Identificação de Nós Maliciosos em Redes Complexas Baseada em Visões Locais

Grazielle Vernize e Luiz Carlos Pessoa Albini

***Resumo*— Muitos sistemas sociais, biológicos e de informação podem ser descritos através de modelos de redes complexas. Redes complexas apresentam características estruturais comuns, como as propriedades de mundo pequeno e livre de escala. Entretanto, nós nessas redes podem não cooperar uns com os outros, apresentando um comportamento egoísta para economizar seus recursos. Além disso, a presença de nós maliciosos pode prejudicar a operação da rede, pois eles podem atacar a rede de diferentes maneiras como inserir, modificar ou eliminar informações. Algoritmos de análise de confiança são um incentivo para encorajar nós egoístas a colaborarem e isolam nós maliciosos. Nós que evitam colaborar ou apresentam um comportamento egoísta possuem valores de confiança baixos e podem ser penalizados, pois os outros nós na rede tendem a cooperar somente com nós com alto valor de confiança. Este artigo apresenta um algoritmo que calcula o número de nós maliciosos e/ou nós egoístas em uma rede, baseado na visão local de confiança que cada nó tem em relação aos seus vizinhos. O algoritmo identifica exatamente quais são esses nós. Resultados de simulações em quatro redes complexas reais demonstram a efetividade da abordagem proposta. Na verdade, os resultados apresentam uma margem de erro inferior a 15% para até 35000 nós maliciosos ou egoístas em redes de 70000 nós. Se o número de nós maliciosos nestas redes está abaixo de 5000, a margem de erro é de cerca de um nó.**

***Palavras-Chave*— Redes complexas, confiança, nós maliciosos.**

***Abstract*— Many social, biological and information systems can be described through complex network models. Complex networks display common structural features, such as the small-world and scale-free properties. However, nodes in these networks may not cooperate with each other, presenting a selfish behavior to preserve their resources. Furthermore, the presence of malicious nodes can damage the network operation, as they may attack the network in several different ways, like inserting, modifying or eliminating information in the network. Trust evaluation algorithms are a useful incentive for encouraging selfish nodes to collaborate and for isolating malicious ones. Nodes which refrain from cooperation or present a malicious behavior get a lower trust value and may be penalized as other nodes tend to cooperate only with highly trusted ones. This paper presents an algorithm to calculate the number of malicious and/or selfish nodes in a network based on local trust views that each node has about their neighbors. The algorithm points out to the network manager exactly which nodes they are. Simulation results over four real complex networks demonstrate the effectiveness of the proposed approach. In fact, it presents an error margin smaller than 15% for up to 35000 malicious or selfish nodes in networks of 70000 nodes. If the number of malicious nodes goes under 5000 for the same networks, the error margin is around one node.**

***Keywords*— Complex networks, trust, malicious nodes.**

Grazielle Vernize e Luiz Carlos Pessoa Albini, Departamento de Informática, Universidade Federal do Paraná, Curitiba-PR, Brasil, E-mails: {gvernize, albini}@inf.ufpr.br

I. INTRODUÇÃO

Muitos sistemas na natureza podem ser descritos através de modelos de redes complexas [1]: a Internet é uma rede de roteadores ou domínios, o cérebro é uma rede de neurônios, uma organização é uma rede pessoas, entre outros. As redes no mundo real podem ser classificadas em quatro categorias [2]: redes sociais, redes de informação, redes biológicas e redes tecnológicas. Uma rede social é um conjunto de pessoas ou grupos de pessoas com algum padrão de contatos ou algumas interações entre elas. Redes de informação ou redes de conhecimento reúnem pessoas e organizações para o intercâmbio de informações. Redes biológicas podem representar qualquer sistema biológico, como o sistema metabólico. Por fim, redes tecnológicas são redes artificiais tipicamente designadas para a distribuição de alguma comodidade ou algum recurso.

O estudo de redes complexas tornou-se um foco comum de muitos ramos da ciência [3]. Estas redes apresentam características estruturais comuns, como as propriedades de mundo-pequeno (*small-world*) e livre de escala (*scale-free*) [4]. Redes mundo-pequeno possuem uma distância média pequena entre os nós e alto grau de agrupamento, enquanto que redes livre de escala são caracterizadas por uma distribuição altamente heterogênea dos graus dos seus nós.

