ISSN: 2526-1371

Estimando Centralidade de Percolação utilizando Amostragem e Teoria da Dimensão Vapnik-Chervonenkis

Alane Marie de Lima[1], Giovanne Marcelo dos Santos[2], André Luis Vignatti[1], Murilo Vicente Gonçalves da Silva[1]

[1]Departamento de Informática – Universidade Federal do Paraná (UFPR)
Caixa Postal 19.031 – 81531-980 – Curitiba – PR – Brasil

[2]Instituto de Matemática e Estatística – Universidade de São Paulo (USP)

{amlima,vignatti,murilo}@inf.ufpr.br, gsantos@ime.usp.br

Resumo. *Medidas de centralidade em redes quantificam a importância relativa de seus nós e conexões, e podem variar de acordo com o contexto em que se aplicam. A centralidade de percolação determina a importância de um nó em aplicações em que há um processo de infestação na rede. O melhor algoritmo exato conhecido para computá-la de maneira exata em uma rede com n nós e m conexões executa em tempo $\mathcal{O}(n^3)$. Neste trabalho apresentamos um algoritmo aleatorizado para estimar a centralidade de percolação de um dado nó da rede. O algoritmo proposto utiliza resultados da Teoria da Dimensão VC e Teorema da ϵ-amostra, e executa em tempo $\mathcal{O}(\max(n^2, (n+m)\frac{c}{\epsilon^2}\ln\frac{1}{\delta}))$ para grafos sem peso e $\mathcal{O}(\max(n^2, (m\log n)\frac{c}{\epsilon^2}\ln\frac{1}{\delta})$ para grafos com peso, apresentando um erro de no máximo ϵ com probabilidade $1-\delta$, para constante $c > 0$.*

Abstract. *In network analysis, a variety of centrality measures are used to quantify the relative importance of entities and connections of a system. Percolation centrality refers to the importance of a node in a network going through a contagious process. The fastest algorithm for computing this measure in a network of n nodes and m edges runs in $\mathcal{O}(n^3)$. In this work, we propose a randomized algorithm to approximate the percolation centrality of a given vertex in a network. Our algorithm relies on VC-Dimension Theory and ϵ-samples, and have running time $\mathcal{O}(\max(n^2, (n+m)\frac{c}{\epsilon^2}\ln\frac{1}{\delta}))$ for unweighted graphs and $\mathcal{O}(\max(n^2, (m\log n)\frac{c}{\epsilon^2}\ln\frac{1}{\delta})$ for weighted graphs, within a factor of ϵ for the error and with probability $1-\delta$, for a constant $c > 0$.*

Keywords: Percolation Centrality, Approximation Algorithm, VC-Dimension

Palavras-chave: Centralidade de Percolação, Algoritmo de Aproximação, Dimensão VC

1. Introdução

No estudo de redes complexas é comum quantificar a centralidade de nós e de conexões de uma rede a fim de determinar a importância relativa que estas entidades têm no sistema. Em particular, a centralidade de um nó pode ser definida em termos de características locais, como *grau*, ou globais, como *intermediação* ou *percolação*. O foco deste trabalho

é a medida de *centralidade de percolação*, proposta por [Piraveenan et al. 2013], que mede a importância de um nó no contexto em que há um processo de infestação da rede (e.g., transmissão de doenças ou divulgação de notícias falsas).

O estudo do fenômeno de percolação em um sistema foi introduzido por [Broadbent e Hammersley 1957], que atribuíram a aleatoriedade ao processo de passagem de um dado fluido, ou seja, cada parte do "meio" onde passa o fluido possui uma probabilidade de transmiti-lo ou não (estado de percolação). No caso de redes, a centralidade de percolação de um nó generaliza a centralidade de intermediação do mesmo. Enquanto esta última se refere a quantidade de caminhos mínimos que passam pelo nó, a centralidade de percolação, além desta quantidade, leva em consideração o estado de percolação dos nós da rede.

