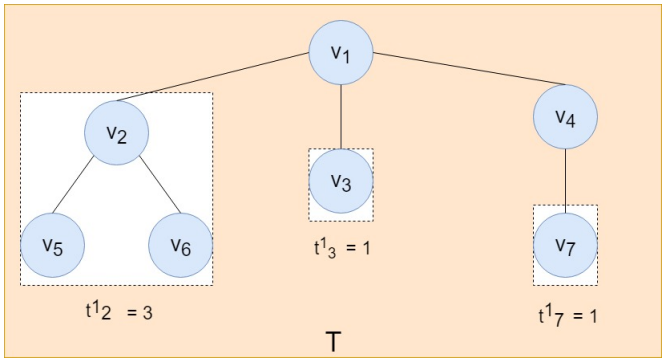

Figura 2.6: Notação para sub-árvores.

corresponde à distância entre esse nó e a raiz, sendo que os nós com a mesma profundidade são nós que estão ao mesmo nível na hierarquia. No nível 0 está a raiz e os seus sucessores encontram-se no nível 1, e assim sucessivamente. A altura da árvore é o seu nível máximo (ou profundidade máxima), ou seja, é a distância máxima entre a raiz e as folhas.

2.3 *Heap*

Uma *heap* é uma estrutura de dados utilizada para melhorar o desempenho de diversos algoritmos devido à sua capacidade de armazenar e manipular dados eficientemente. Na sua forma binária recorre a uma árvore binária que é preenchida da esquerda para a direita. Todos os nós constituintes têm um valor real associado designado por chave que possibilita trocas de posições entre os mesmos.

Esta estrutura de dados pode ser máxima ou mínima, dependendo da utilidade dada à mesma. Para uma *heap* máxima a raiz é o nó que contém o valor chave mais elevado, no caso da *heap* mínima é o contrário. Neste trabalho iremos utilizar a *heap* mínima em que para cada nó existe uma relação de ordem predecessor-sucessor. Neste caso a chave do predecessor é menor ou igual à chave dos seus sucessores.

Numa *heap* as operações exequíveis são as seguintes [1]:

- **cria(H):** Cria uma *heap* H vazia.
- **insere(v_i,H):** Insere o nó v_i na *heap* H.
- **encontra-min(v_i,H):** Encontra o nó v_i com chave mínima na *heap* H.

- **apaga-min(v_i,H):** Apaga o nó v_i com chave mínima na *heap* H.

- **diminui-chave(valor,v_i,H):** Diminui o valor da chave do nó v_i, substituindo por *valor* e de seguida restaura a ordenação na árvore binária.

A *heap* aumenta a eficiência do cálculo do caminho mais curto através do algoritmo de Dijkstra e a eficiência do algoritmo de Prim na obtenção da árvore abrangente mínima.

2.4 Algoritmo de Dijkstra

O algoritmo de Dijkstra permite encontrar o caminho com menor custo entre um dado nó e todos os outros considerando os custos dos arcos não negativos. Para realizar as operações é necessário receber como entrada o nó v_s para iniciar o cálculo dos caminhos mais curtos desde esse nó para todos os outros. Para cada nó $v_i \in G$, o algoritmo guarda uma chave ($chave(v_i)$) que corresponde a um limite superior do custo do caminho mais curto (ou seja, o caminho de menor custo) até chegar ao nó v_s. Inicialmente, é atribuído ao nó v_s uma chave e predecessor nulos ($chave(v_s) = 0$ e $pred(v_s) = 0$) e a todos os outros nós $v_i \in V$ uma chave infinita ($chave(v_i) = \infty$). À medida que o algoritmo vai decorrendo os predecessores vão sendo atualizados, assim como as chaves dos nós para um valor representativo do custo do caminho mais curto desde o nó v_s até ao nó v_i. Como tal, é selecionado um nó v_i com chave mínima e são analisados os seus arcos para atualizar a chave dos nós adjacentes v_j. O algoritmo termina quando não houver mais custos para atualizar (i. e. a *heap* está vazia). No algoritmo 1 apresenta-se o pseudo-código correspondente ao algoritmo de Dijkstra [1].

Algoritmo 1 DIJKSTRA USANDO UMA *heap*

Entrada: v_s

Saída: T^{v_s} – Árvore dos caminhos mais curtos do nó v_s para todos os outros nós da rede.

início

- $T^{v_s} = \emptyset$
- cria a *heap*(H)
- chave(v_j)= ∞ *para todo* $v_j \in V$
- chave(v_s)= 0
- pred(v_s)=0
- insere(v_s,H)
- **enquanto** $H \neq \emptyset$ **faça**
  - encontra-min(v_i,H)
  - apaga-min(v_i,H)
  - $T^{v_s} = T^{v_s} \cup (pred(v_i), v_i, chave(v_i))$
  - **para cada** $e_{(i,j)} \in A(i)$ **faça**
    - $valor := chave(v_i) + c_{ij}$
    - **se** $chave(v_j) > valor$ **então**
      - **se** $chave(v_j) := \infty$ **então**
        - chave(v_j)= valor
        - pred(v_j)=v_i
        - insere(v_j,H)
      - **fim**
      - **senão**
        - chave(v_j)=valor
        - pred(v_j)=v_i
        - diminui-chave(valor,v_j,H)
      - **fim**
    - **fim**
  - **fim**
- **fim**

fim

Como exemplo de funcionamento deste algoritmo, podemos observar a figura 2.7 e

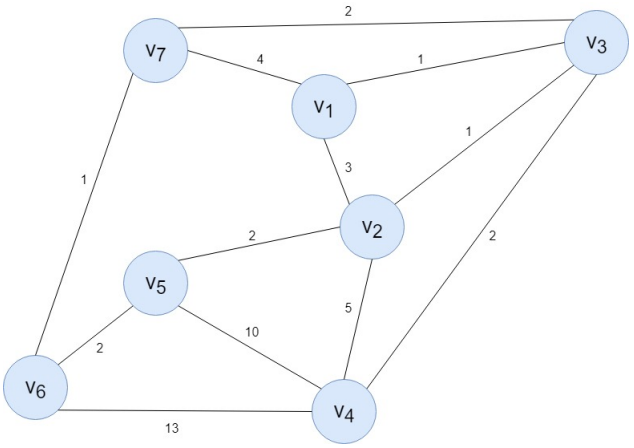

Figura 2.7: Rede 1

obter o caminho do nó v_1 até ao nó v_5. Inicialmente, ao ser considerado o nó v_1 como partida, a sua chave é declarada nula porque para obter este nó não há qualquer custo devido a estarmos a começar por ele. Todos os outros nós da rede, ainda nesta fase inicial, são colocados com chave infinita. O algoritmo atualiza as chaves dos nós através da expressão: $valor = chave(v_i) + c_{ij}$, para qualquer nó v_j adjacente a v_i. Tendo $chave(v_j) > valor$, se $chave(v_j) = \infty$ o valor de $chave(v_j)$ terá de ser atualizado para $chave(v_j) = valor$ e v_j inserido na *heap*. Se $chave(v_j) \neq \infty$ e $chave(v_j) > valor$, o valor da sua chave passa a ser $chave(v_j) = valor$ e, é atualizada a sua chave na *heap* através do método $diminui{-}chave(valor, v_j, H)$. Começando, por exemplo, no nó v_7, em que $chave(v_7) = \infty$, neste caso o valor do peso do arco é 4, ou seja a chave do nó v_7 ao ser atualizada fica $chave(v_7) = 4$, e este valor é inserido na *heap*. De igual forma são inseridos os nós v_2 e v_3 com chaves 3 e 1, respetivamente. Depois de terem sido analisados todos os sucessores de v_1 é retirado da *heap* o nó com chave mínima. Neste caso o nó v_3. De seguida são analisados todos os nós adjacentes a v_3, ou seja, v_7, v_2 e v_4. Como v_4 é o único com chave igual a infinito a nova chave é $chave(v_4) = chave(v_3) + c_{3,4}$, logo $chave(v_4)$ é igual a 3 e este valor é inserido na *heap*. Como os nós v_2 e v_7 estão na *heap* com chave 3 e 4, respetivamente estas chaves, $chave(2)$ e $chave(7)$, são agora atualizadas para 2 e 3 respetivamente, através do método $diminui - chave$. De seguida, é retirado da *heap* o nó v_2 dado que é o nó com chave mínima. O algoritmo prossegue da mesma forma até serem retirados todos os nós da *heap*. Na tabela 2.1 está representada a árvore dos caminhos mais curtos do nó v_1 para todos os outros.

v_i	Pred(v_i)	chave(v_i)
v_1	0	0
v_3	v_1	1
v_2	v_3	2
v_7	v_3	3
v_4	v_3	3
v_5	v_2	4
v_6	v_7	4

Tabela 2.1: Árvore de caminhos mais curtos para a rede na figura 2.7

Neste exemplo, assim que é obtido o predecessor do nó v_5 conseguimos encontrar o caminho mais curto até ao nó v_1, através dos outros predecessores: $pred(v_5) = v_2$; $pred(v_2) = v_3$; $pred(v_3) = v_1$. Ou seja, o caminho mais curto do nó v_1 ao nó v_5 é: $v_1 - v_3 - v_2 - v_5$. Já para obter a distância mínima do caminho mais curto entre estes dois nós basta consultarmos $chave(v_5)$, que neste caso é igual a 4.

2.5 Algoritmo de Prim

O algoritmo de Prim foi desenvolvido para obter uma árvore abrangente mínima T a partir de qualquer nó $v_i \in G$. Durante a execução do algoritmo a sub-árvore abrangente vai sendo constituída por um subconjunto de nós S e pelos arcos que os conectam. Para adicionar novos nós a esta sub-árvore é necessário encontrar o arco com menor custo que interliga S com um nó adjacente v_j a este subconjunto. Assim, os arcos vão sendo adicionados à sub-árvore e os nós a S, até que S contenha todos os nós do grafo em análise.

Para iniciar o algoritmo é necessário escolher aleatoriamente um nó $v_i \in G$. A soma dos arcos da árvore abrangente mínima terá o mesmo valor. No algoritmo 2 é apresentado o pseudo-código corresponde ao algoritmo de Prim [1].

O resultado do algoritmo de Prim deverá apresentar uma árvore abrangente mínima T que contém todos os nós presentes no grafo G, com $N - 1$ arcos.

Ao iniciar este algoritmo num dado nó v_i, as ligações com os nós adjacentes são analisadas e é escolhida a que tiver menor custo (ou chave). Considerando a rede na figura 2.7 e começando no nó v_1, a ligação com menor custo é com o nó v_3 (com uma chave igual a 1), $e_{1,3}$, logo v_3 é o segundo nó inserido em S. S contém agora dois nós v_1 e v_3, logo o próximo passo é analisar os custos dos arcos que interligam aos nós adjacentes a este subconjunto. De seguida identifica-se o arco $e_{3,2}$ com menor custo, logo v_2 é o terceiro nó a ser inserido em S. Seguidamente são colocados v_4, v_5, v_7 e, por último, v_6. Quando o conjunto S está completo, i. e., contém todos os nós existentes em G, então a árvore abrangente T está completa. Deste modo a árvore abrangente mínima, T é constituída pelos seguintes arcos: $e_{1,3}$ - $e_{3,2}$ - $e_{3,4}$ - $e_{2,5}$ - $e_{3,7}$ - $e_{7,6}$. É de notar que os custos de $e_{2,5}$ e $e_{3,7}$ são iguais ao custo de $e_{5,6}$, logo a árvore abrangente final poderia conter o arco $e_{5,6}$ em vez de um dos dois arcos de igual custo mencionados. Pode-se concluir que neste caso existem várias árvores abrangentes mínimas que são ótimos alternativos.

Algoritmo 2 PRIM USANDO UMA *heap*

Entrada: G
Saída: T
início
    cria a *heap*(H)
    para cada $v_j \in V \setminus \{v_1\}$ **faça**
        chave(v_j)= C+1
        // C representa o custo máximo dos arcos contidos em T
    fim
    chave(v_1)= 0
    pred(v_1)=0
    para cada $v_j \in V$ **faça**
        insere(v_j,H)
    fim
    $T = \emptyset$
    enquanto $|T| < (N-1)$ **faça**
        // $|T|$ representa o número de arcos constituintes de T
        encontra-min(v_i, H)
        apaga-min (v_i, H)
        $T = T \cup (pred(v_i), v_i, chave(v_i))$
        para cada $e_{i,j} \in A(i)$ *com* $v_j \in H$ **faça**
            se *chave*(v_j)> c_{ij} **então**
                chave(v_j)=c_{ij}
                pred(v_j)=v_i
                diminui-chave(c_{ij}, v_j, H)
            fim
        fim
    fim
fim

Capítulo 3

Falhas em cascata

3.1 Introdução

Neste capítulo é apresentada uma métrica de centralidade que permite identificar o nó mais crítico do ponto de vista das falhas em cascata, numa rede com topologia em árvore. Esta métrica permite assim identificar o nó que deve ser protegido de modo a evitar falhas em cascata com maior impacto.

Para modelar falhas em cascata nas infraestruturas de rede, foi assumido um modelo epidémico básico conhecido por "Susceptible-Infected(SI) Spreading Model"[3]. Neste modelo, existem dois tipos de nós: i) nós suscetíveis a falhas e ii) nós afetados que podem causar falhas aos seus adjacentes. Se falhar um nó suscetível a falhas, ele permanecerá nesse estado perpetuamente, afetando os outros nós suscetíveis próximos e desencadeando assim falhas em cascata.

3.2 Métrica de Centralidade

Considere-se que numa árvore T com raiz v_y o número de permutações possíveis para os N nós existentes, sendo que o primeiro elemento da permutação é a raiz da árvore, é dado por $R(v_y, T)$.

$R(v_y, T)$ é uma medida de centralidade do nó v_y na árvore T. Esta centralidade é apresentada em [4] como sendo *rumor centrality* e foi usada para identificar fontes de rumores em redes podendo ter ainda outros tipos de aplicações tais como, por exemplo,

identificação de fontes de propagação de vírus na Internet e identificação da origem de falhas em cascata. É no contexto de limitar o efeito de uma falha em cascata que esta métrica é usada em [5] e [6] permitindo escolher os nós de maior centralidade para serem protegidos evitando assim falhas em cascata de maiores dimensões.

Seja T_x^y a sub-árvore de T em que o nó v_y é raiz de T e $v_x \in child(v_y)$ é raiz dessa sub-árvore. Então $R(v_x, T_x^y)$ representa o número de permutações de nós iniciados em v_x na sub-árvore T_x^y. Existe uma relação recursiva natural entre $R(v_y, T)$ e o número de permutações nas sub-árvores resultantes dos seus sucessores $R(v_x, T_x^y)$ [4]:

$$R(v_y, T) = (N-1)! \prod_{v_x \in child(v_y)} \frac{R(v_x, T_x^y)}{t_x^y!} \tag{3.1}$$

Conhecendo os tamanhos das sub-árvores t_x^y , ao prosseguir com os cálculos até alcançar o nó raiz iremos encontrar o número total de permutações permitidas em T e iniciadas em v_y:

$$\begin{aligned} R(v_y, T) &= (N-1)! \prod_{v_x \in child(v_y)} \frac{(t_x^y - 1)!}{t_x^y!} \prod_{v_w \in child(v_x)} \frac{R(v_w, T_w^y)}{t_w^y!} \\ &= (N-1)! \prod_{v_x \in child(v_y)} \frac{1}{t_x^y} \prod_{v_w \in child(v_x)} \frac{R(v_w, T_w^y)}{t_w^y!} \\ &= (N-1)! \prod_{v_x \in T \setminus v_y} \frac{1}{t_x^y} \\ &= N! \prod_{v_x \in T} \frac{1}{t_x^y} \end{aligned} \tag{3.2}$$

É de notar que $t_y^y = N$ é o tamanho de T.

De modo a relacionar a centralidade de dois nós adjacentes considere-se duas árvores com duas raízes diferentes obtidas a partir da mesma árvore inicial. Na figura 3.1 à esquerda está representada uma árvore com raiz em v_1 e à direita a mesma árvore com raiz em v_2. Pode verificar-se que os nós v_2 e v_1 são adjacentes e que $t_2^1 = N - t_1^2$, sendo t_1^2 a sub-árvore com raiz em v_1 da árvore com raiz em v_2 e T_2^1 a sub-árvore com raiz em v_2 da árvore com raiz em v_1. Note que $v_4 \in child(v_2)$ em ambas as árvores logo $t_4^1 = t_4^2$.