Nesse tipo de rede, não existe uma boa razão para assumir que os nós irão cooperar e prover serviços uns para os outros. Na verdade, o contrário é mais provável. Alguns nós podem apresentar um comportamento egoísta preservando seus recursos ao invés de usá-los para realizar tarefas para outros nós. Além disso, a presença de nós maliciosos em uma rede também pode prejudicar sua operação, pois eles podem interferir e degradar o seu desempenho. Nós com comportamento malicioso tentam atacar a rede de diversas maneiras possíveis, como inserir, modificar ou eliminar informações. Neste artigo, somente o termo malicioso(s) será usado, pois ele engloba os comportamentos egoísta e malicioso (atacante).

Cálculos de confiança tentam prevenir esses tipos de comportamento. Nós que não cooperam ou não funcionam corretamente devem possuir um valor de confiança baixo. Esses nós podem inclusive ser penalizados, pois os outros nós tendem a cooperar com aqueles que possuem um valor de confiança alto [5]. O estabelecimento de relações de confiança entre os nós participantes da rede pode permitir a utilização de um sistema de métricas de confiança da rede [6], visando desestimular comportamentos egoístas ou maliciosos. Exemplos de como usar confiança em redes sociais podem ser encontrados em [7], [8], [9]. Exemplos do uso de modelos do estabelecimento da confiança em redes de informação podem ser encontrados em [10], [11].

Em redes tecnológicas o nível de confiança é definido como uma probabilidade de confiança variando de 0 (não confia) até 1 (confia) [12]. Grafo de confiança ou grafo de confiabilidade é um grafo dirigido e ponderado. Nesse grafo, uma aresta $a \xrightarrow{x} b$ significa que o nó a tem um valor x de nível de confiança em b. Muitos trabalhos sobre confiança podem ser encontrados na literatura, como: [13], [14], [15], [16], [17].

Este artigo apresenta um algoritmo para calcular o número de nós maliciosos em uma rede baseado em um dado estado do grafo de confiança da rede. Além disso, o algoritmo indica ao administrador da rede exatamente quais são esses nós, não apenas a quantidade de nós maliciosos. O grafo de confiança é uma visão local que cada nó possui em relação aos seus vizinhos. Baseados nessas evidências, eles declaram se confiam ou não em cada um de seus vizinhos. A visão local de cada nó é uma coleção desses valores. Não faz parte do escopo deste artigo gerar o grafo de confiança, mas se basear nas observações locais de confiança do grafo de confiança para dizer quem são os nós maliciosos.

O algoritmo é dividido em duas partes. A primeira parte usa o algoritmo de Tarjan [18] para encontrar as componentes fortemente conexas. Nós que confiam nos seus vizinhos ficam na mesma componente. Então, o algoritmo constrói um grafo tendo as componentes como vértices. A segunda parte do algoritmo consiste em classificar as componentes fortemente conexas em corretas ou maliciosas usando um algoritmo de coloração com um número mínimo de cores [19].

O algoritmo foi avaliado através de simulações em quatro redes complexas reais. As redes são grafos de redes sociais disponíveis em *SNAP: Stanford Network Analysis Platform*. Como será apresentado, o algoritmo é extremamente preciso. Nessas redes, ele apresenta uma margem de erro inferior a 15% quando o número de nós maliciosos no sistema é menor que 50% da rede inteira, i.e. cerca de 35000 nós maliciosos. Se o número de nós maliciosos é abaixo de 20% (aproximadamente 15000 nós maliciosos), a margem de erro fica abaixo de 7%. Se o número de nós maliciosos fica abaixo de 5000, a margem de erro é abaixo de 0.5%. Recordando que o administrador do sistema pode usar esses resultados para reparar ou remover as unidades apontadas como maliciosas e, então, reexecutar o algoritmo para uma nova checagem.

O restante deste artigo está organizado da seguinte forma: a Seção II relata as heurísticas utilizadas para aproximar o número de nós maliciosos; a Seção III detalha o algoritmo proposto; a Seção IV contém os resultados das simulações; a Seção V conclui o artigo e sugere direções futuras.