Os melhores algoritmos para o cálculo exato da centralidade de intermediação dos nós da rede dependem da computação de todos os seus caminhos mínimos [Riondato e Kornaropoulos 2016] e, consequentemente, o mesmo vale para o cálculo da centralidade de percolação. Até onde os autores sabem, isso se aplica mesmo ao caso de se calcular a centralidade de percolação de um único nó da rede. O melhor algoritmo conhecido para o problema possui complexidade de tempo $\mathcal{O}(n^3)$, e pode ser reduzida para $\mathcal{O}(nm)$ por meio de uma adaptação do algoritmo proposto por [Brandes 2001] em uma versão mais simples do problema que não é a que lidamos aqui. Segundo [Riondato e Kornaropoulos 2016], em redes de larga escala, este algoritmo é custoso e, além disso, resultados aproximados respeitando parâmetros de qualidade e confiança podem ser suficientes na prática. Em nosso trabalho, propomos um algoritmo para *estimar* a centralidade de percolação de um dado vértice, satisfazendo parâmetros de qualidade e confiança arbitrários fornecidos juntos com a entrada. Este trabalho se baseia na abordagem de [Riondato e Kornaropoulos 2016] para estimar a centralidade de intermediação usando resultados da teoria da Dimensão Vapnik-Chervonenkis (VC) e ϵ-amostra. O algoritmo proposto executa em tempo $\mathcal{O}(\max(n^2, (n+m)\frac{c}{\epsilon^2}\ln\frac{1}{\delta}))$ para grafos sem peso e $\mathcal{O}(\max(n^2, (m\log n)\frac{c}{\epsilon^2}\ln\frac{1}{\delta})$ para grafos com peso, apresentando um erro de no máximo ϵ com probabilidade $1-\delta$, para constante $c > 0$. Na prática os parâmetros ϵ e δ podem ser tratados como constantes, sendo que tipicamente fazendo $\delta = 0.1$ e $\epsilon = 0.015$, obtém-se resultados bastante satisfatórios [Riondato e Kornaropoulos 2016].

2. Preliminares Matemáticos

Nesta seção, apresentamos as definições, notação e resultados utilizados neste trabalho.

2.1. Grafos e Centralidade de Percolação

Dado um grafo $G = (V, E)$ (direcionado ou não), os estados de percolação $x_v, \forall v \in V$, e $(u, w) \in V^2$, seja S_{uw} o conjunto de todos os caminhos mínimos entre u e w, e $\sigma_{uw} = |S_{uw}|$. A quantidade de caminhos mínimos entre u e w em que $v \in V$ é vértice interno é denotada por $\sigma_{uw}(v)$, para $u \neq v \neq w$. O conjunto de vértices internos de um caminho mínimo $p_{uw} \in S_{uw}$ é denotado por $Int(p_{uw})$. O conjunto de *predecessores* de w em p_{uw} é definido como $P_u(w) = \{s | s \in V \text{ e } (s, w) \in E_{p_{uw}}\}$, onde $E_{p_{uw}}$ denota o conjunto de arestas de p_{uw}. Seja $0 \leq x_v \leq 1$ o estado de percolação do vértice $v \in V$. Dizemos que

v está totalmente percolado se $x_v = 1$, não percolado se $x_v = 0$ e parcialmente percolado se $0 < x < 1$. A centralidade de percolação $p(v)$ é definida abaixo.

Definição 1. Seja $R(x) = \max\{x, 0\}$. Dado grafo $G = (V, E)$ e estados de percolação $x_v, \forall v \in V$, a *centralidade de percolação* $p(v)$ do vértice $v \in V$ é $p(v) = \sum_{\substack{(u,w)\in V^2 \\ u\neq v\neq w}} \frac{\sigma_{uw}(v)}{\sigma_{uw}} \frac{R(x_u - x_w)}{\sum_{\substack{(f,d)\in V^2 \\ f\neq v\neq d}} R(x_f - x_d)}$.

A Definição 1 segue a notação original. Para simplificar, seja $r_{uw} = R(x_u - x_w)$ e $S(v) = \sum_{\substack{(f,d)\in V^2 \\ f\neq v\neq d}} R(x_f - x_d)$. Assim, $p(v) = \frac{1}{S(v)} \sum_{\substack{(u,w)\in V^2 \\ u\neq v\neq w}} \frac{\sigma_{uw}(v) r_{uw}}{\sigma_{uw}}$.