Sendo v_y adjacente de v_x em T então:

$$\frac{R(v_x, T)}{R(v_y, T)} = \frac{t_x^y}{t_y^x} = \frac{t_x^y}{N - t_x^y} \tag{3.3}$$

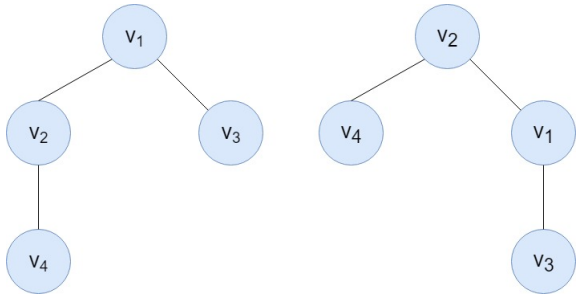

Figura 3.1: Árvore T para análise

Para exemplificar o cálculo da centralidade de um nó vai-se considerar o nó v_1 na árvore da esquerda na figura 3.1:

$$R(v_1, T) = \frac{4!}{t_4^1 \times t_3^1 \times t_2^1 \times t_1^1} = \frac{4!}{1 \times 1 \times 2 \times 4} = 3 \tag{3.4}$$

Existem portanto 3 permutações permitidas em T considerando o nó v_1 como raiz: $\{v_1, v_2, v_3, v_4\}$, $\{v_1, v_3, v_2, v_4\}$, $\{v_1, v_2, v_4, v_3\}$.

3.3 Nó central

O nó de maior centralidade, isto é o nó com maior valor de $R(v, T)$, é designado por nó central ou centróide (*rumor center* em [4]).

Se o nó central numa árvore T com N vértices for o nó v_c , então prova-se [4] que qualquer sub-árvore com raiz em v_x adjacente a v_c, tem a seguinte propriedade:

$$t_x^c \leq \frac{N}{2} \tag{3.5}$$

Prova-se que numa rede em árvore pode existir no máximo 2 nós centrais.

É de notar que um centróide é um nó que ao ser removido, divide uma rede em árvore em sub-árvores, de maneira que cada uma delas tenha no máximo metade do número de nós da árvore original.

3.4 Centralidade em termos de distância

Para uma árvore T a centralidade em termos de distância de $v_y \in T$ é dada por $D(v_y, T) = \sum_{v_x \in T} d(v_y, v_x)$, onde $d(v_y, v_x)$ é a distância do caminho mais curto entre os

nós v_y e v_x.

O centro de distância de uma árvore é o nó v_y mais perto de todos os outros, ou seja, é o que tem menor valor de $D(v_y, T)$. Prova-se que numa árvore T o nó central ou centróide é também o centro de distância, se for único.

3.5 Árvore centróide

Para construir uma árvore centróide T_c de uma árvore T é necessário obter o centróide dessa árvore e considerar de seguida as sub-árvores resultantes da remoção desse centróide da árvore original. De seguida vão ser encontrados novos centróides nessas sub-árvores que darão origem a sub-árvores de sub-árvores e assim sucessivamente. Assim a construção da árvore centróide T_c segue os passos seguintes:

- Obter o centróide da árvore T e considerá-lo como raiz da árvore T_c.
- Obter as sub-árvores resultantes da remoção dos centróides que vão sendo obtidos.
- Colocar os centróides dessas sub-árvores em T_c como sucessores dos centróides que deram origem a essas sub-árvores.

No capítulo 4 é ilustrada a obtenção da árvore centróide T_c.

3.6 Algoritmo de passagem de mensagens

O cálculo da centralidade para todos os nós da rede pode ser realizado de forma eficiente através do algoritmo de passagem de mensagens. É de notar que este algoritmo de passagem de mensagens foi baseado no algoritmo apresentado em [4] tendo sido corrigido dado que tal como está escrito na referência mencionada está incorreto.

O algoritmo de passagem de mensagens é constituído por duas partes de cálculo distintas, uma consiste no cálculo do tamanho das sub-árvores com raiz em $v_i \in T$ com $i = 1, .., N$ e outra na obtenção da centralidade para cada nó v_i, como pode ser observado no algoritmo 3.

Algoritmo 3 PASSAGEM DE MENSAGENS

Entrada: T
Saída: $C(T)$ – Centralidade de todos os nós de T
v_v = raiz de T

para $v_u \in T \setminus v_v$ **faça**

  se v_u *é uma folha* **então**

    $t^{up}_{u \to pred(u)} = 1$;

    $p^{up}_{u \to pred(u)} = 1$;

  fim

  senão

    $t^{up}_{u \to pred(u)} = \sum_{v_j \in child(v_u)} t^{up}_{j \to u} + 1$;

    $p^{up}_{u \to pred(u)} = t^{up}_{u \to pred(u)} \prod_{v_j \in child(v_u)} p^{up}_{j \to u}$;

  fim

fim

para $v_u \in T$ **faça**

  se v_u *é raiz* v_v **então**

    $\forall v_{v'} \in child(v_v) c^{down}_{v \to v'} = \frac{N!}{N \prod_{v_j \in child(v_v)} p^{up}_{j \to v}}$;

  fim

  senão

    $\forall v_{u'} \in child(v_u) c^{down}_{u \to u'} = c^{down}_{pred(u) \to u} \frac{t^{up}_{u \to pred(u)}}{N - t^{up}_{u \to pred(u)}}$;

  fim

fim

As mensagens que são passadas dos sucessores para predecessores são definidas por "up", e no sentido contrário são consideradas mensagens "down". Existem 2 tipos de mensagens "up", a $t^{up}_{u \to pred(u)}$ que informa o predecessor de v_u ($pred(v_u)$) sobre o tamanho da sub-árvore com raiz em v_u, e a $p^{up}_{u \to pred(u)}$ que informa o predecessor de v_u sobre o produto dos tamanhos de todas as sub-árvores geradas pelos descendentes de v_u.

É necessário então calcular primeiro o tamanho das sub-árvores, sendo mais fácil começar pelos nós folha e prosseguir até à raiz de T. Tendo o tamanho das sub-árvores podemos calcular a centralidade de todos os nós, começando pela raiz, e transmiti-la aos seus sucessores através da mensagem $c^{down}_{u \to u'}$ com $v_{u'} \in child(v_u)$.

Assim que todas as centralidades forem calculadas, iremos comparar os valores obtidos

para cada nó. O nó com maior centralidade é designado por nó central ou centróide.

3.7 Algoritmo para identificação dos nós que devem ser protegidos para limitar o efeito de falhas em cascata

Nesta secção são apresentados os dois algoritmos baseados em procedimentos propostos em [6]. O algoritmo 4 tem como função obter a árvore centróide T_c a partir dos centróides da árvore T e das sub-árvores T_j resultantes da decomposição da árvore original, sendo que j corresponde ao índice da sub-árvore.

Algoritmo 4 Construção da árvore centróide

Entrada: T
Saída: T_c
início
  $T_c = \emptyset$
  $C(T)$=Passagem de Mensagens (T);
  v_c = Deteção do centróide $(C(T))$;
  $\mathcal{T} = \{T_j\}$ = Decomposição da Árvore (T, v_c);
  $T_c = T_c \cup \{v_c\}$
  enquanto $\mathcal{T} \neq \emptyset$ **faça**
    se $|T_j| = 1$ **então**
      $T_c = T_c \cup T_j$
    fim
    senão
      $C(T_j)$=Passagem de Mensagens (T_j);
      v_c = Deteção do centróide $(C(T_j))$;
      $\mathcal{T} = \mathcal{T} \cup$ Decomposição da Árvore (T_j, v_c);
      $T_c = T_c \cup \{v_c\}$
    fim
    $\mathcal{T} = \mathcal{T} \setminus T_j$
  fim
fim
retorna T_c

Descrevem-se de seguida dois métodos utilizados no algoritmo 4:

Deteção do centróide - Após obtermos a centralidade de cada nó, este método identifica o nó de maior centralidade como sendo o nó centróide v_c.

Decomposição da árvore - Este método identifica as novas sub-árvores T_j resultantes da decomposição de uma árvore depois de retirado o respetivo centróide e adiciona-as ao conjunto de sub-árvores $\mathcal{T}$. As variáveis de entrada são uma árvore e o centróide anteriormente calculado. O número de sub-árvores criadas é igual ao número de nós

adjacentes ao centróide.

Finalmente o último algoritmo é complementar do algoritmo [4] e tem como objetivo a obtenção do conjunto de nós a proteger, V_p, depois de conhecida a árvore centróide. Neste algoritmo começa-se por calcular o tamanho de todas as sub-árvores de T_c seguindo a mesma estratégia que foi utilizada no algoritmo [4]. De seguida os nós são ordenados em função do tamanho da respetiva sub-árvore. V_p é então preenchido com os k primeiros nós nessa ordenação, sendo k o número de nós que se pretende proteger.

Capítulo 4

Análise de resultados

4.1 Introdução

Neste capítulo ir-se-á analisar os resultados obtidos pelo algoritmo para identificação dos nós que devem ser protegidos para limitar o efeito de falhas em cascata para redes com topologia em árvore. As redes utilizadas nos testes realizados são esparsas e divididas por conjuntos consoante o número de nós e de arcos que as constituem. As redes escolhidas foram todas as redes do conjunto B e as dez primeiras redes do conjunto C que podem ser encontradas em *http://steinlib.zib.de/testset.php*. O conjunto das redes B é constituído por 18 redes, 6 de 50 nós, 6 de 75 nós e 6 de 100 nós. Em relação às redes C todas elas são formadas por 500 nós. Como a estrátegia desenvolvida aplica-se apenas a redes com topologia em árvore, a cada rede foi aplicado inicialmente o algoritmo de Prim de maneira a obter uma árvore abrangente mínima que será então a rede de entrada dos algoritmos desenvolvidos. A linguagem utilizada neste trabalho foi o Java. De maneira a criar as imagens das árvores para este capítulo e para o Apêndice foi utilizado o graphiz(http://graphiz.org). Os restantes gráficos foram produzidos através do MATLAB.

4.2 Cálculo da centralidade dos nós

Para o cálculo da métrica de centralidade desenvolvida é necessário o cálculo do fatorial do número de nós, como se pode ver na expressão (3.2). Como o maior número (usando

uma variável *double*) em Java para o qual ainda é possível calcular o fatorial é 170, foi necessário pensar numa estratégia de modo a que o algoritmo funcionasse para redes com mais de 170 nós. É de notar que o cálculo da centralidade para o nó raiz é feito através da expressão (3.2), mas para os restantes nós é feito através da expressão (3.3), pelo que se pode concluir que o valor da centralidade do nó raiz está presente no cálculo da centralidade dos outros nós. Como é um fator multiplicativo, a solução adotada foi atribuir à centralidade do nó raiz um valor igual a 1. Assim foi possível contornar o limite imposto pelo fatorial e obter os mesmos resultados (árvore centróide e classificação dos nós de protecção) dado que apenas estamos interessados na importância relativa dos nós, em termos de centralidade, e não no valor absoluto da centralidade para cada nó.

Pretende-se então classificar os nós de acordo com o seu grau de criticidade para que o impacto de uma eventual falha, com potencial para se propagar na rede, seja o menor possível. O nó considerado mais crítico é o que está colocado na rede de tal forma que, se for origem de uma falha deste tipo, ou se for contaminado por uma falha originada noutro nó, tem maior capacidade para disseminar a falha por toda a rede. Esse é também o nó de maior centralidade. Se esse nó for protegido, está-se então a limitar a abrangência de uma falha em cascata, dado que todos os outros nós, em principio, têm menor capacidade de propagar uma falha deste tipo.

No caso da rede não estar protegida contra falhas em cascata, o potencial de impacto deste tipo de falha é de 100 %, ou seja, pode abranger todos os nós da rede (considerando o modelo "Susceptible-Infected(SI) Spreading Model"[3]). No entanto, com a proteção de alguns nós é possível limitar a propagação deste tipo de falhas, como se ilustra a seguir.

Tomando como exemplo a árvore abrangente mínima obtida pelo algoritmo de Prim para a rede b01, com raiz em v_1, exibida na figura 4.1, explica-se de seguida a estratégia de escolha dos nós a proteger de modo a evitar a propagação de falhas em cascata.

O primeiro passo é a obtenção da árvore centróide para a rede representada na figura 4.1 e de seguida ordena-se os nós de acordo com o tamanho da sua sub-árvore na árvore centróide, ou seja, de acordo com o tamanho das sub-árvores com raiz em v_i nessa árvore (com $i = 1, ..., N$).

O primeiro centróide obtido é o nó com maior centralidade e está representado na figura 4.1 pela cor azul (nó v_{20}). Após a obtenção do 1º centróide, este é retirado da

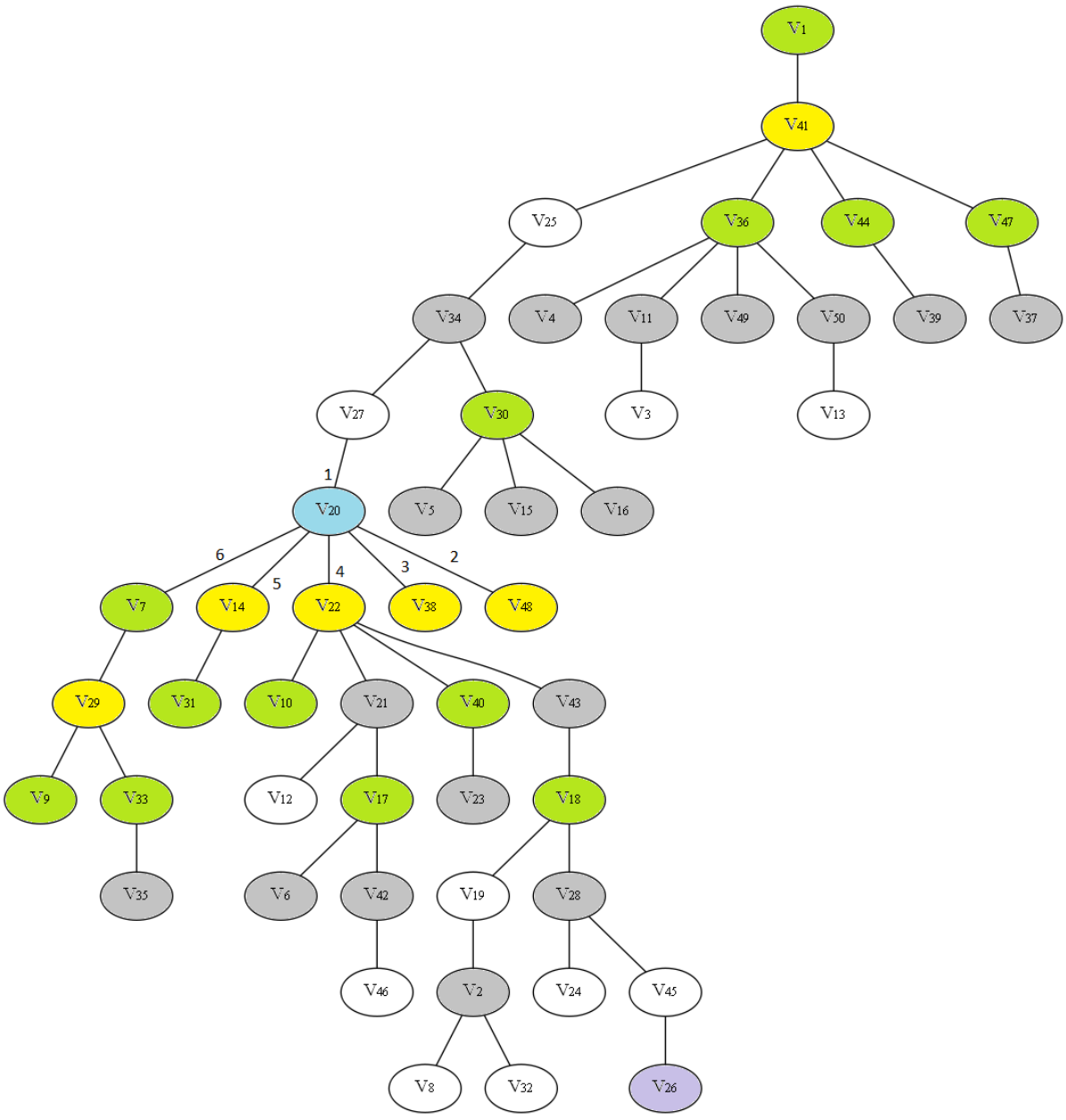

Figura 4.1: Árvore abrangente mínima obtida pelo algoritmo de Prim da rede b01 iniciado no nó v_1.

árvore ficando esta dividida em seis sub-árvores, dado que v_{20} tem grau seis, como se pode verificar pela análise da figura 4.1. As primeiras sub-árvores obtidas são então numeradas de 1 a 6 (ver figura 4.1) e à medida que mais sub-árvores vão surgindo estas vão sendo numeradas sucessivamente. Cada uma destas sub-árvores é de seguida analisada da mesma maneira que a árvore original, sendo daqui obtidos os 2º centróides que estão assinalados a amarelo na figura 4.1. Continuando com o mesmo procedimento, depois da obtenção destes 2º centróides e depois destes serem retirados da respetiva sub-árvore, surgem novas sub-árvores. Nas novas sub-árvores são encontrados os 3º centróides que estão assinalados a verde. Os 4º's, 5º's e 6º's centróides são então obtidos sucessivamente seguindo o mesmo procedimento e estão assinalados na figura 4.1 a cinzento, a branco e a lilás, respetivamente. Tendo esta fase concluída, chegamos à árvore centróide da figura 4.2.