II. Heurísticas

Esta Seção descreve as heurísticas utilizadas para aproximar a quantidade de nós maliciosos em uma rede. A Subseção II-A relata o algoritmo de Tarjan para encontrar as componentes fortemente conexas em um grafo. A Subseção II-B descreve o algoritmo de coloração usando um número mínimo de cores.

A. Algoritmo de Tarjan para encontrar as componentes fortemente conexas

Este algoritmo de Tarjan é um algoritmo de Teoria dos Grafos para encontrar as componentes fortemente conexas de um grafo direcionado [18]. Uma componente fortemente conexa de um grafo orientado $G = (V, E)$ é um conjunto máximo de vértices $C \subseteq V$ tal que, para todo par de vértices u e v em C, existe um caminho de u para v e vice-versa [20].

O algoritmo tem como entrada um grafo direcionado $G = (V, E)$ e produz uma partição dos vértices do grafo que são as componentes. Todo vértice do grafo aparece em apenas uma delas. A ideia básica do algoritmo é uma busca em profundidade que começa em um vértice qualquer e continua nos outros vértices ainda não procurados, até explorar todos eles.

B. Coloração em grafos usando um número mínimo de cores

Coloração de grafos é um caso especial de rotulagem de grafos, sendo um processo de assimilar rótulos chamados de "cores" a elementos do grafo [19]. Dado um grafo $G = (V, E)$, no qual V é o conjunto dos vértices e E o conjunto das arestas, uma coloração dos vértices de G é uma atribuição de cores para os vértices de G de tal forma que vértices adjacentes não possuam a mesma cor [21]. Assim, uma coloração é um mapeamento $f : V \rightarrow \{1, 2, \ldots\}$, em que para quaisquer $(v_i, v_j) \in E, f(v_i) \neq f(v_j)$ [21].

O algoritmo realiza uma coloração usando um número mínimo de cores. Primeiramente, todos os vértices do grafo recebem uma mesma cor. Para cada aresta do grafo, é verificado se os vértices origem e destino possuem a mesma cor. Caso possuam, o vértice destino recebe a cor de maior índice já designada a outro vértice. Se esse vértice já possui a cor de maior índice, ele recebe a cor de maior índice mais um. Os autores não provam que o algoritmo colore com um número mínimo de cores. O resultado pode ser uma coloração viável, mas não necessariamente mínima.

III. Aproximando a Quantidade de Nós Maliciosos

O objetivo do algoritmo proposto é aproximar a quantidade de nós maliciosos que gera uma dada configuração na rede, baseado na visão local de cada nó. As informações locais de confiança reportadas pelos nós seguem a regra de invalidação da Tabela I. Um nó correto sempre reporta corretamente o estado dos seus vizinhos, enquanto que um nó malicioso pode ou não reportar corretamente sua visão local dos seus vizinhos. Esta invalidação é similar a usada em diagnóstico a nível de sistema [22]. Em outras palavras, se um nó correto a confia em um dos seus vizinhos, nó b, o nó a reporta o valor 1 para o nó b. Se o nó a não confia no nó b, ele reporta 0. Nós maliciosos podem reportar qualquer valor em relação aos seus vizinhos, assim a informação reportada por eles não é confiável.

TABELA I
Regra de invalidação.

Nó	Vizinhos	Informação Local
Correto	Correto	Correto (1)
Correto	Malicioso	Malicioso (0)
Malicioso	Correto	Indeterminado (0 ou 1)
Malicioso	Malicioso	Indeterminado (0 ou 1)

A entrada do algoritmo é um grafo $G = (V, E)$ com as visões locais de todos os nós, maliciosos ou não. Neste

grafo, nós são representados pelos vértices (conjunto V) e sua conexão local é representada pelas arestas (conjunto E). Todas as arestas são direcionadas, uma aresta $a \rightarrow b$ representa a conexão entre o nó a com o nó b. Sem perder a generalidade, a heurística é restrita a grafos bidirecionais. Assim, se a é vizinho de b, então b também é vizinho de a, portanto ambas as arestas, $a \rightarrow b$ e $b \rightarrow a$ existem. Cada aresta possui um valor associado, um peso, representando a visão local de um nó sobre seus vizinhos. Se o nó a acredita que o nó b é correto, o peso da aresta $a \rightarrow b$ é igual a 1, caso contrário é igual a 0. Note que as arestas $a \rightarrow b$ e $b \rightarrow a$ podem possuir valores diferentes.