2.2. Dimensão Vapnik-Chervonenkis

Em algoritmos que utilizam amostragem, a análise de complexidade de amostra relaciona o tamanho mínimo necessário que uma amostra aleatória deve ter para que os resultados aproximados obtidos estejam consistentes com os parâmetros de confiança e qualidade desejados [Mitzenmacher e Upfal 2017]. Limitantes superiores para a Dimensão VC do espaço de intervalos (definido abaixo) modelado para um determinado problema estabelecem limitantes sobre o tamanho da amostra com essas características. A definição de ϵ-amostra formaliza isso (Definição 3). Mais detalhes acerca das definições e resultados apresentados a seguir podem ser encontrados em [Shalev-Shwartz e Ben-David 2014].

Um *espaço de intervalos* é um par $R = (X, \mathcal{I})$, onde X é um domínio (finito ou infinito) e $\mathcal{I}$ é uma coleção de subconjuntos de X, denominados *intervalos*. Dado $S \subseteq X$, a *projeção* de $\mathcal{I}$ em S é o conjunto $\mathcal{I}_S = \{S \cap I | I \in \mathcal{I}\}$. Se $|\mathcal{I}_S| = 2^{|S|}$ então dizemos que S é *despedaçado* por $\mathcal{I}$.

Definição 2. A Dimensão VC de um espaço de intervalos $R = (X, \mathcal{I})$, denotada por $VCDim(R)$, é $VCDim(R) = \max\{d : \exists S \subseteq X$ tal que $|S| = d$ e $|\mathcal{I}_S| = 2^d\}$.

Definição 3. Sejam $R = (X, \mathcal{I})$ um espaço de intervalos, e $\mathcal{D}$ uma distribuição de probabilidade em X. Um conjunto $S \subseteq X$ é uma ϵ-amostra para X com respeito a $\mathcal{D}$ se para todos os conjuntos $I \in \mathcal{I}$, $|\Pr_{\mathcal{D}}(I) - |S \cap I|/|S|| \leq \epsilon$, onde $\Pr_{\mathcal{D}}(I)$ é a probabilidade de um elemento pertencer ao intervalo I de acordo com a distribuição $\mathcal{D}$.

Teorema 1. (prova em [Li et al. 2001]) Sejam $R = (X, \mathcal{I})$ um espaço de intervalos tal que $VCDim(R) \leq d$ e $\mathcal{D}$ uma distribuição de probabilidade em X. Para qualquer $\epsilon, \delta \in [0, 1]$, existe $m = \frac{c}{\epsilon^2}\left(d + \ln \frac{1}{\delta}\right)$ tal que uma amostra aleatória com respeito a $\mathcal{D}$ de tamanho m é uma ϵ-amostra para X com probabilidade $1 - \delta$, para constante $c > 0$.

Segundo [Löffler e Phillips 2009], a constante c é aproximadamente 0.5. A seguir, demonstramos que $VCDim(R) = 0$ se $|\mathcal{I}| = 1$.

Teorema 2. Um espaço de intervalos $R = (X, \mathcal{I})$ com $|\mathcal{I}| = 1$ possui $VCDim(R) = 0$.

Demonstração. Seja $VCDim(R) = k$, onde $k \in \mathbb{N}$. Então existe um conjunto $S \subseteq X$ de tamanho k que é despedaçado por $\mathcal{I}$, isto é, existem $2^{|S|}$ intervalos distintos em $\mathcal{I}$ tais que $|\mathcal{I}_S| = 2^k$. Por outro lado, $|\mathcal{I}| = 1$, e então, como $2^k = 1$, temos que $k = 0$. □

3. Algoritmo Proposto

Nesta seção, modelamos o problema em termos de um espaço de intervalos e apresentamos um algoritmo cuja corretude e tempo de execução sustentam-se no Teorema 1.