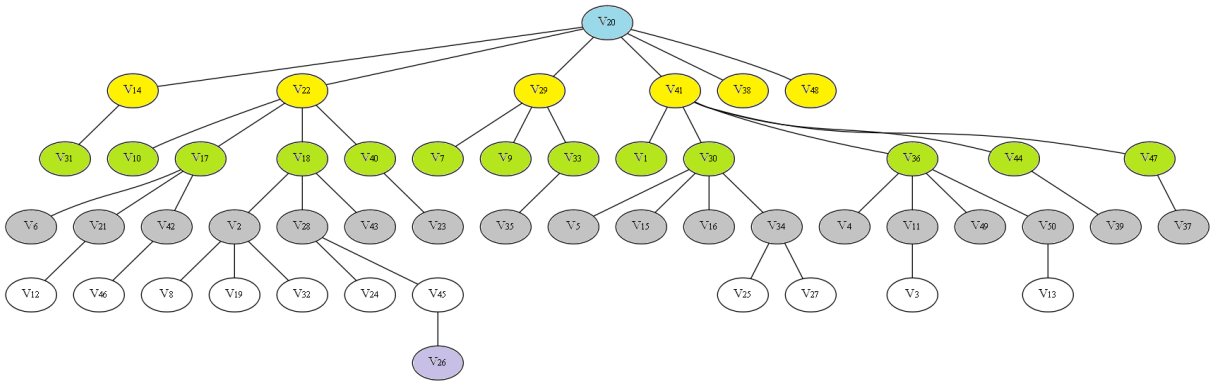

Figura 4.2: Árvore centróide da rede representada na figura 4.1.

De seguida é necessário calcular os tamanhos das sub-árvores com raiz em v_i, com $i = 1, ..., N$ e ordenar os nós de acordo com o tamanho da sua sub-árvore. A classificação obtida está na tabela 4.1.

Tabela 4.1: Tamanho da sub-árvore, na árvore centróide da figura 4.2, com origem em v_i.

v_i	$\|desc(v_i)\|+1$	v_i	$\|desc(v_i)\|+1$	v_i	$\|desc(v_i)\|+1$	v_i	$\|desc(v_i)\|+1$
20	50	14	2	4	1	25	1
41	20	50	2	38	1	13	1
22	20	11	2	1	1	3	1
18	10	21	2	49	1	12	1

Tabela 4.1: (Continuação)

v_i	$\|desc(v_i)\|+1$	v_i	$\|desc(v_i)\|+1$	v_i	$\|desc(v_i)\|+1$	v_i	$\|desc(v_i)\|+1$
36	7	42	2	7	1	46	1
30	7	44	2	6	1	10	1
17	6	47	2	9	1	24	1
29	5	45	2	43	1	19	1
28	4	16	1	48	1	32	1
2	4	5	1	31	1	8	1
34	3	15	1	23	1	26	1
40	2	39	1	35	1		
33	2	37	1	27	1		

4.3 Proteção contra falhas em cascata

Vamos considerar, por exemplo, que queremos proteger 3 nós na árvore da figura 4.1. Neste caso os nós escolhidos são o v_{20}, v_{41} e v_{22} dado que são os nós de maior centralidade, como se pode ver na tabela 4.1. Supondo que uma falha em cascata é iniciada no nó v_{13} podemos verificar na figura 4.1 que são afetados 7 dos 50 nós existentes, uma vez que ao protegermos o nó v_{41} este nó não permite que a falha se propague.

Para avaliar a eficácia da escolha dos três nós a proteger referidos anteriormente avaliou-se a abrangência da falha em cascata considerando que esta teve origem em cada um dos nós da rede. Os resultados estão indicados na tabela 4.2.

Tabela 4.2: Número de nós afetados por nó origem da falha (v_i) na condição dos nós v_{20}, v_{41} e v_{22} estarem protegidos.

v_i	Nº Afet.	v_i	Nº Afet.	v_i	Nº Afet.	v_i	Nº Afet.
1	1	14	2	27	7	40	2
2	10	15	7	28	10	41	0
3	7	16	7	29	5	42	6
4	7	17	6	30	7	43	10
5	7	18	10	31	2	44	2

Tabela 4.2: (Continuação)

v_i	Nº Afet.	v_i	Nº Afet.	v_i	Nº Afet.	v_i	Nº Afet.
6	6	19	10	32	10	45	10
7	5	20	0	33	5	46	6
8	10	21	6	34	7	47	2
9	5	22	0	35	5	48	1
10	1	23	2	36	7	49	7
11	7	24	10	37	2	50	7
12	6	25	7	38	1		
13	7	26	10	39	2		

Como se pode verificar, pela análise da tabela 4.2 o máximo de nós afetados que obtemos ao proteger os 3 primeiros nós é de 10 nós, ou seja, no caso de uma eventual falha em cascata, só 1/5 da rede será afetado, tendo em conta que os nós protegidos não falharam no acto de proteger. Com estes resultados é interessante avaliarmos o comportamento das falhas em cascata em função do número de nós protegidos.

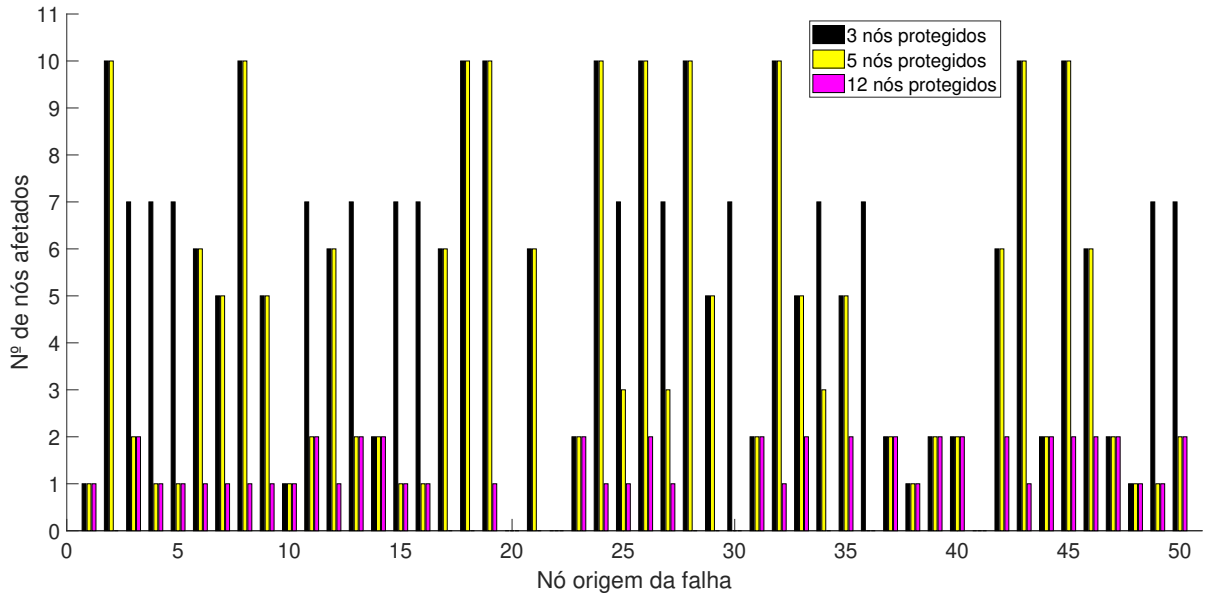

Figura 4.3: Número de nós afetados em função de uma falha em cascata iniciada em cada nó da rede da figura 4.1, considerando 3, 5 e 12 nós protegidos.

Na figura 4.3 são apresentados os resultados obtidos para 3, 5 e 12 nós protegidos. É

notório que quantos mais nós protegemos, igual ou menor número de nós serão afetados pela falha em cascata qualquer que seja o nó que a inicie.

Podemos verificar na figura 4.4 a diminuição do máximo e média do número de nós afetados ao proteger mais nós na rede de acordo com a classificação da tabela 4.1, obtida através da estratégia apresentada baseada na métrica de centralidade (CTR). Este gráfico

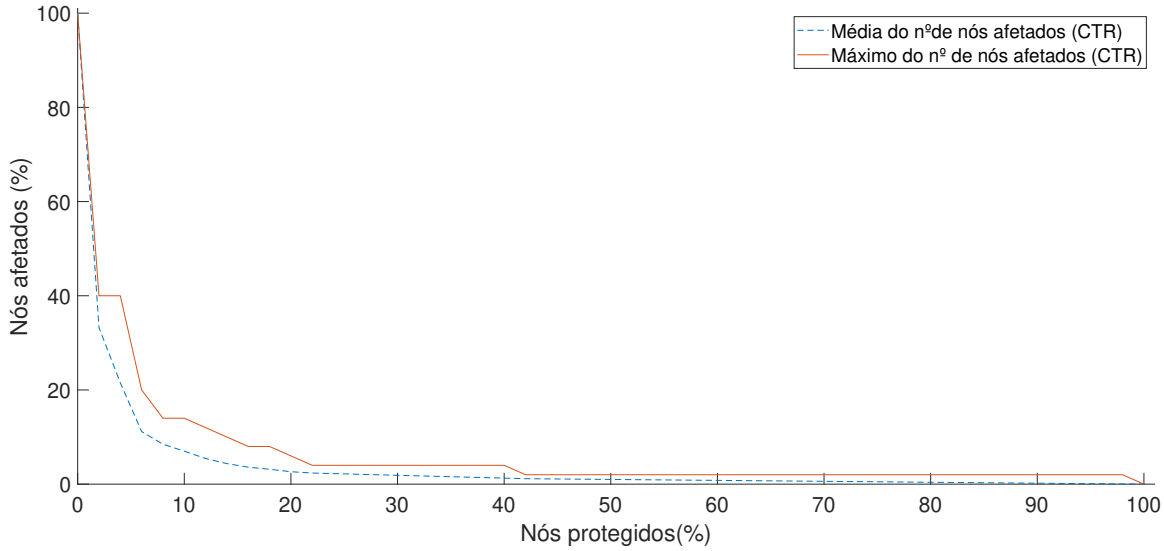

Figura 4.4: Percentagem do número médio e máximo de nós afetados em função da percentagem de nós protegidos na rede da figura 4.1.

mostra uma exponencial negativa, i. e., é notório que a diferença é mais considerável para percentagens de nós protegidos mais baixas. A partir de 42 % o valor do máximo mantém-se constante porque estamos a proteger nós em que os seus adjacentes já foram protegidos. Devido à mesma razão, também a partir da mesma percentagem a linha da média vai descendo com declive constante até atingir o valor nulo.

Como este comportamento foi verificado em todas as redes analisadas, os resultados que se seguem (referentes à média e ao máximo de nós afetados) foram registados considerando até 40 % de nós protegidos.

4.4 Comparação com o grau do nó

De modo a avaliar a eficácia da métrica de centralidade desenvolvida optou-se por comparar os resultados obtidos com esta métrica com os resultados obtidos protegendo os

nós da rede de maior grau. Não houve critério de desempate entre dois nós com o mesmo valor de centralidade, porque as estratégias que foram experimentadas para desempate não conduziram a resultados significativamente diferentes dos que são apresentados nesta dissertação. Assim o primeiro nó que surgiu nos cálculos com maior valor de centralidade foi o escolhido como centróide.

Na tabela 4.3 estão ordenados os nós de acordo com o seu respetivo grau para o caso da rede representada na figura 4.1. Quando o grau de dois nós é igual, o tamanho das sub-árvores com raiz em v_i foi usado como critério de desempate. Desta maneira, o nó com maior sub-árvore é considerado primeiro e está colocado numa posição mais acima na tabela.

Tabela 4.3: Grau de v_i na rede representada na figura 4.1.

v_i	Grau	v_i	Grau	v_i	Grau	v_i	Grau
20	6	27	2	1	1	10	1
41	5	43	2	5	1	4	1
22	5	7	2	15	1	23	1
36	5	19	2	16	1	24	1
30	4	11	2	31	1	12	1
34	3	40	2	32	1	46	1
18	3	33	2	6	1	26	1
21	3	42	2	3	1	48	1
28	3	14	2	35	1	49	1
29	3	44	2	8	1	13	1
17	3	45	2	38	1		
2	3	47	2	39	1		
25	2	50	2	9	1		

Na figura 4.5 estão representados os valores médio e máximo de nós afetados em função do número de nós protegidos quando estes são escolhidos de acordo com a métrica de centralidade (CTR) referida anteriormente e de acordo com o grau do nó (Grau), para o caso da rede na figura 4.1.

Como se pode verificar pela análise da figura 4.5, a média e o máximo de nós afetados

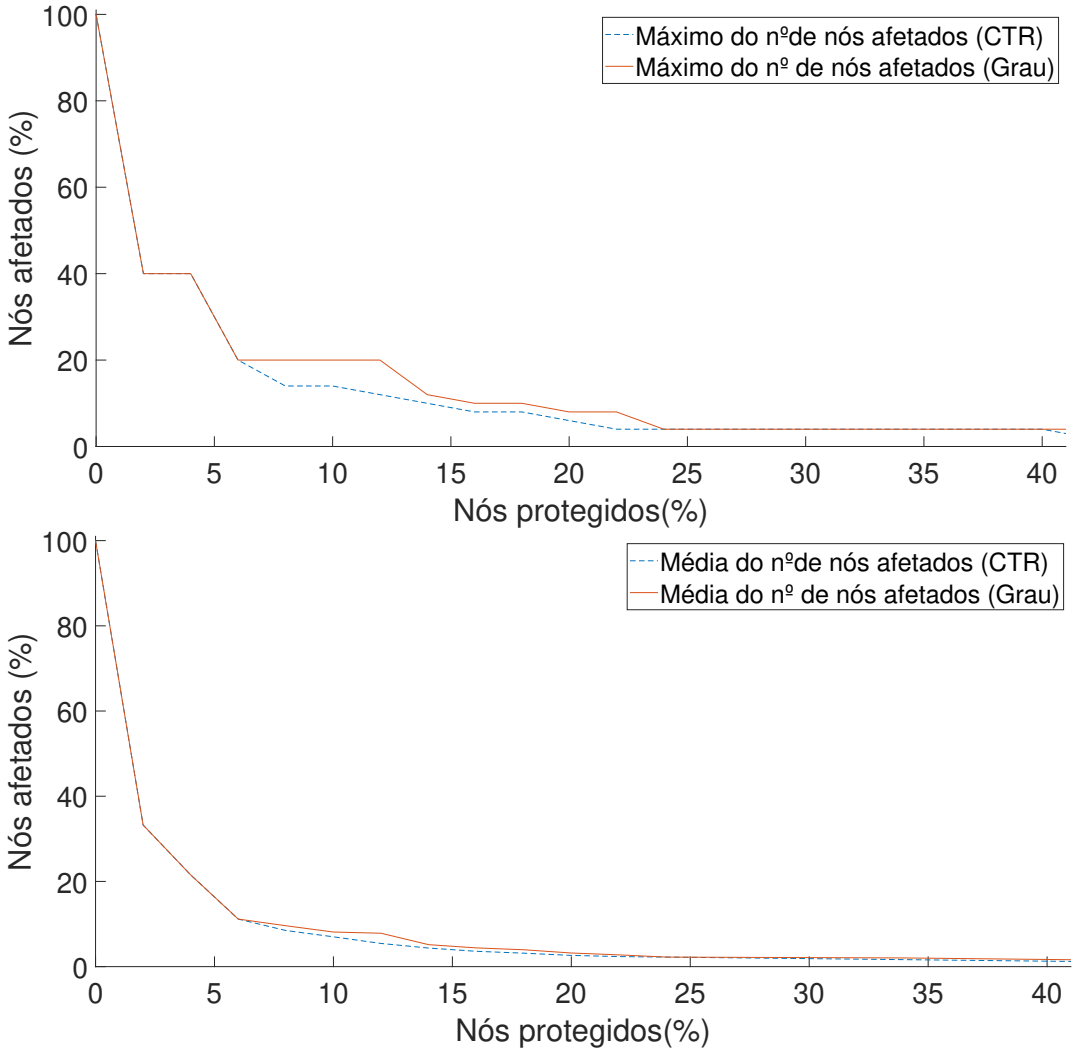

Figura 4.5: Percentagem do número máximo (em cima) e médio (em baixo) e de nós afetados em função da percentagem de nós protegidos para a rede da figura 4.1.

é menor quando os nós protegidos são escolhidos de acordo com a métrica de centralidade estudada. De facto, este é o comportamento mais frequente no caso das redes estudadas, tal como se pode ver pelos resultados apresentados nos apêndices A.4 e B.1.

No entanto, nem sempre a centralidade conduz ao melhor desempenho da rede no caso de falhas em cascata como se pode ver na figura 4.6. Ao analisar esta figura podemos verificar que a métrica de centralidade não é perfeita e nem sempre é a melhor opção. Neste caso se se pretendesse proteger somente 4 % dos nós, seria mais vantajoso optar pela métrica grau do nó do que pela centralidade. Mas se o objetivo fosse proteger 15 % dos nós a métrica de centralidade já seria a que conduzia a melhores resultados.