Definir a visão local dos nós não faz parte do escopo deste artigo. Pode ser feito através de algoritmos de diagnóstico a nível de sistema [23], [24], através de sistemas de estabelecimento de confiança a um pulo de distância [13], [10], [25], [26] ou através de qualquer outro esquema capaz de fornecer as visões locais. A escolha de qualquer um desses algoritmos não afeta o algoritmo proposto ou os resultados apresentados.

Após receber como entrada o grafo G com as visões locais, o algoritmo de Tarjan para encontrar as componentes fortemente conexas é executado. Cada componente é formada por nós conectados por arestas $a \overset{1}{\underset{1}{\leftarrow\rightarrow}} b$. Um grafo $T = (V', E')$ com as componentes é criado. Então, a heurística de coloração é executada sobre o grafo $T = (V', E')$.

Note que o algoritmo de Tarjan não classifica dois nós vizinhos em T como corretos. A existência de dois nós vizinhos em T classificados como correto é uma contradição. Se existem nós vizinhos corretos, o algoritmo de Tarjan os deixa na mesma componente fortemente conexa, um mesmo nó em T.

A coloração classifica os nós como maliciosos ou corretos da seguinte forma: a cor que possui mais nós no grafo original é classificada como correta no grafo T; as outras cores são classificadas como maliciosas. A Tabela II apresenta a complexidade das heurísticas utilizadas.

TABELA II
COMPLEXIDADE DE TODOS OS ALGORITMOS UTILIZADOS PELO ALGORITMO PROPOSTO.

Algoritmo	Complexidade
Algoritmo de Tarjan [18]	$O(\|V\| + \|E\|)$
Algoritmo de Coloração [19]	$O(\|E\|)$

Após classificar os nós em T como corretos ou maliciosos, o algoritmo retorna para o grafo original, grafo G e reproduz para o usuário todos os nós classificados como maliciosos. Neste momento, o usuário ou o administrador da rede pode analisá-los pessoalmente ou, até mesmo, removê-los da rede. O algoritmo pode ser reexecutado para verificar a existência de nós maliciosos adicionais ou não. Este processo pode ser repetido até que não haja mais nós maliciosos na rede. Neste caso, todos os nós permanecem na mesma componente fortemente conexa depois de executar o algoritmo de Tarjan.

IV. RESULTADOS

O algoritmo proposto foi avaliado através de simulações em quatro redes complexas reais. Elas são grafos de redes sociais disponibilizadas por *SNAP: Stanford Network Analysis Platform*.

A Rede 1 tem 77360 nós e 828161 arestas. É um site com notícias relacionadas à tecnologia. A rede contém ligações amigo/inimigo entre os usuários e foi obtida em novembro de 2008. A Rede 2 com 82168 nós e 870161 arestas é a mesma rede, porém foi obtida em fevereiro de 2009. A Rede 3 tem 75888 nós e 508837 arestas. É uma rede social *online* do tipo quem-confia-em-quem de um site de opinião geral dos consumidores. A Rede 4 possui 8298 nós e 103689 arestas. Ela contém todos os votos até janeiro de 2008 para entrar na Wikipédia como administrador.

Para simular o comportamento malicioso, um determinado número de nós foi escolhido para ser maliciosos, a quantidade de nós maliciosos varia de 0% a 50% da rede, pois redes com mais de 50% de nós maliciosos são redes completamente comprometidas. Estes nós foram selecionados antes da execução do algoritmo proposto. Foram simulados dois tipos de comportamento malicioso dos nós: (i) nós que invertem a visão local para todos os seus vizinhos e (ii) nós que escolhem aleatoriamente alguns nós vizinhos para inverter a visão local desses nós. O último é um cenário mais realístico, pois o comportamento de um nó malicioso não pode ser previsto. Os resultados apresentados são médias de 100 simulações.