3.1. Espaço de Intervalos sobre Caminhos Mínimos

Seja S_G o conjunto de todos os caminhos mínimos em G, isto é, $S_G = \bigcup_{(u,w)\in V^2:u\neq w} S_{uw}$. Dado $v \in V$, cada caminho $p_{uw} \in S_G$ é amostrado com probabilidade $\mathcal{D}^v(p_{uw}) = \frac{r_{uw}}{S(v)\sigma_{uw}}$. Para cada vértice v, seja τ_v o conjunto de caminhos mínimos $p_{uw} \in S_G$ em que v é interno, isto é, $\tau_v = \{p_{uw} | p_{uw} \in S_G, v \in Int(p_{uw})$ e $u, w \in V\}$. Seja também $R^v = (S_G, \mathcal{I})$ o espaço de intervalos definido para v onde $\mathcal{I} = \{\tau_v\}$. Pelo Teorema 2, como $|\mathcal{I}| = 1$, então $VCDim(R^v) = 0$.

3.2. Algoritmo

Inicialmente, a matriz de diferenças R, de dimensão $(n+2) \times (n+1)$, é pré-computada no Algoritmo 1. Cada posição $R[u][w]$ corresponde ao valor r_{uw}, para $(u, w) \in V^2$, como descrito na Definição 1. As posições $R[n+1][w]$ e $R[w][n+1]$ contém os valores $\sum_{u\in V} r_{uw}$ e $\sum_{u\in V} r_{wu}$, respectivamente. A posição $R[n+1][n+1]$ contém a soma $\sum_{(u,w)\in V^2:u\neq w} r_{uw}$, enquanto $R[n+2][v]$ contém o valor $S(v)$.

Algoritmo 1: PRECALCULADIFERENCAS(x)

Entrada: O estado de percolação x_v para todo vértice $v \in V$.
Saída: Matriz R de tamanho $(n+2) \times (n+1)$.

```
1  R[i][j] ← 0,  ∀(i, j) ∈ (n + 2) × (n + 1)
2  para i ← 1 até n + 1 faça
3      para j ← 1 até n + 1 faça
4          dif ← (x[i] − x[j]);
5          se dif > 0 então
6              R[i][n+1] ← R[i][n+1] + dif,    R[n+1][j] ← R[n+1][j] + dif;
7              R[i][j] ← dif,    R[n + 1][n + 1] ← R[n + 1][n + 1] + dif;
8  para i ← 1 até n faça
9      R[n + 2][i] ← R[n + 1][n + 1] − R[n + 1][i] − R[i][n + 1];
10 retorna R
```

No Algoritmo 2 é apresentado o método proposto, dado um grafo com um estado de percolação x_v, para todo $v \in V$, e os parâmetros de erro e confiabilidade $\epsilon, \delta \in [0, 1]$, respectivamente. Em linhas gerais, dois vértices u e w são amostrados de acordo com os valores de R (linha 3), e em seguida um caminho $p_{uw} \in S_{uw}$ é amostrado uniformemente nas linhas 5–10. Se o vértice v é interno a p_{uw}, então $\tilde{p}(v)$ é incrementado em $1/r$ (passo que corresponde a média amostral, demonstrada no Teorema 3). O algoritmo utilizado na linha 4 calcula σ_{uz}, σ_{ut} e $P_u(t)$, necessários para o passo da linha 8.

Teorema 3. Para uma amostra S de tamanho $r = \frac{c}{\epsilon^2} \ln \frac{1}{\delta}$ e constante $c > 0$, a saída do Algoritmo 2 possui erro de no máximo ϵ tal que $\Pr(|p(v) - \tilde{p}(v)| \leq \epsilon) \geq$

Algoritmo 2: CENTRALIDADEDEPERCOLACAO(G,x,v,ϵ,δ)

Entrada: Grafo $G = (V, E)$ com $n = |V|$, estados de percolação x, vértice $v \in V$, erro ϵ, confiança δ.
Saída: Aproximação $\tilde{p}(v)$ para a centralidade de percolação do vértice v.