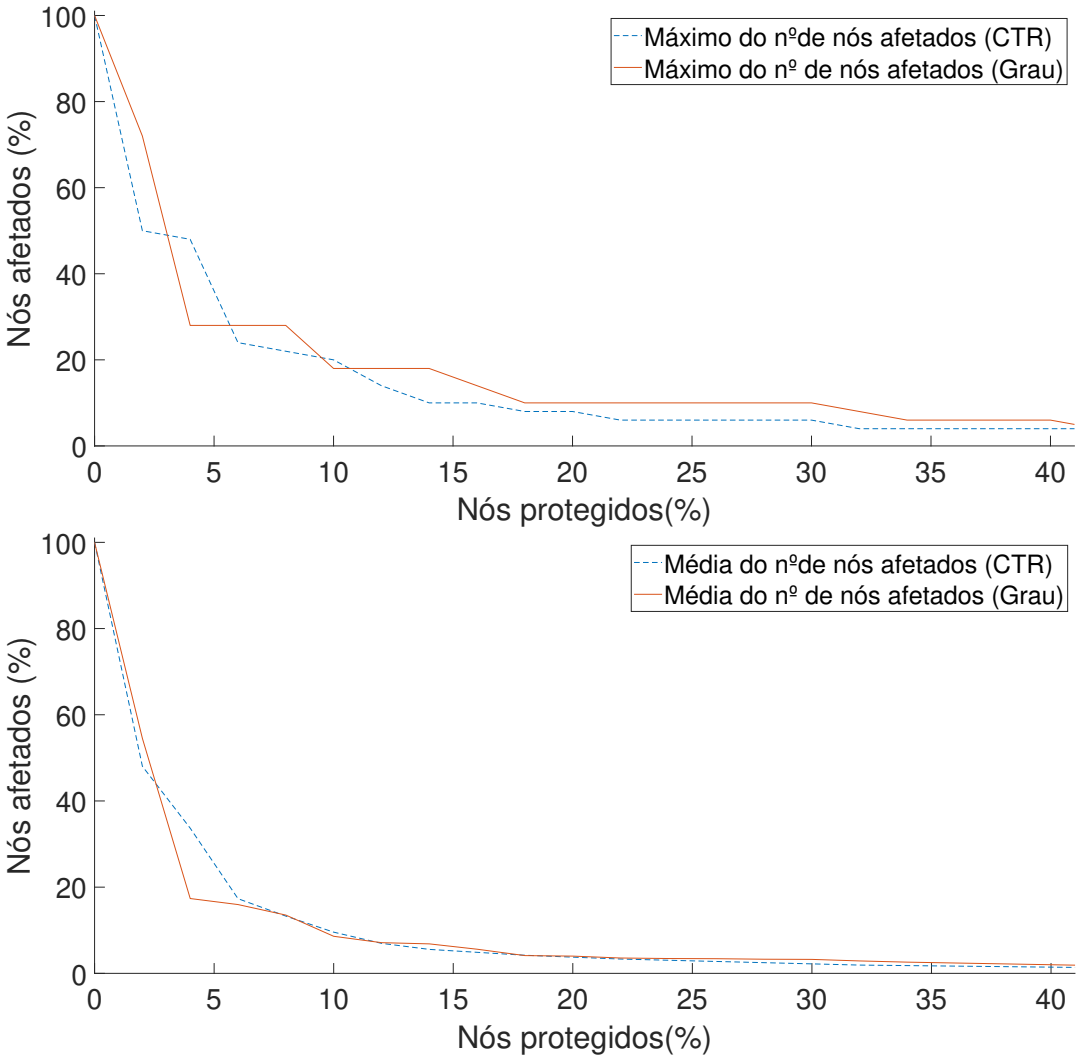

Figura 4.6: Percentagem do número máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a rede da figura A.1.

Apesar de na maior parte dos casos a centralidade apresentar melhores resultados, o grau do nó permite pontualmente, tal como no caso anterior, obter melhores resultados. Das 11 redes estudadas, para as quais os resultados são apresentados nos apêndices A.4 e B.1, em 5 casos o grau do nó conduz a melhores resultados para percentagens pontuais de nós protegidos. Na globalidade a métrica de centralidade estudada conduz a melhores resultados, tal como foi referido anteriormente.

De modo a verificar se os resultados anteriores se mantinham para redes maiores, foram usadas 4 redes com 500 nós do conjunto C em *http://steinlib.-zib.de/testset.php* apresentando-se os resultados para a primeira rede na figura 4.7. Os restantes resultados

podem ser consultados no Apêndice B.1. Como se pode verificar, a métrica de centralidade conduz a melhores resultados no geral, mantendo o padrão já verificado nas redes anteriores. Pontualmente a métrica grau do nó consegue ligeiramente melhores resultados, como também já foi referido.

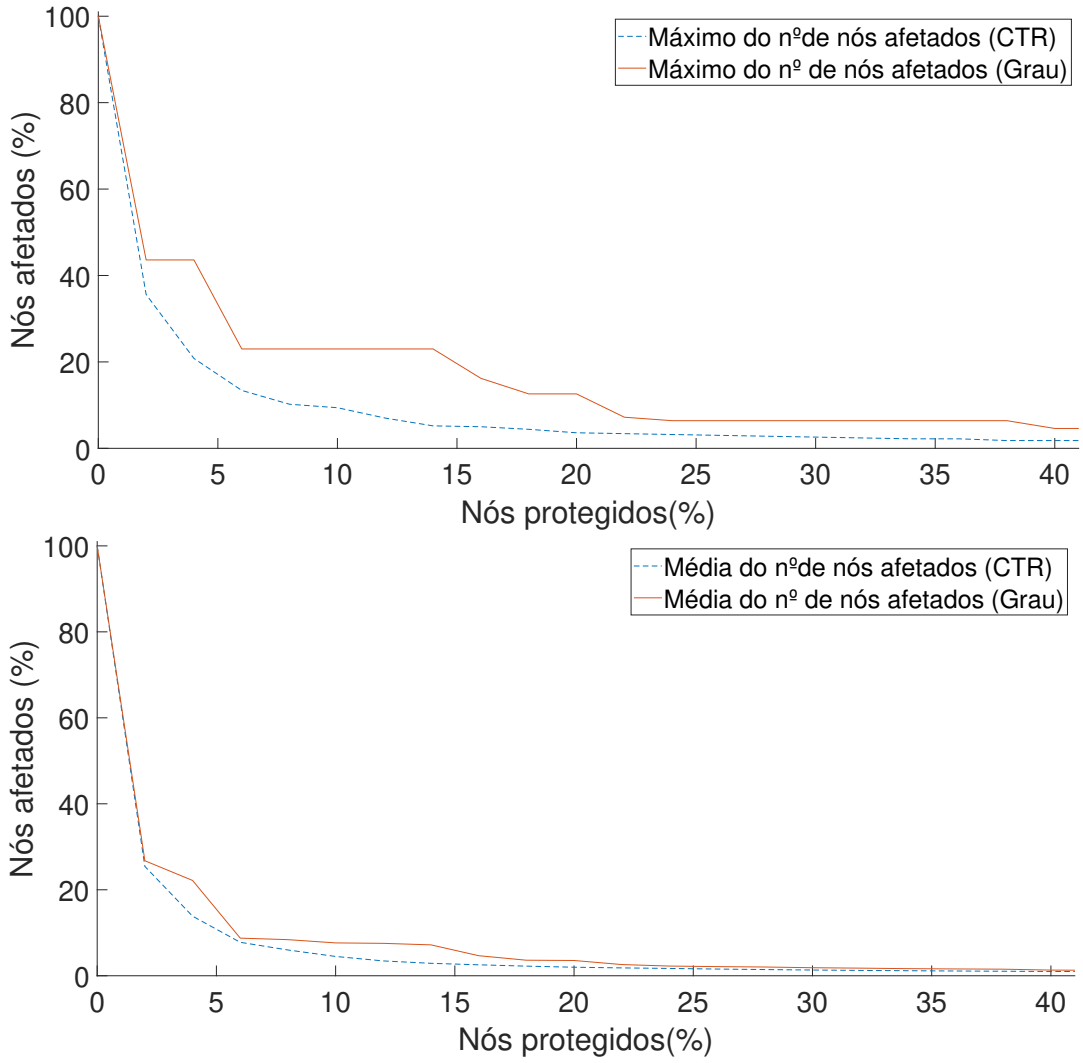

Figura 4.7: Percentagem do número e máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a rede c01 (após correr o Prim).

4.5 Tempos de execução

Nesta secção são apresentados os tempos de execução do algoritmo de classificação de nós a proteger contra falhas em cascata. Para retirar estes tempos foram feitas 10 corridas para cada uma das redes em árvore e contabilizou-se apenas os tempos de cálculo da centralidade, da construção da árvore centróide e da classificação dos nós. Não está aqui contabilizado o tempo que demorou a verificar a quantidade de nós afetados em função de um determinado número de nós protegidos. É de referir que nos apêndices A e B são apresentados resultados complementares relativos a este conjunto de redes.

Tabela 4.4: Tempos de execução dos algoritmos para obtenção da árvore centróide e classificação dos nós.

N	Rede	Média (ms)	Desvio padrão (ms)
50	b01	5,3082	1,4707
50	b02	4,5221	1,4506
75	b09	11,0663	1,6452
75	b10	9,3964	1,2331
75	b11	10,1097	0,9748
100	b16	13,0841	1,2035
100	b17	13,5745	0,7545
500	c01	209,9140	12,0089
500	c02	194,7564	10,5886
500	c06	257,7652	18,3469
500	c07	278,2321	20,6128

Os tempos apresentados na tabela 4.4 estão em *ms*. O computador utilizado tem um processador Intel i7-4712MQ a 2,30GHz com 8 GB de RAM. Os tempos obtidos mostram que os algoritmos desenvolvidos são relativamente rápidos tendo demorado cerca de 1/5 a 1/4 de segundo no caso das redes de 500 nós estudadas.

De modo a avaliar melhor os tempos obtidos utilizou-se o algoritmo de Dijkstra no cálculo da árvore centróide. Tal como já foi apresentado no capítulo 3, o centróide de uma

rede pode ser calculado recorrendo a este algoritmo. O tempo médio obtido na construção da árvore centróide para a rede da figura 4.2 foi então de 13,0692 *ms* e o desvio padrão de 1,8006 *ms*. O tempo médio foi cerca de 2,5 vezes superior ao obtido anteriormente para a mesma rede (5,3082 *ms*) e o desvio padrão diferiu 0,33 *ms*, como se pode ver na tabela 4.4. No caso da rede c01 (após executar o Prim) o tempo médio foi de 299,3048 *ms* e o desvio padrão de 6,2479 *ms*. Desta maneira, verificou-se que o tempo médio foi aproximadamente 1,43 vezes superior e o desvio padrão foi reduzido para metade.

Contudo, foi comprovado que o algoritmo de passagem de mensagens é mais rápido que o algoritmo de Dijkstra para a obtenção do centróide.

Capítulo 5

Conclusão

Nesta dissertação de mestrado foi abordado o tema das falhas em cascata em redes com topologia em árvore. Assim, o estudo realizado pretende ser um contributo para a identificação de melhores abordagens para proteção das redes contra eventuais falhas em cascata.

Para tratar o tema estudou-se uma métrica de centralidade discutida nos artigos [5], [4] e [6]. Com base nesses mesmos artigos foram desenvolvidos algoritmos para determinar uma classificação em que os nós são ordenados por criticidade, sendo que o nó mais crítico é o de maior centralidade na rede. Estando esses nós protegidos limita-se o alcance de uma falha em cascata.

Depois de realizada uma análise de resultados em que se comparou o uso das métricas de centralidade e grau do nó para escolha dos nós a proteger, chegou-se à conclusão que dependendo do número de nós a proteger, a métrica que conduz aos melhores resultados pode não ser a mesma, embora em geral a métrica de centralidade estudada conduza a melhores desempenhos na rede.

Os resultados obtidos permitem concluir que a métrica estudada é uma boa métrica para escolha de nós críticos embora talvez sejam necessárias estratégias mais sofisticadas do que as usadas neste trabalho para melhorar o desempenho da rede.

A escolha de nós críticos baseou-se na abordagem apresentada em [5] e [6] embora nesta dissertação tenha sido feita uma análise exaustiva do comportamento da estratégia apresentada para escolha dos nós a proteger que foi muito além dos resultados apresentados em [5] e [6].

O estudo de nós críticos em redes com topologias gerais, e não apenas com topologia em árvore, é um aspeto importante para estudo em trabalhos futuros.

Bibliografia

[1] Ravindra K Ahuja, Thomas L Magnanti, and James B Orlin. Network flows. 1988.

[2] L. Martins, R. Girao-Silva, L. Jorge, A. Gomes, F. Musumeci, and J. Rak. Interdependence between power grids and communication networks: A resilience perspective. In *DRCN 2017 - Design of Reliable Communication Networks; 13th International Conference, IEEE*, pages 1–9, March 2017.

[3] Wenjun Mei, Shadi Mohagheghi, Sandro Zampieri, and Francesco Bullo. On the dynamics of deterministic epidemic propagation over networks. *Annual Reviews in Control*, 44:116–128, 2017.

[4] Devavrat Shah and Tauhid Zaman. Rumors in a network: Who's the culprit? *IEEE Transactions on Information Theory*, 57(8):5163–5181, 2011.

[5] Pei-Duo Yu, Chee Wei Tan, and Hung-Lin Fu. Averting cascading failures in networked infrastructures: Poset-constrained graph algorithms. *IEEE Journal of Selected Topics in Signal Processing*, 12(4):733–748, 2018.

[6] Pei Duo Yu, Chee Wei Tan, and Hung-Lin Fu. Graph algorithms for preventing cascading failures in networks. In *2018 52nd Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6. IEEE, 2018.

Apêndice A

Redes B

Apresentam-se de seguida resultados complementares relativos a um conjunto representativo das redes B em *http://steinlib.zib.de/testset.php*, nomeadamente as redes b02, b09, b10, b11, b16 e b17.

Na secção A.1 apresentam-se as redes em árvore obtidas com o algoritmo de Prim tendo como nó raiz o nó v_1. É de notar que como podem existir várias redes abrangentes mínimas (que são ótimos alternativos), obtidas por este algoritmo, apresentam-se a seguir as árvores obtidas de modo a permitir replicar os resultados apresentados nesta dissertação. Estas árvores são então as redes de entrada para os algoritmos desenvolvidos.

Nas secções A.2 e A.3 apresentam-se respetivamente as árvores centróide e a avaliação do número de nós afetados em função do número de nós protegidos.

Finalmente, na secção A.4 é feita a comparação, em termos da abrangência de uma falha em cascata, quando os nós a proteger são escolhidos usando a métrica de centralidade desenvolvida e a métrica grau do nó.

A.1 Árvores obtidas pelo algoritmo de Prim

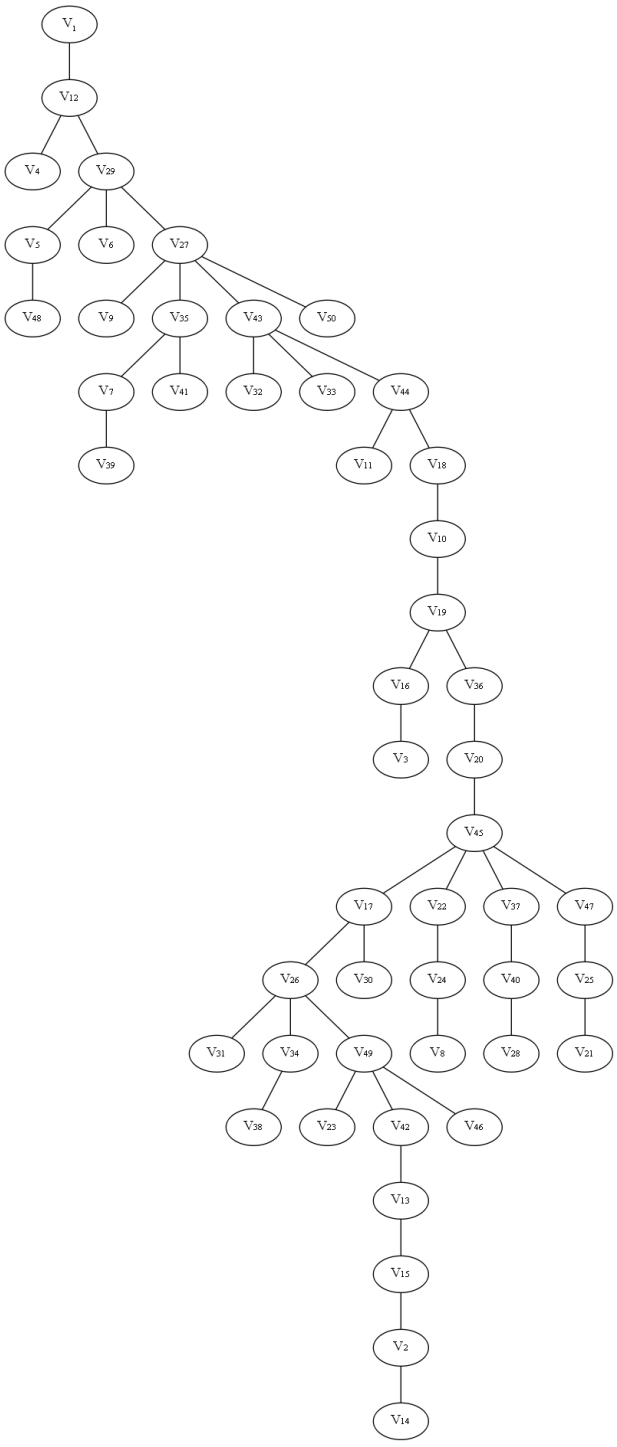

Figura A.1: Árvore abrangente mínima obtida pelo algoritmo de Prim, para a rede b02.