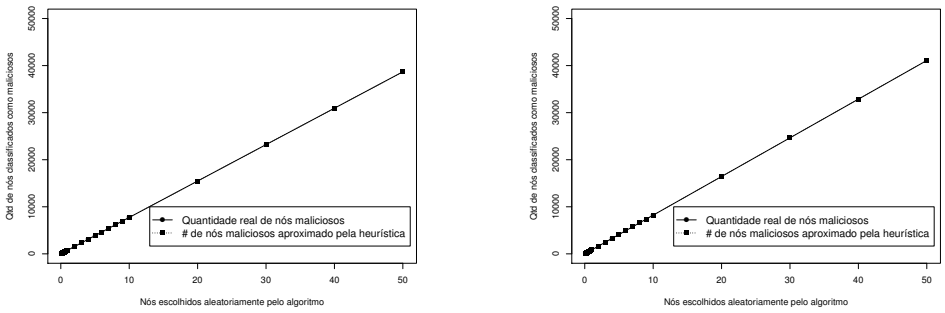
Nos gráficos a seguir, o eixo y - *Qtd de nós classificados como maliciosos* varia de $0-50000$ para todas as redes, exceto para a Rede 4. Para a Rede 4, o eixo y varia de $0 - 5000$. Essa diferença no tamanho do eixo y foi decidida para uma melhor visualização do leitor. Em todas os gráficos, o eixo x - *Nós escolhidos aleatoriamente pelo algoritmo* varia entre $0\% - 50\%$.

Para melhorar a legibilidade, todos os gráficos mostram a quantidade exata de nós maliciosos no sistema, a reta inteira. Note que esse valor não está disponível para o administrador da rede. Ele é usado apenas para demostrar a precisão do algoritmo proposto. A reta pontilhada nos gráficos representa o resultado do algoritmo. A comparação entre a reta inteira e a reta pontilhada demonstra a precisão do mesmo.

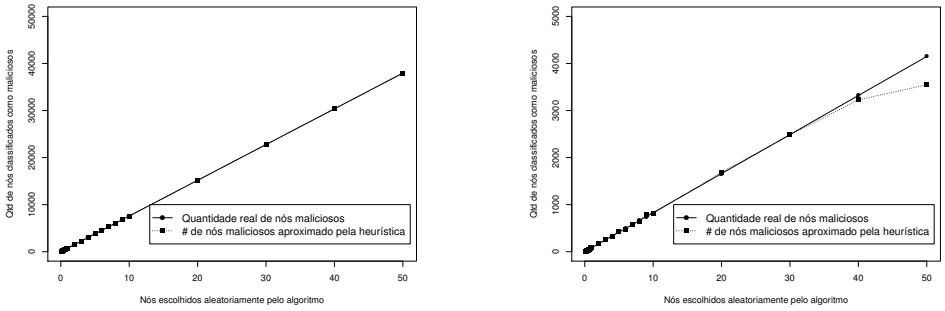
As Figuras 1 e 2 exibem os resultados retornados pelo algoritmo. Na primeira figura, os nós maliciosos invertem o peso de todas as arestas, enquanto que na segunda eles escolhem aleatoriamente.

A Tabela III apresenta a diferença entre a quantidade exata de nós maliciosos e a quantidade retornada pelo algoritmo proposto, i.e. o erro do algoritmo esta em números reais e não em porcentagens. Para as Redes 1, 2 e 3, o algoritmo possui o mesmo comportamento, i.e. ele é extremamente preciso para até 50% de nós maliciosos invertendo todos os pesos das arestas. Nestes casos, o erro é menor do que 1 nó. Para o caso em que os nós maliciosos invertem aleatoriamente o peso das arestas, o algoritmo é extremamente preciso para até 20% de nós maliciosos, tendo uma taxa de erro menor do que 7%. Considerando o pior caso, i.e. 50% de nós maliciosos, o algoritmo tem uma taxa de erro em torno de 15%.