1 $R \leftarrow$ preCalculaDiferencas(x), $\quad r \leftarrow \frac{c}{\epsilon^2} \ln \frac{1}{\delta}$, $\quad \tilde{p}(v) \leftarrow 0$;
2 **para** *cada* $i \leftarrow 1$ *até* r **faça**
3 amostre w com probabilidade $\frac{R[n+1][w]}{R[n+2][w]}$ e u com probabilidade $\frac{R[u][w]}{R[n+1][w]}$;
4 $S_{uw} \leftarrow$ todosCaminhosMínimos(u, w);
5 **se** $S_{uw} \neq \emptyset$ **então**
6 $t \leftarrow w$;
7 **enquanto** $t \neq u$ **faca**
8 amostre $z \in P_u(t)$ com probabilidade $\frac{\sigma_{uz}}{\sigma_{ut}}$;
9 **se** $z \neq u$ *e* $z = v$ **então** $\tilde{p}(z) \leftarrow \tilde{p}(z) + 1/r$;
10 $t \leftarrow z$;
11 **retorna** $\tilde{p}(v)$

$1 - \delta$. *Demonstração.* O Algoritmo 2 amostra um caminho mínimo p_{uw} com probabilidade $\mathcal{D}^v(p_{uw})$, dada a maneira como os vértices u e w são selecionados e que o laço da linhas 7–10 amostra um caminho mínimo p_{uw} de maneira uniforme sobre o conjunto S_{uw} (Lema 5 em [Riondato e Kornaropoulos 2016]). Seja $\tilde{p}(v)$ a estimativa para a centralidade de percolação do vértice v e seja $S = \{p_1, ..., p_r\}$ o conjunto de r elementos de S_G amostrados pelo Algoritmo 2. Temos que $\Pr_{\mathcal{D}}(\tau_v)$ é igual a $\sum_{p_{uw} \in \tau_v} \frac{r_{uw}}{S(v)} \frac{1}{\sigma_{uw}} = \sum_{\substack{(u,w) \in V^2 \\ u \neq v \neq w}} \sum_{p \in S_{uw}} \frac{r_{uw}}{S(v)} \frac{1}{\sigma_{uw}} = \sum_{\substack{(u,w) \in V^2 \\ u \neq v \neq w}} \frac{r_{uw}}{S(v)} \frac{\sigma_{uw}(v)}{\sigma_{uw}} = p(v)$. Temos também que $\tilde{p}(v)$ corresponde a média amostral dada por $\tilde{p}(v) = \frac{1}{r} \sum_{p \in S} 1_{\tau_v(p)} = \frac{|S \cap \tau_v|}{|S|}$, onde $1_{\tau_v(p)}$ é a função indicadora para o conjunto τ_v (o valor de $1_{\tau_v(p)}$ é 1 se $p \in \tau_v$ e 0 caso contrário). Como $\left| \Pr_{\mathcal{D}}(\tau_v) - \frac{|S \cap \tau_v|}{|S|} \right| = |p(v) - \tilde{p}(v)|$, então podemos aplicar os Teoremas 1 e 2 para obter $d = VCDim(R^v) = 0$ e obter que S é uma ϵ-amostra de tamanho $r = \frac{c}{\epsilon^2} \ln \frac{1}{\delta}$ tal que $\Pr(|p(v) - \tilde{p}(v)| \leq \epsilon) \geq 1 - \delta$. □