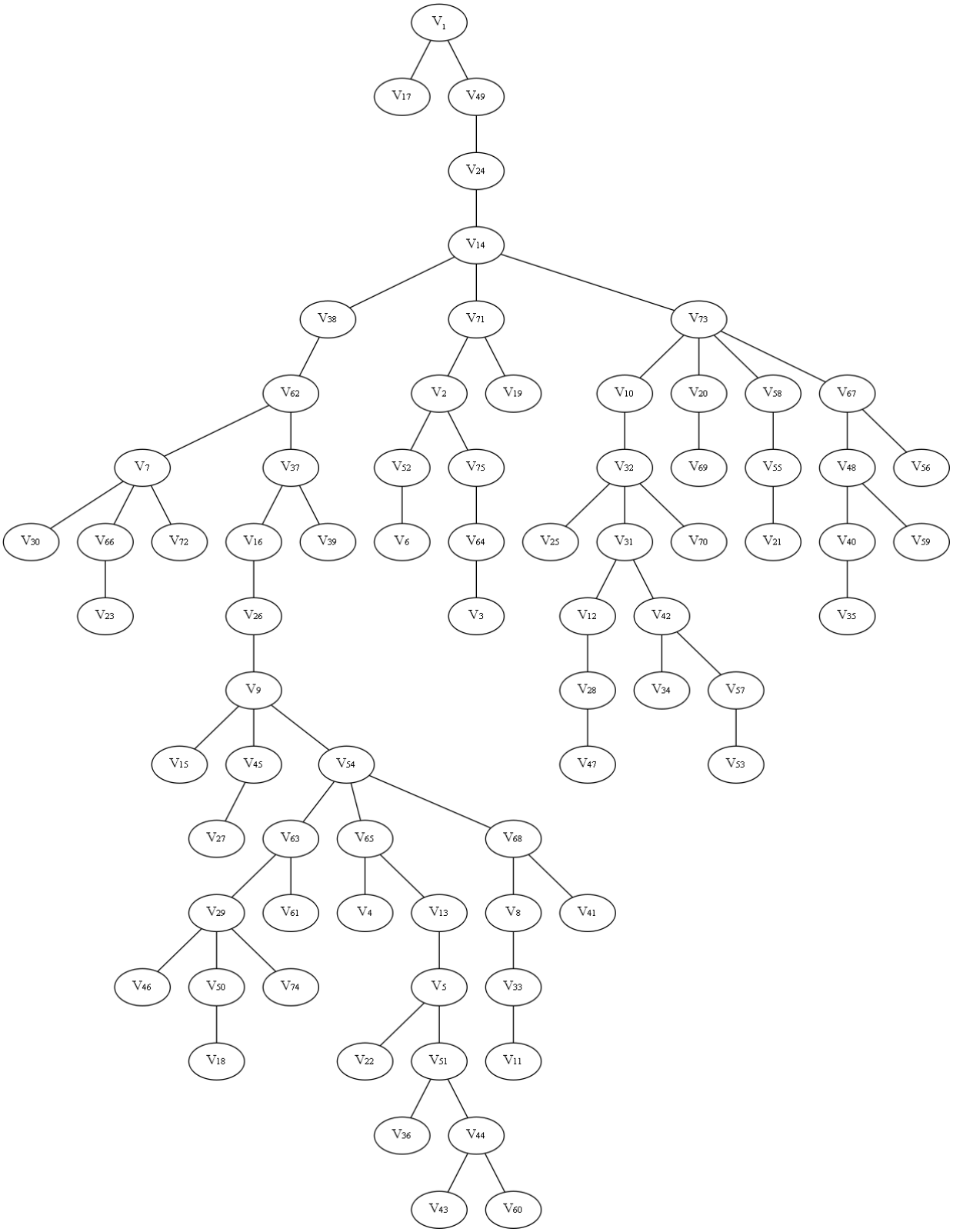

Figura A.2: Árvore abrangente mínima obtida pelo algoritmo de Prim, para a rede b09.

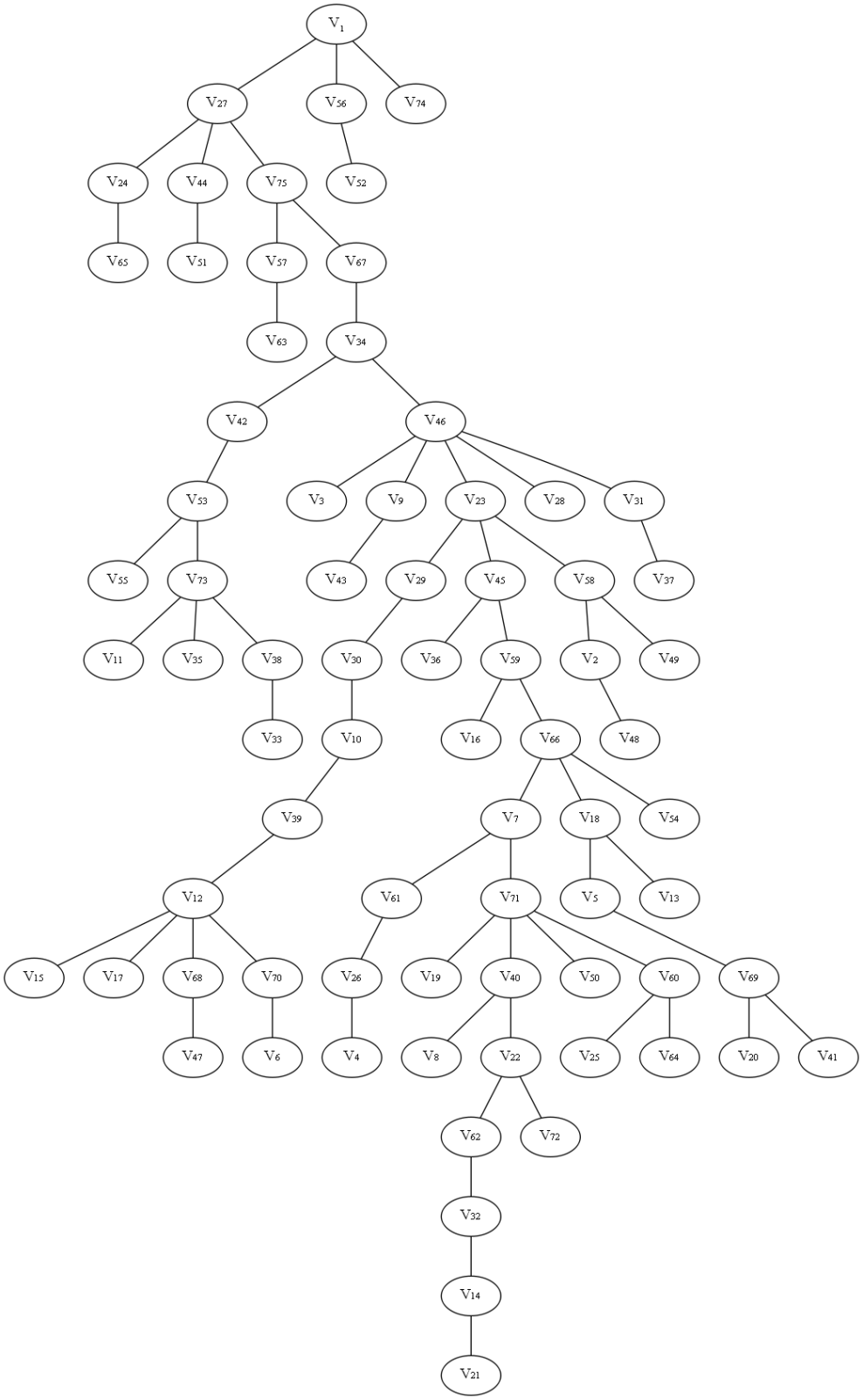

Figura A.3: Árvore abrangente mínima obtida pelo algoritmo de Prim, para a rede b10.

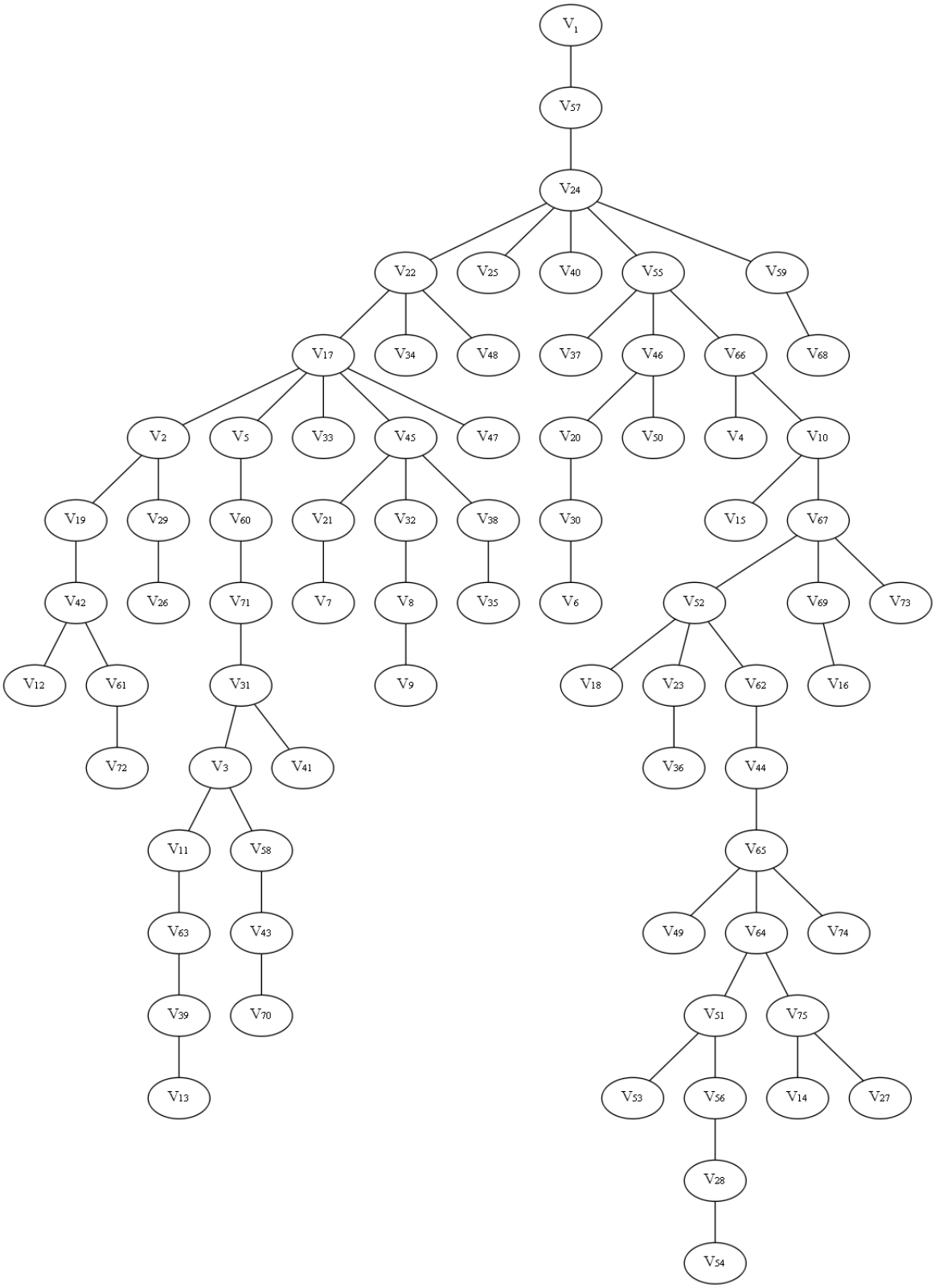

Figura A.4: Árvore abrangente mínima obtida pelo algoritmo de Prim, para a rede b11.

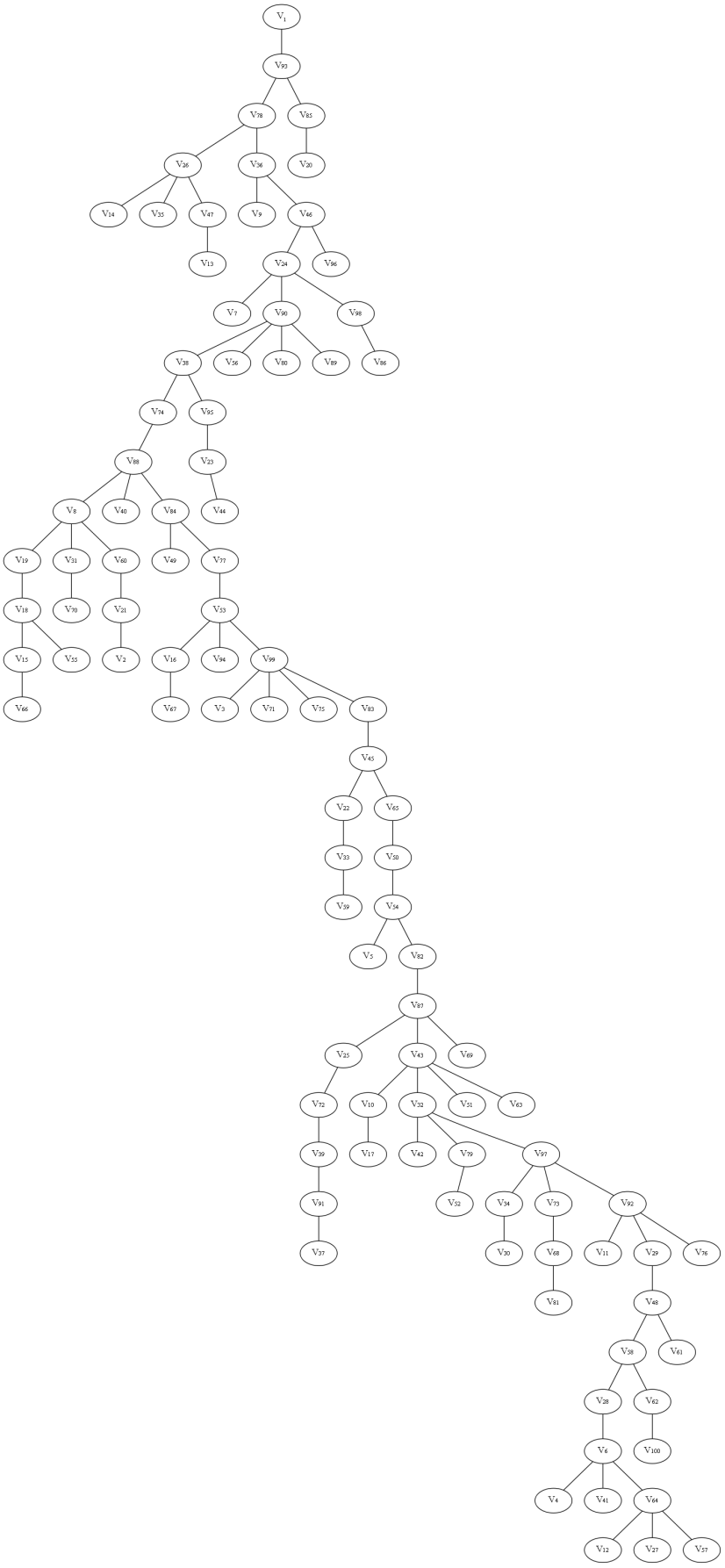

Figura A.5: Árvore abrangente mínima obtida pelo algoritmo de Prim, para a rede b16.