A Rede 4 é a mais complexa de todas, pois possui a maior conectividade. Considerando o caso em que os nós maliciosos invertem o peso de todas as arestas para os seus vizinhos, o algoritmo apresenta um erro menor do que 3% para até

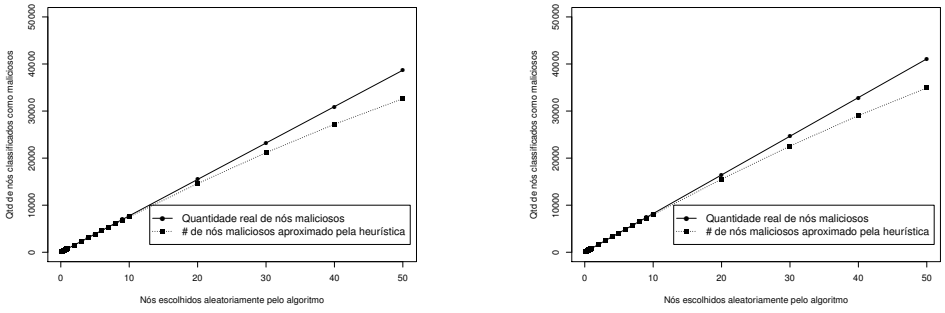


(a) Rede 1. (b) Rede 2.

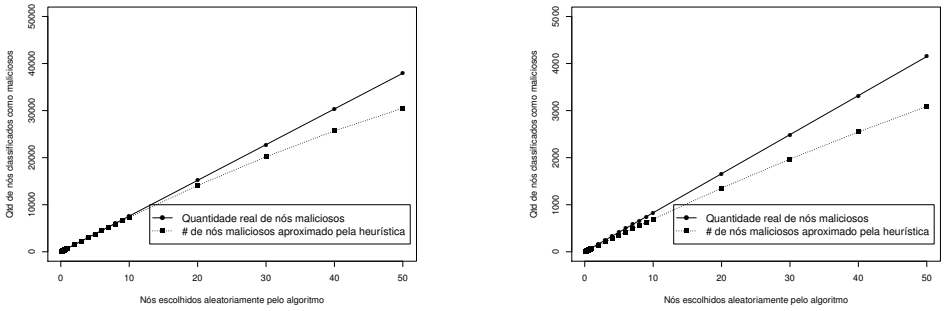


(c) Rede 3. (d) Rede 4.

Fig. 1. Nós maliciosos invertem o peso de todas as arestas para os seus vizinhos.



(a) Rede 1. (b) Rede 2.



(c) Rede 3. (d) Rede 4.

Fig. 2. Nós maliciosos invertem aleatoriamente o peso das arestas para os seus vizinhos.

40% de nós maliciosos, e em torno de 14% para 50% de nós maliciosos. No caso da inversão aleatória dos pesos, a taxa de erro do algoritmo fica entre 10% e 25%. A taxa de erro é diretamente relacionada com a quantidade de nós maliciosos no sistema. Apesar dos resultados para a Rede 4 serem menos precisos, ainda é uma aproximação muito boa para todas as quantidades de nós maliciosos. Recordando que o administrador do sistema pode usar esses resultados para reparar ou remover as unidades apontadas como maliciosas e, então, reexecutar o algoritmo para uma nova checagem.

Também é importante mencionar que o número de falsos positivos, os valores negativos na Tabela III, são muito pequenos. Para o comportamento aleatório dos nós maliciosos, eles são zero em todos os casos. Para o caso em que os nós maliciosos invertem o peso de todas as arestas, o número de falsos positivos é relevante apenas em alguns casos da Rede

TABELA III

MARGEM DE ERRO PARA O ALGORITMO DE COLORAÇÃO.