Teorema 4. O Algoritmo 2 executa em tempo $\mathcal{O}(\max(n^2, (n + m)\frac{c}{\epsilon^2} \ln \frac{1}{\delta}))$ para grafos sem peso e em tempo $\mathcal{O}(\max(n^2, (m \log n)\frac{c}{\epsilon^2} \ln \frac{1}{\delta}))$ para grafos com peso. *Demonstração.* Seja T_R o tempo de execução do Algoritmo 1. Note que $T_R = \Theta(n^2)$. Para a amostragem utilizada nas linhas 3 e 8 utilizamos o algoritmo de [Vose 1991], de tempo linear. Sejam T_{su} e T_{sw} os tempos de execução da amostragem do vértice u e do vértice w, respectivamente. Então $T_{su} = T_{sw} = \mathcal{O}(n)$. Sejam T_p o tempo de execução do trecho das linhas 7–10 e $T_{S_{uw}}$ o tempo de execução da linha 4. Como $|P_u(w)| \leq d_G(w)$, onde $d_G(w)$ denota o grau do vértice w em G, e o laço do trecho das linhas 7–10 é executado no máximo n vezes caso o caminho amostrado passe por todos os vértices do grafo, então $T_p = \sum_{v \in V} d_G(v) = 2m = \mathcal{O}(m)$. O algoritmo executado na linha 4 é Dijkstra ou BFS, dependendo do grafo ter ou não pesos. Portanto $T_{S_{uw}}$ é, respectivamente, $\mathcal{O}(m \log n)$ ou $\mathcal{O}(n + m)$. Como o laço principal

das linhas 2–10 é executado r vezes, e $r \in \mathcal{O}(\frac{1}{\epsilon^2} \ln \frac{1}{\delta})$, então o custo total do Algoritmo 2 é $\mathcal{O}(\max(T_R, T_{su}, T_p, T_{S_{uw}})) = \mathcal{O}(\max(n^2, r(n, m, T_{S_{uw}})))$, que corresponde a $\mathcal{O}(\max(n^2, \frac{c}{\epsilon^2} \ln \frac{1}{\delta}(n+m)))$ em grafos sem peso e $\mathcal{O}(\max(n^2, \frac{c}{\epsilon^2} \ln \frac{1}{\delta}(m \log n)))$ em grafos com peso. □

4. Conclusão

Apresentamos um algoritmo baseado em amostragem e conceitos da teoria da Dimensão VC para o cálculo da centralidade de percolação de um vértice que permite a escolha entre baixo tempo de execução ou alta precisão na estimativa por meio parâmetros ϵ e δ. Para o caso de ϵ e δ constantes, este é o algoritmo mais rápido conhecido para o problema. Observamos que o algoritmo proposto neste trabalho, quando modificado para o problema mais geral de estimar a centralidade de percolação de todos os vértices do grafo, torna-se menos eficiente que o melhor algoritmo exato conhecido. Dessa forma, como trabalhos futuros, outras técnicas e resultados da área de complexidade de amostra como médias de Rademacher poderiam ser utilizadas para reduzir o número de amostras necessárias, servindo como base no projeto de um algoritmo de tempo $o(n^3)$ para calcular a centralidade em todos os vértices do grafo. Finalmente, uma análise experimental comparativa dos algoritmos seria útil para verificar o comportamento do algoritmo proposto na prática e como usá-lo para equilibrar tempo de execução com os parâmetros de qualidade e confiança desejados.

Referências

Brandes, U. (2001). A faster algorithm for betweenness centrality. *Journal of Mathematical Sociology*, 25(163):163–177.

Broadbent, S. e Hammersley, J. M. (1957). Percolation processes: I. crystals and mazes. *Math. Proc. of the Cambridge Philosophical Society*, 53(3):629–641.

Li, Y., Long, P. M., e Srinivasan, A. (2001). Improved bounds on the sample complexity of learning. *Journal of Computer and System Sciences*, 62(3):516–527.

Löffler, M. e Phillips, J. M. (2009). Shape fitting on point sets with probability distributions. In *European Symposium on Algorithms*, pages 313–324. Springer.

Mitzenmacher, M. e Upfal, E. (2017). *Probability and computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge University Press.

Piraveenan, M., Prokopenko, M., e Hossain, L. (2013). Percolation centrality: Quantifying graph-theoretic impact of nodes during percolation in networks. *PLOS ONE*, 8(1):1–14.

Riondato, M. e Kornaropoulos, E. M. (2016). Fast approximation of betweenness centrality through sampling. *Data Mining and Knowledge Discovery*, 30(2):438–475.

Shalev-Shwartz, S. e Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.

Vose, M. D. (1991). A linear algorithm for generating random numbers with a given distribution. *IEEE Transactions on software engineering*, 17(9):972–975.