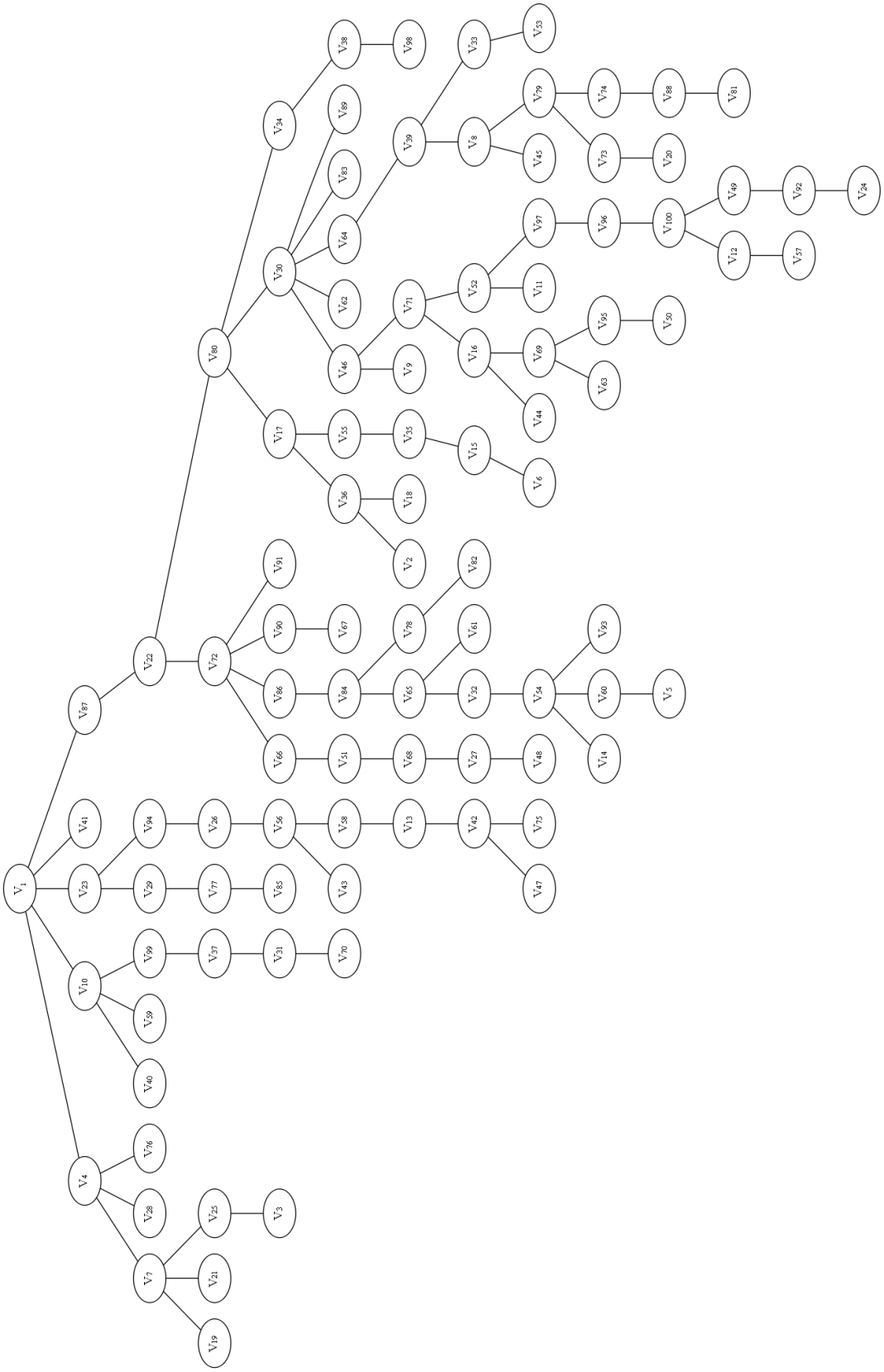

Figura A.6: Árvore abrangente mínima obtida pelo algoritmo de Prim, para a rede b17.

A.2 Árvores centróide

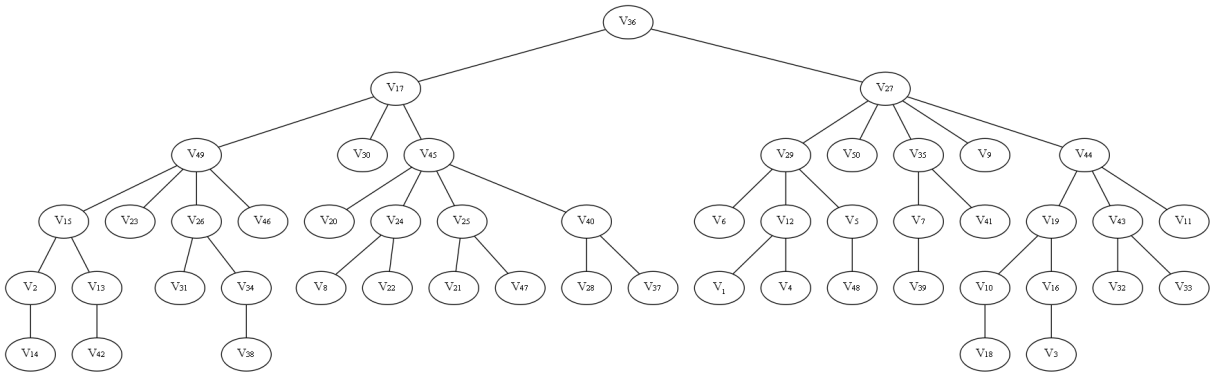

Figura A.7: Árvore centróide da figura A.1.

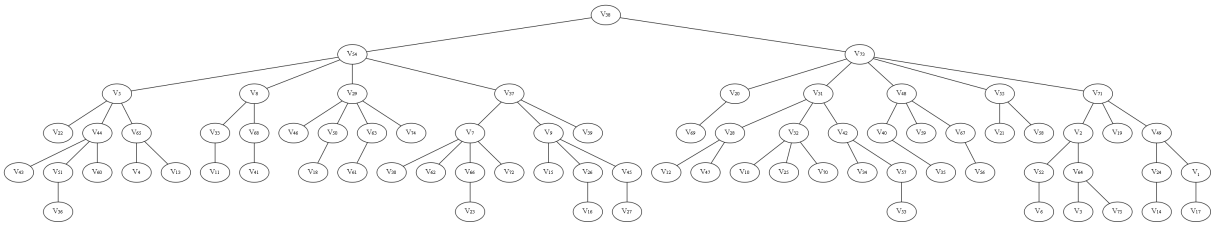

Figura A.8: Árvore centróide da figura A.2.

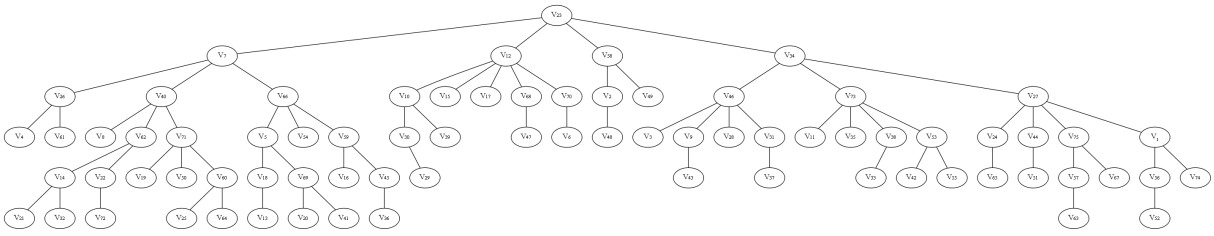

Figura A.9: Árvore centróide da figura A.3.

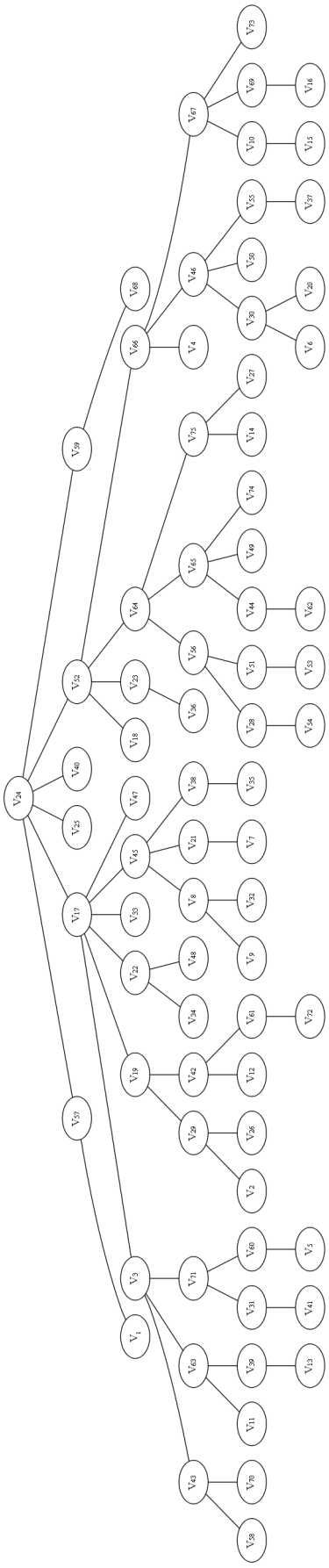

Figura A.10: Árvore centróide da figura A.4.

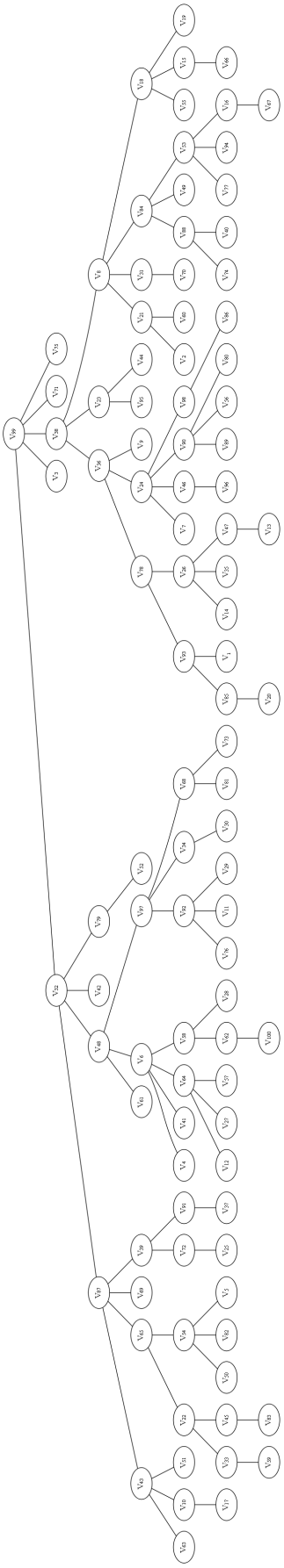

Figura A.11: Árvore centróide da figura A.5.

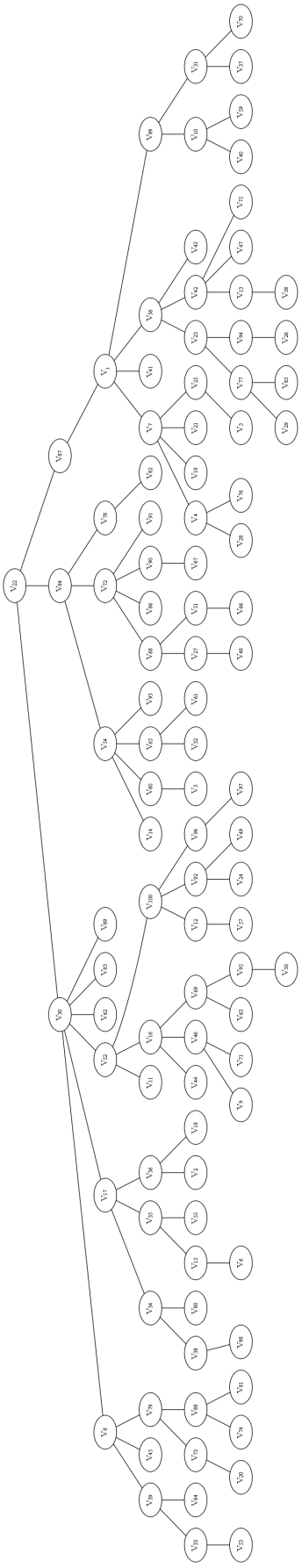

Figura A.12: Árvore centróide da figura A.6.

A.3 Número de nós afetados em função do número de nós protegidos

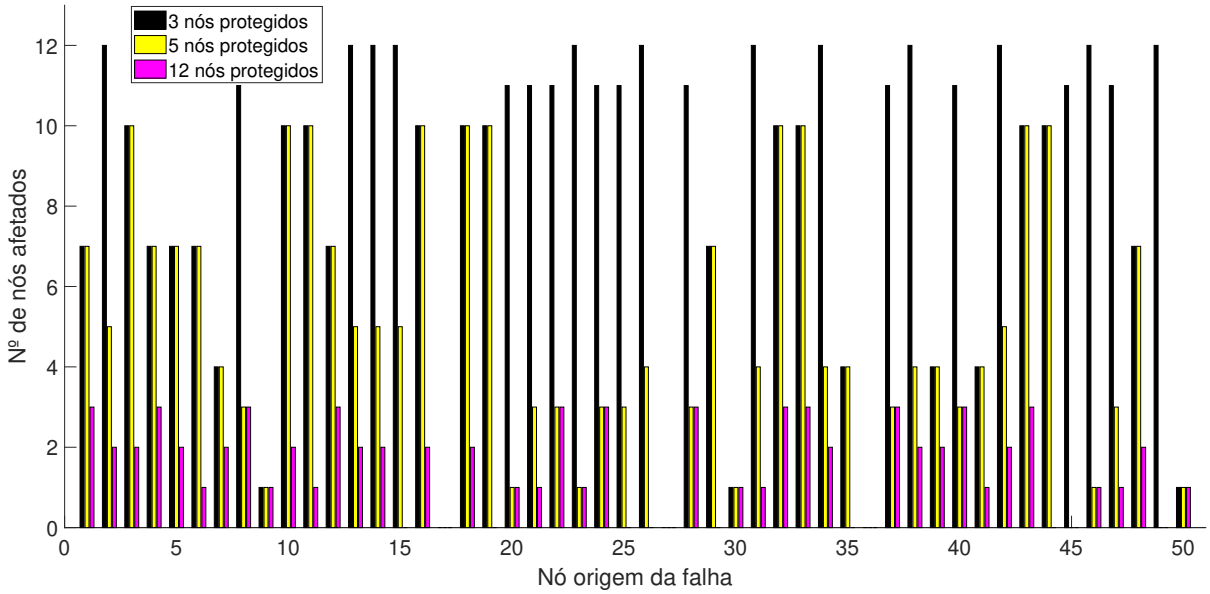

Figura A.13: Número de nós afetados em função de uma falha em cascata iniciada em cada nó da rede da figura A.1, considerando 3, 5 e 12 nós protegidos.

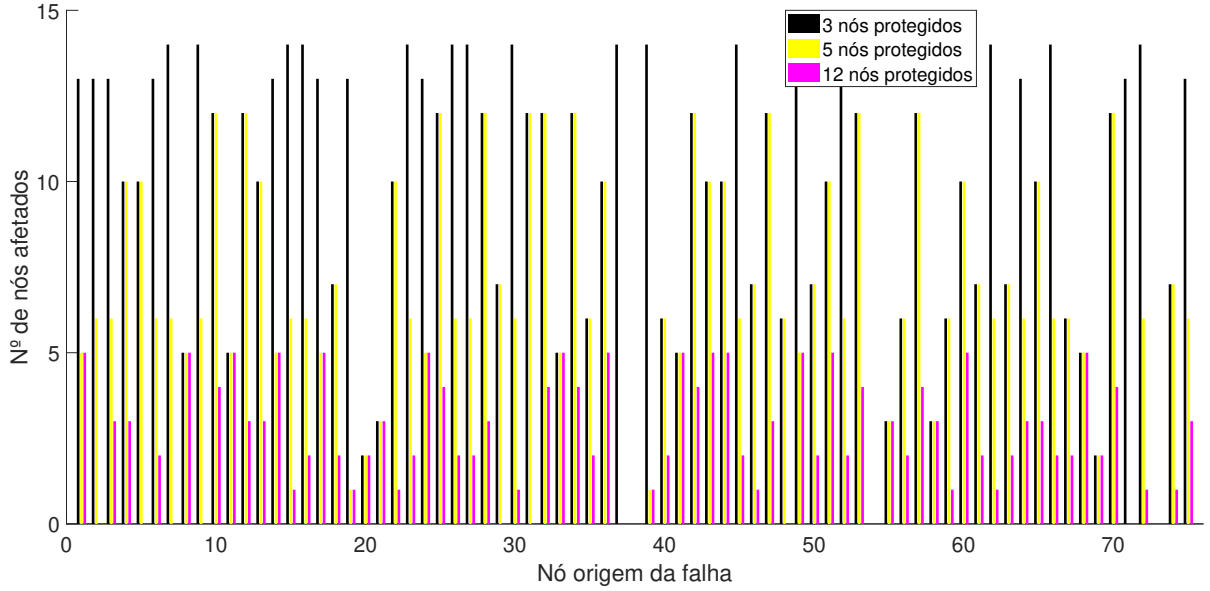

Figura A.14: Número de nós afetados em função de uma falha em cascata iniciada em cada nó da rede da figura A.2, considerando 3, 5 e 12 nós protegidos.