% 1's e 0's	Algoritmo de Coloração							
	Inverte				Aleatório			
	R1	R2	R3	R4	R1	R2	R3	R4
99,9-0,1	0,360	0,168	0,888	1,418	0,380	0,208	0,928	1,388
99,8-0,2	0,720	0,336	0,826	2,846	0,820	0,406	0,886	2,746
99,7-0,3	0,080	0,504	0,694	3,974	0,270	0,674	0,914	4,214
99,6-0,4	0,440	0,672	0,442	4,842	0,880	1,062	1,052	4,672
99,5-0,5	0,800	0,840	0,470	5,740	1,340	1,450	1,060	6,770
99,4-0,6	0,160	0,008	0,298	-4,472	1,190	0,778	1,378	7,738
99,3-0,7	0,520	0,176	0,326	8,206	1,630	1,336	1,566	8,416
99,2-0,8	0,880	0,344	0,074	-14,966	2,430	1,684	1,874	9,574
99,1-0,9	0,240	0,512	1,062	11,472	1,940	2,382	3,272	11,772
99-1	0,600	0,680	1,020	0,480	2,670	2,760	3,580	12,650
98-2	0,200	0,360	0,780	0,110	8,750	9,270	11,600	24,300
97-3	0,800	0,040	0,630	-1,050	18,800	18,940	25,130	38,130
96-4	0,400	0,720	0,490	-1,050	35,120	35,440	44,650	50,480
95-5	0,000	0,400	-0,150	-13,530	52,370	54,330	67,970	65,940
94-6	0,600	0,080	0,520	35,510	74,850	79,190	98,680	78,100
93-7	0,200	0,760	-0,400	-2,130	104,790	107,220	131,940	92,070
92-8	0,800	0,440	0,000	21,220	136,920	141,790	172,280	106,990
91-9	0,400	0,120	0,730	-50,600	178,010	178,770	220,550	120,650
90-10	0,000	0,800	1,220	19,890	215,120	218,560	273,480	135,860
80-20	0,000	0,600	0,520	-28,220	880,790	898,630	1102,120	305,870
70-30	0,000	0,400	-0,080	8,280	2049,370	2089,780	2579,600	519,130
60-40	0,000	0,200	0,680	90,900	3749,620	3824,520	4659,440	772,810
50-50	0,000	0,000	4,910	603,590	6064,030	6179,500	7430,150	1062,510

4. Para as redes 1 e 2 o número de falsos positivos é zero, enquanto para a rede 3 o máximo de falsos positivos é uma aproximação com 0.4 de erro na quantidade de nós maliciosos.

V. CONCLUSÕES E TRABALHOS FUTUROS

Nós maliciosos ou egoístas podem prejudicar a operação correta de uma rede complexa. O desenvolvimento de um algoritmo que estime a quantidade desses nós na rede é crucial. O uso de algoritmos de estabelecimento de confiança tenta prevenir esses comportamentos. Nós que evitam cooperar ou apresentam comportamento enganoso possuem valores de confiança baixos e podem ser penalizados, pois os outros nós tendem a cooperar somente com nós com altos valores de confiança.

Este artigo apresentou um algoritmo para calcular o número de nós maliciosos e egoístas em uma rede, baseado nas visões locais de confiança que cada nó tem em relação aos seus vizinhos. O algoritmo indica para o administrador da rede exatamente quais são esses nós.

O algoritmo proposto foi avaliado através de simulações em quatro redes complexas reais. Os resultados demonstram que o algoritmo é extremamente preciso. Em três das redes escolhidas, ele apresenta uma margem de erro menor que 15% quando o número de nós maliciosos no sistema é menor que 50% da rede inteira, i.e. cerca de 35000 nós maliciosos. Se o número de nós maliciosos está abaixo de 20% (aproximadamente 15000 nós maliciosos), a margem de erro é abaixo de 7%. Se o número de nós maliciosos fica abaixo de 5000, a margem de erro é cerca de um nó.

Trabalhos futuros incluem a aplicação do algoritmo proposto em outras redes complexas, como as MANETs. Também inclui o estudo da utilização do algoritmo de identificação de falhas para sistemas distribuídos.

REFERÊNCIAS

[1] Wang, X. F. and Chen, G. (2003) Complex networks: small-world, scale-free and beyond. *Circuits and Systems Magazine, IEEE*, **3**, 6–20.

[2] Newman, M. E. J. (2003) The structure and function of complex networks. *SIAM Review*, **45**, 167–256.

[3] Lü, L. and Zhou, T. (2011) Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, **390**, 1150–1170.

[4] Motter, A. E., Zhou, C. S., and Kurths, J. (2005) Enhancing complex-network synchronization. *EPL (Europhysics Letters)*, **69**, 334.