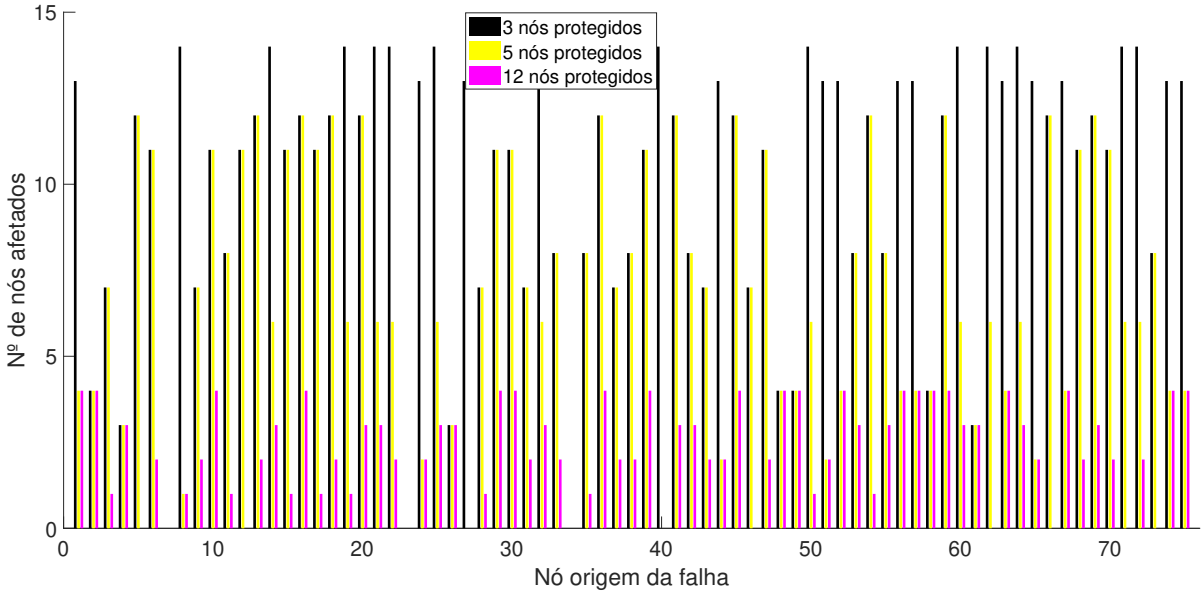

Figura A.15: Número de nós afetados em função de uma falha em cascata iniciada em cada nó da rede da figura A.3, considerando 3, 5 e 12 nós protegidos.

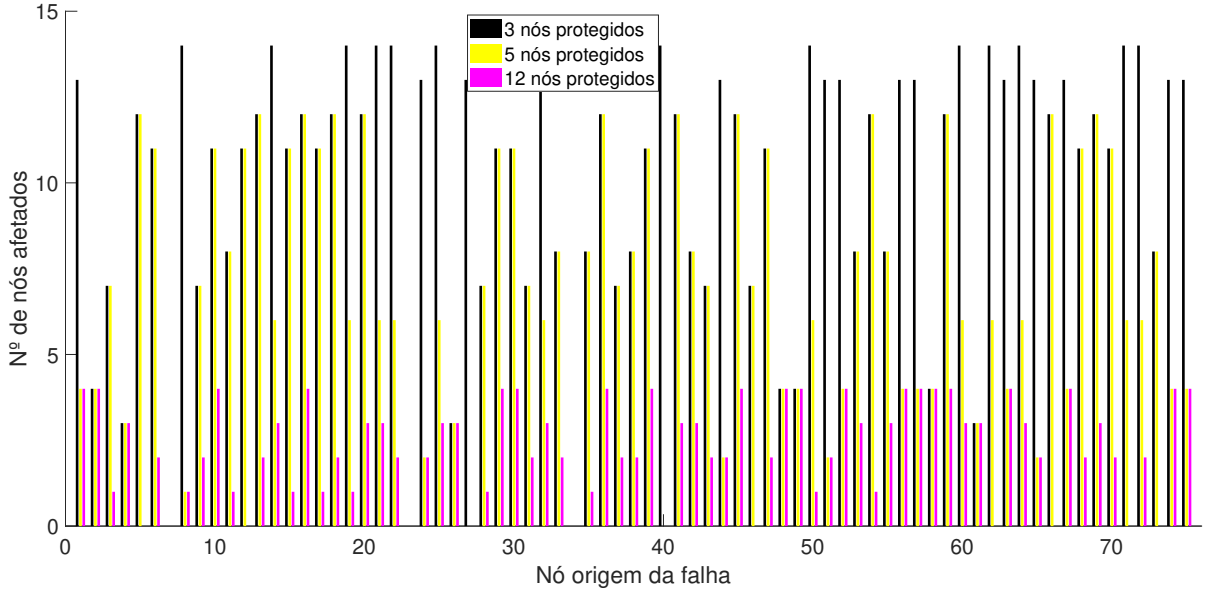

Figura A.16: Número de nós afetados em função de uma falha em cascata iniciada em cada nó da rede da figura A.4, considerando 3, 5 e 12 nós protegidos.

A.4 Centralidade vs grau do nó

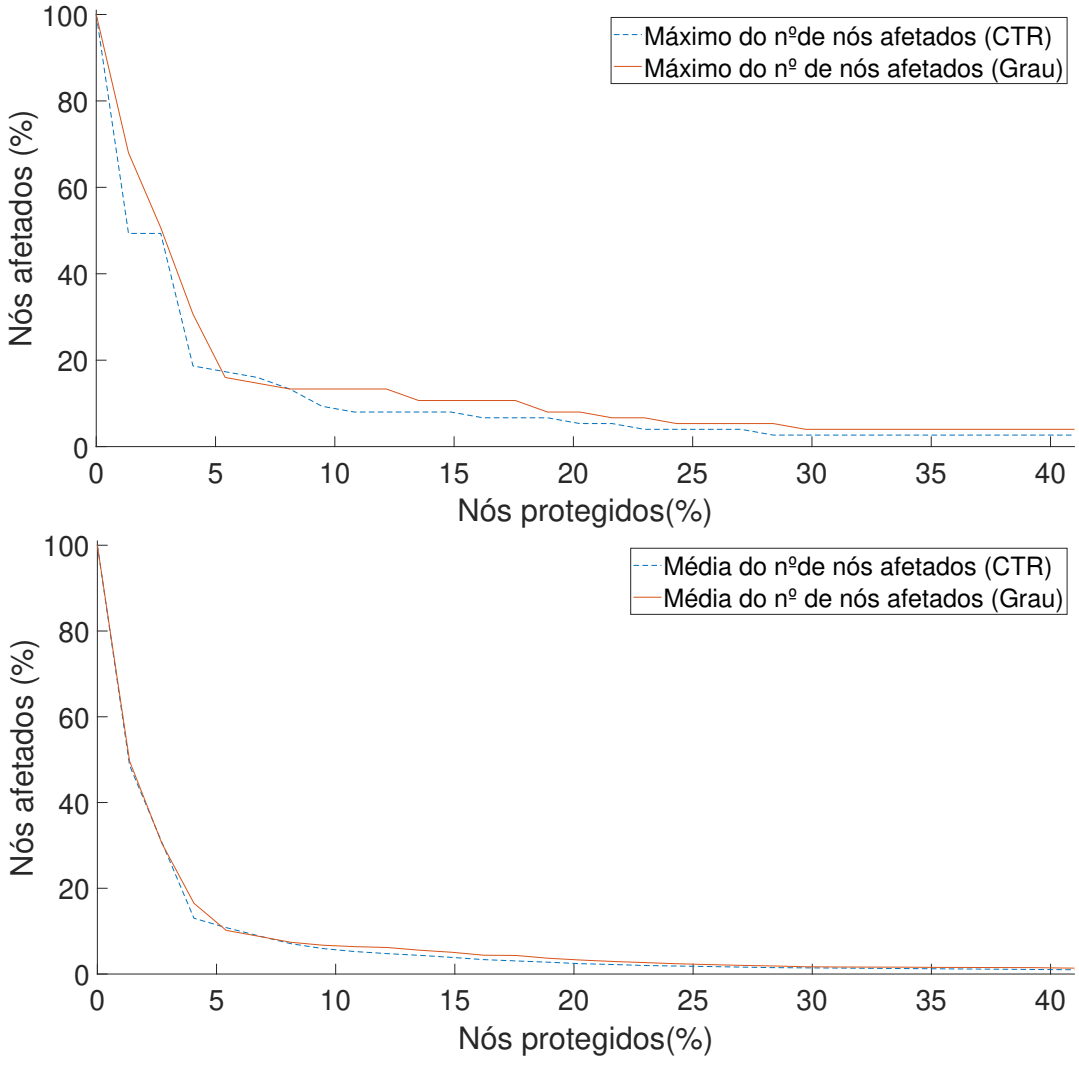

Figura A.17: Percentagem do número máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a árvore da figura A.2.

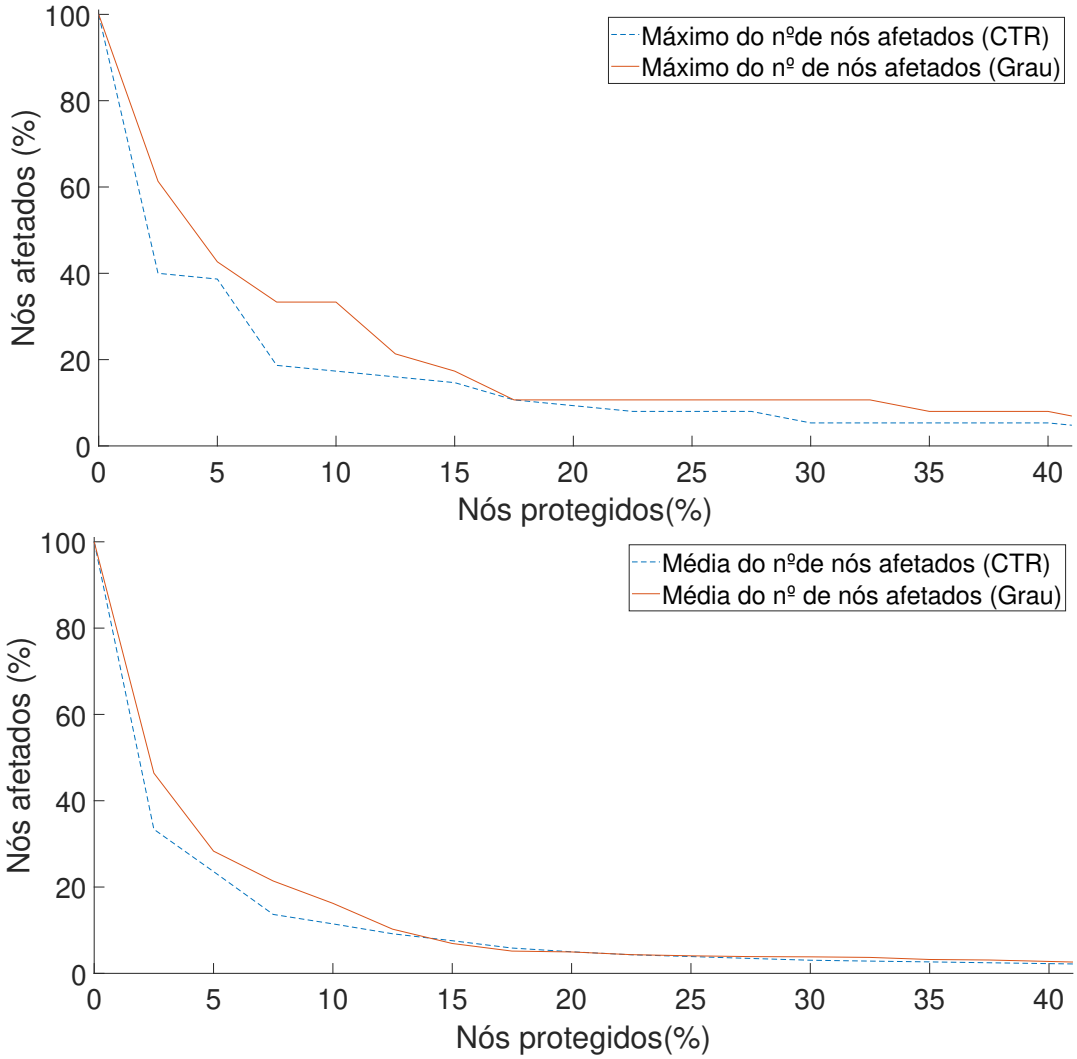

Figura A.18: Percentagem do número máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a rede da figura A.3.

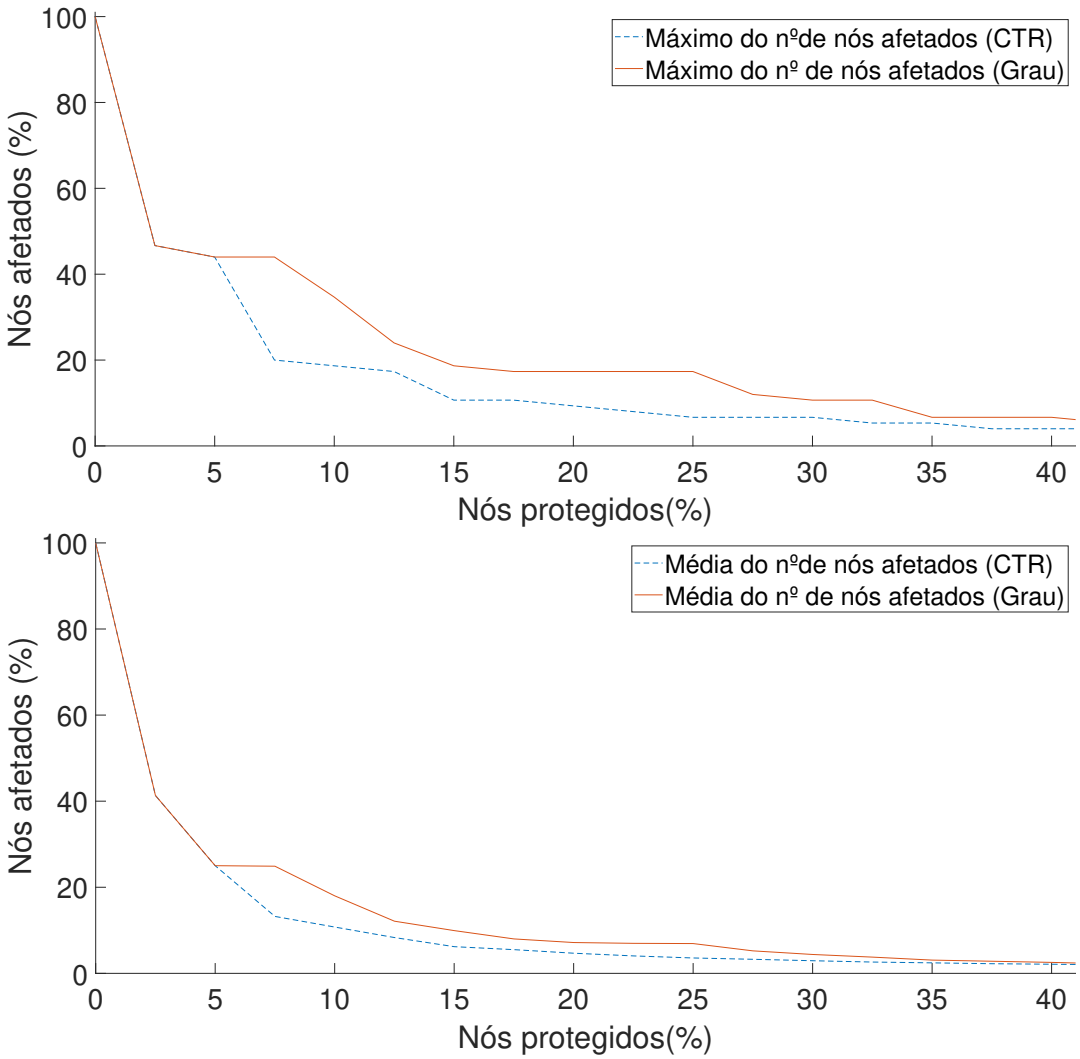

Figura A.19: Percentagem do número máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a árvore da figura A.4.

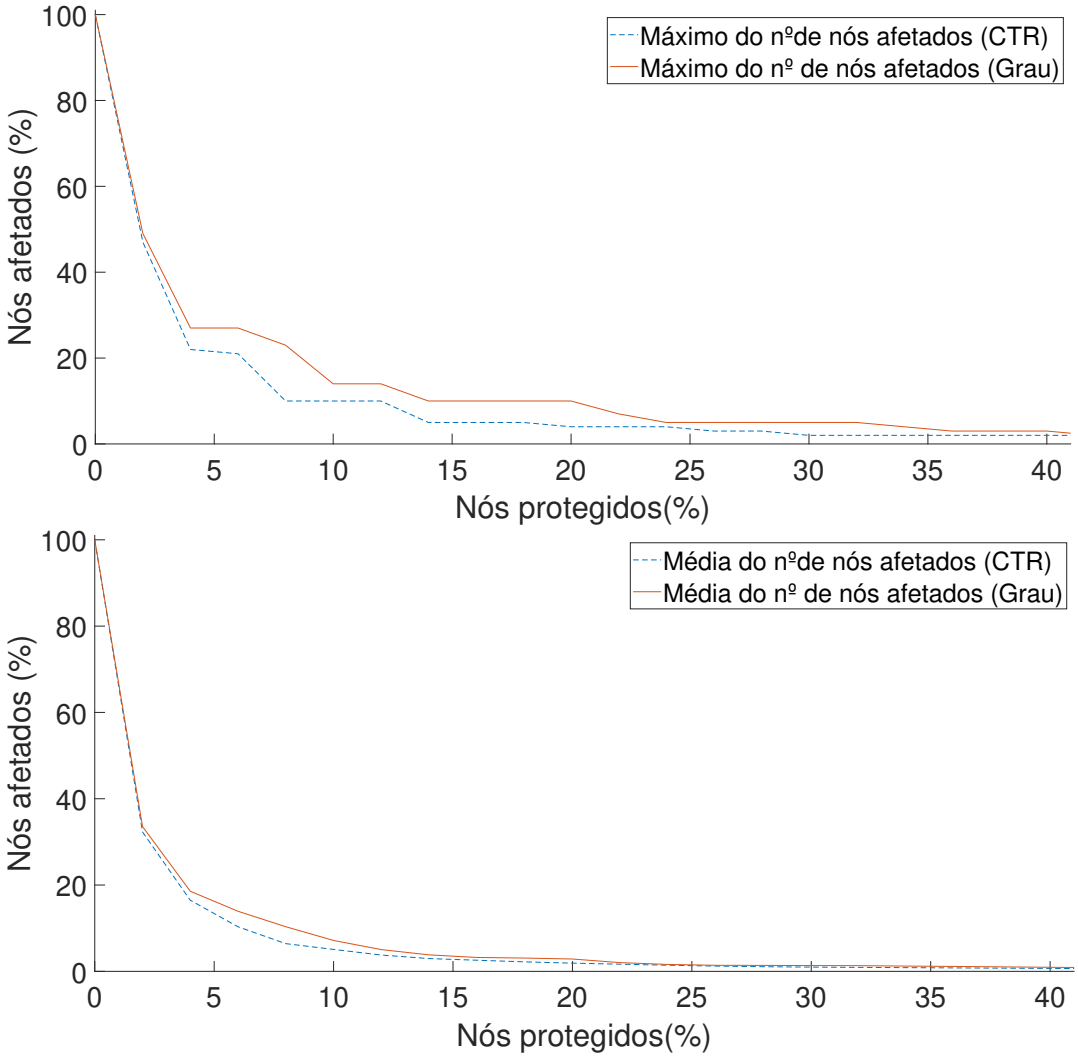

Figura A.20: Percentagem do número máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a árvore da figura A.5.

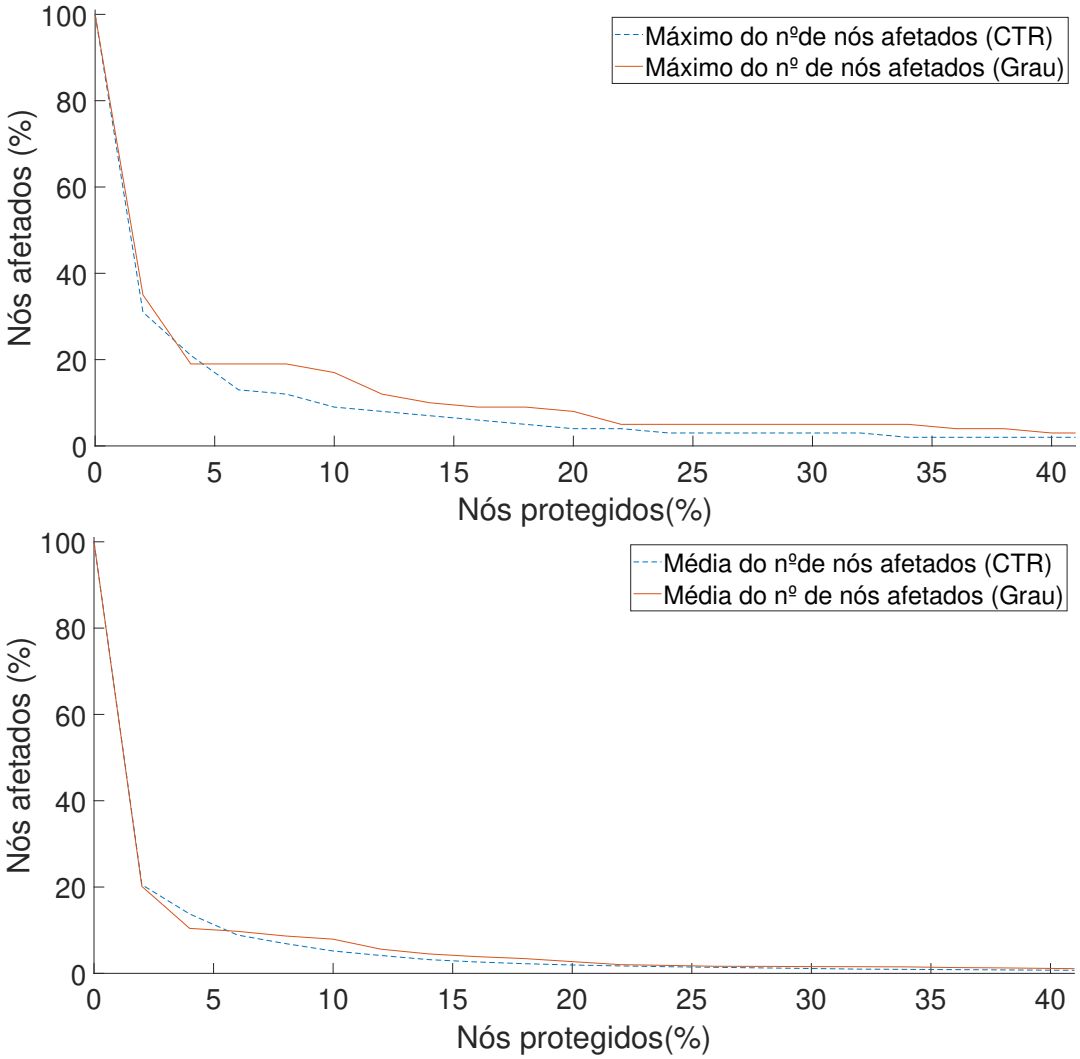

Figura A.21: Percentagem do número máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a árvore da figura A.6.

Apêndice B

Redes C

Apresentam-se de seguida resultados complementares relativos a um conjunto representativo das redes C em *http://steinlib.zib.de/testset.php*, nomeadamente as redes c02, c06 e c07. É de notar que como estas redes têm 500 nós foram omitidos os resultados obtidos com o algoritmo de Prim e apenas se apresenta a seguir a comparação do desempenho da rede quando os nós protegidos são obtidos pela centralidade ou são escolhidos entre os nós de maior grau.

B.1 Centralidade vs grau do nó

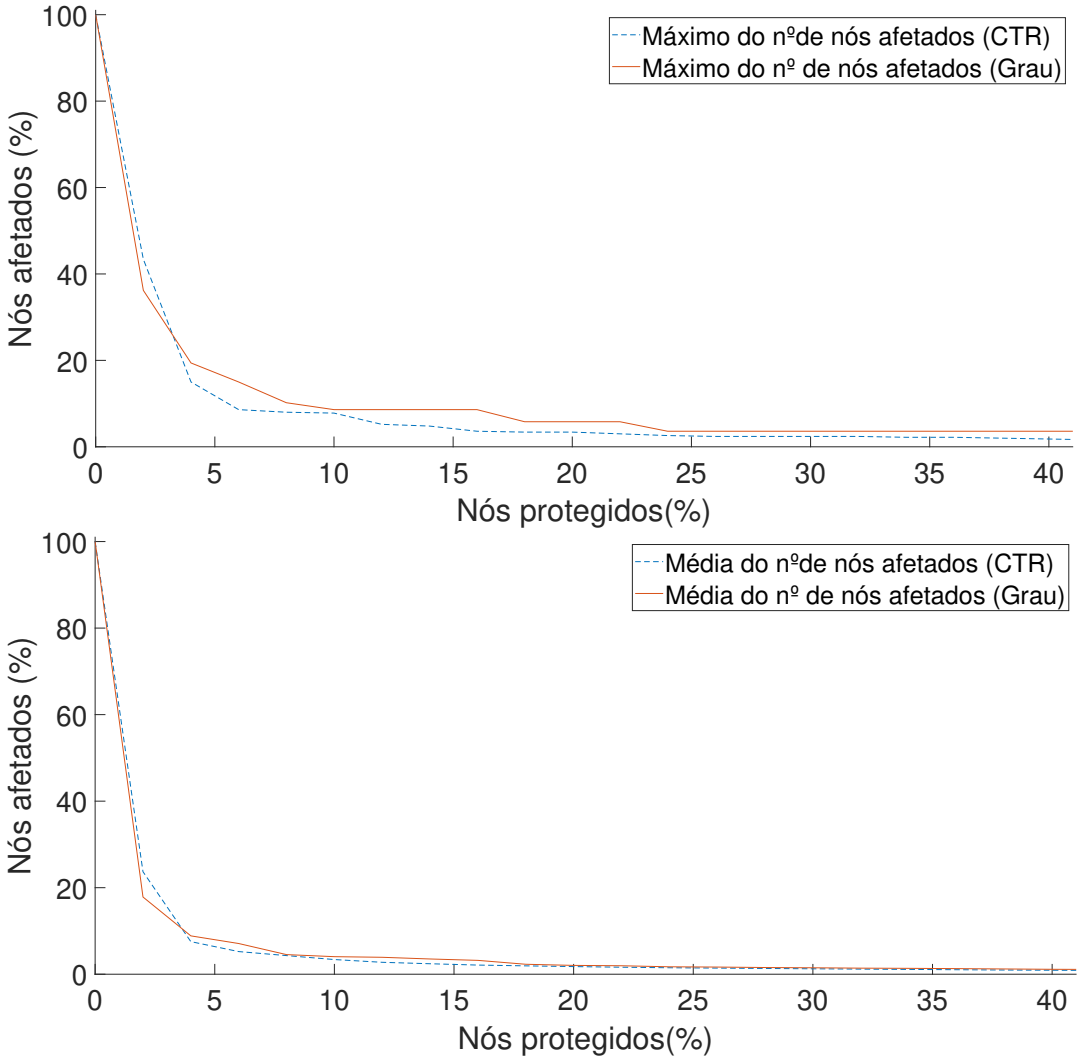

Figura B.1: Percentagem do número máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a rede c02 (após correr o Prim).

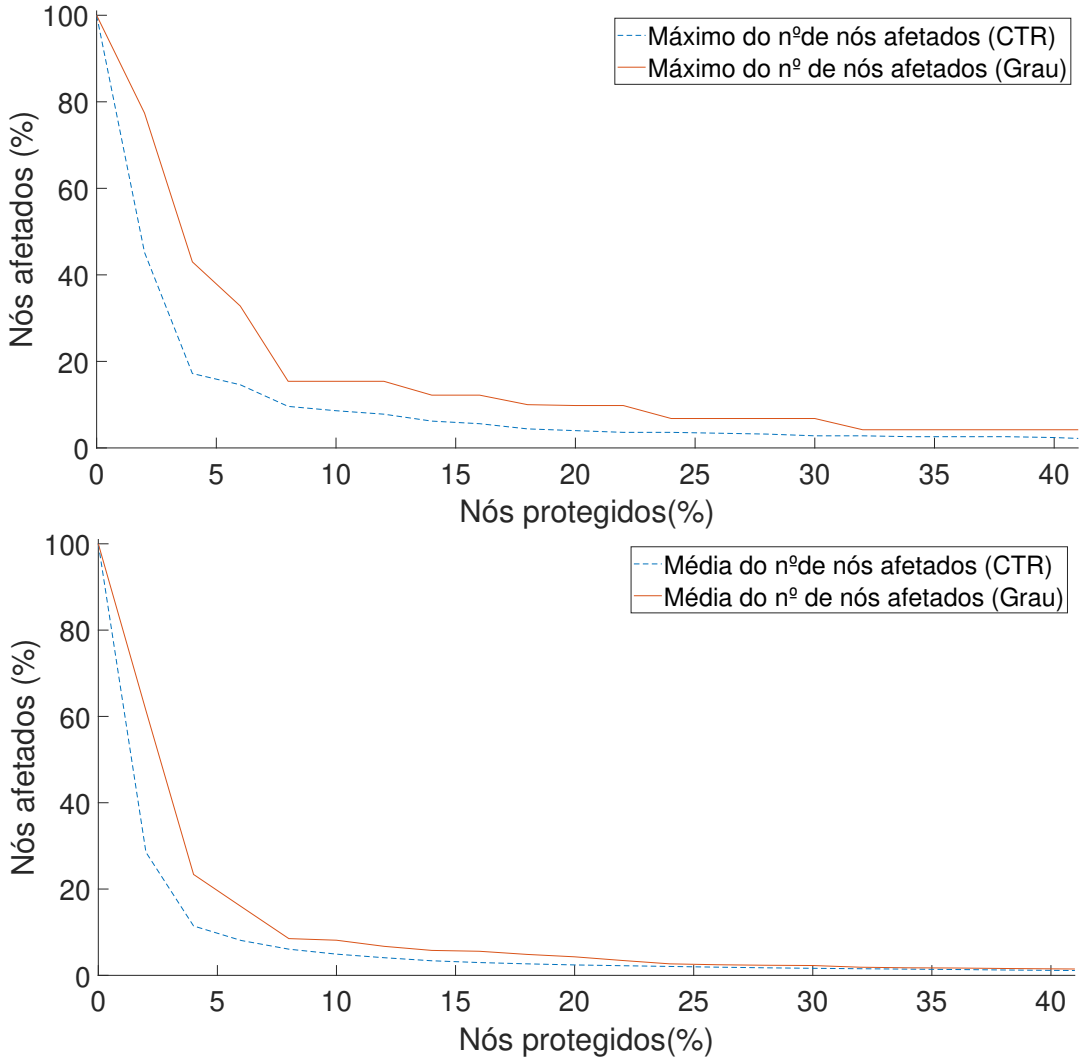

Figura B.2: Percentagem do número máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a rede c06 (após correr o Prim).

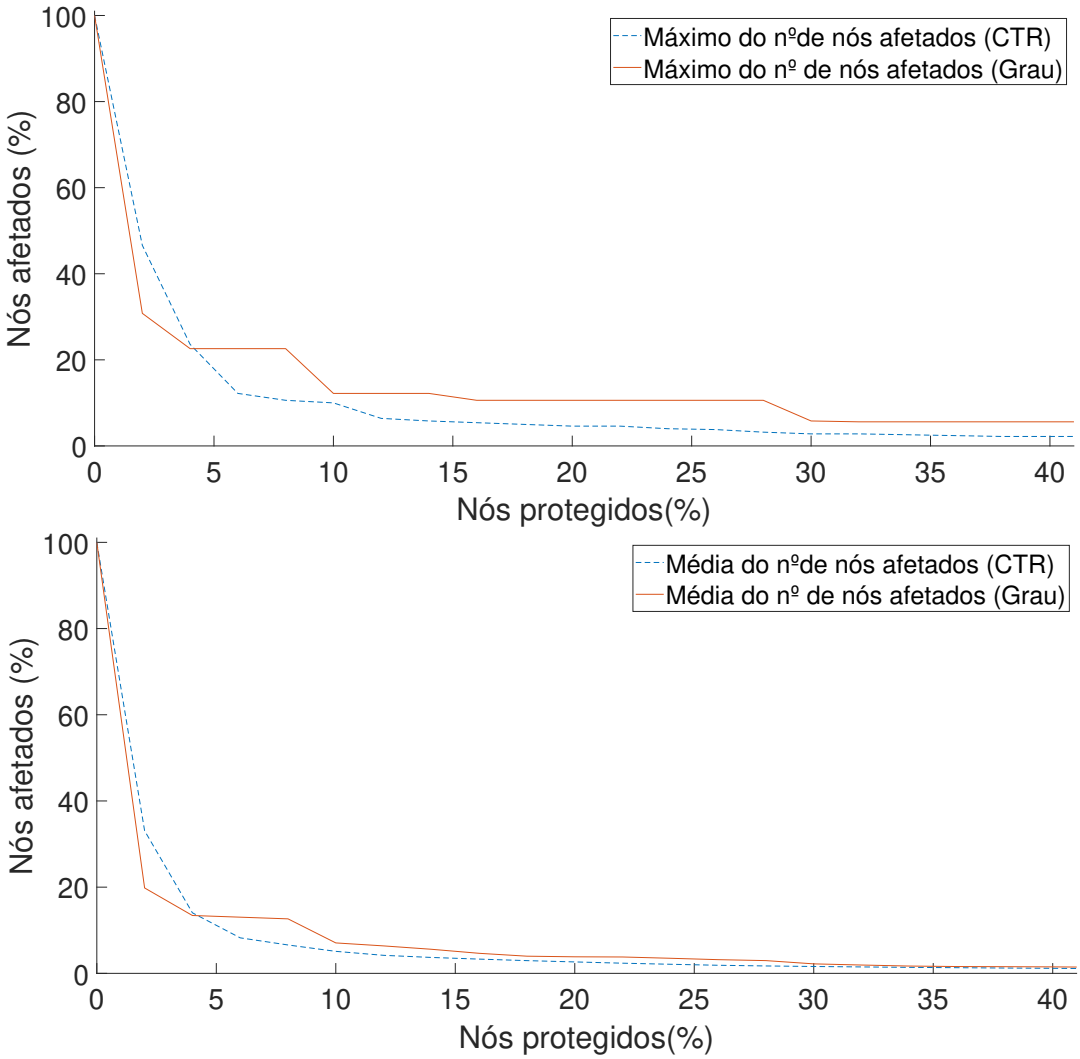

Figura B.3: Percentagem do número máximo (em cima) e médio (em baixo) de nós afetados em função da percentagem de nós protegidos para a rede c07 (após correr o Prim).