[5] Baras, J. S. and Jiang, T. (2005) Cooperation, trust and games in wireless networks. *Advances in Control, Communication Networks, and Transportation Systems*, pp. 183–202. Birkhäuser Boston.

[6] K. Seshadri, A. A. C. and Kasiviswanth, N. (2010) A survey on trust management for mobile ad hoc networks. *IJNSA International Journal of Network Security & Its Applications*, **2**, 75–85.

[7] Ali, B., Villegas, W., and Maheswaran, M. (2007) A trust based approach for protecting user data in social networks. *Proceedings of the 2007 conference of the center for advanced studies on Collaborative research*, pp. 288–293.

[8] Walter, F., Battiston, S., and Schweitzer, F. (2008) A model of a trust-based recommendation system on a social network. *Autonomous Agents and Multi-Agent Systems*, **16**, 57–74.

[9] Golbeck, J. A. and Hendler, J. (2006) Inferring binary trust relationships in web-based social networks. *ACM Trans. Internet Technol.*, **6**, 497–529.

[10] Kamvar, S. D., Schlosser, M. T., and Garcia-Molina, H. (2003) The eigentrust algorithm for reputation management in p2p networks. *Proceedings of the 12th international conference on World Wide Web*, pp. 640–651.

[11] Aberer, K. and Despotovic, Z. (2001) Managing trust in a peer-2-peer information system. *Proceedings of the tenth international conference on Information and knowledge management*, pp. 310–317.

[12] Cho, J.-H., Swami, A., and Chen, I.-R. (2011) A survey on trust management for mobile ad hoc networks. *Communications Surveys Tutorials, IEEE*, **13**, 562–583.

[13] Jiang, T. and Baras, J. S. (2005) Autonomous trust establishment 1. *Proceedings of 2nd International Network Optimization Conference*.

[14] Theodorakopoulos, G. and Baras, J. S. (2006) On trust models and trust evaluation metrics for ad hoc networks. *Selected Areas in Communications*, **24**, 318–328.

[15] Theodorakopoulos, G. and Baras, J. S. (2006) Enhancing benign user cooperation in the presence of malicious adversaries in ad hoc networks. *Securecomm and Workshops, 2006*, pp. 1–6.

[16] Jiang, T. and Baras, J. S. (2007) Trust document distribution in manets. *Military Communications Conference, 2007. MILCOM 2007. IEEE*, pp. 1–7.

[17] Somasundaram, K. K. and Baras, J. S. (2011) Path optimization and trusted routing in manet: An interplay between ordered semirings. *Advances in Networks and Communications*, pp. 88–98. Springer Berlin Heidelberg.

[18] Tarjan, R. (1972) Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, **1**, 146–160.

[19] Mittal, A., Jain, P., Mathur, S., and Bhatt, P. (2011) Graph coloring with minimum colors: An easy approach. *2011 International Conference on Communication Systems and Network Technologies (CSNT)*, pp. 638–641.

[20] Cormen, T. H., Stein, C., Rivest, R. L., and Leiserson, C. E. (2001) Introduction to Algorithms. The MIT Press.

[21] Pardalos, P. M., Mavridou, T., and Xue, J. (1998) The graph coloring problem: A bibliographic survey. *Handbook of Combinatorial Optimization*, pp. 331–395.

[22] Preparata, F. P., Metze, G., and Chien, R. T. (1967) On the connection assignment problem of diagnosable systems. *Electronic Computers, IEEE Transactions on*, **EC-16**, 848–854.

[23] Hosseini, S., Kuhl, J., and Reddy, S. (1984) A diagnosis algorithm for distributed computing systems with dynamic failure and repair. *Computers, IEEE Transactions on*, **C-33**, 223–233.

[24] Duarte, E. P., Jr., Ziwich, R. P., and Albini, L. C. (2011) A survey of comparison-based system-level diagnosis. *ACM Comput. Surv.*, **43**, 22:1–22:56.

[25] Buchegger, S. and Le Boudec, J.-Y. (2002) Nodes bearing grudges: towards routing security, fairness, and robustness in mobile ad hoc networks. *Parallel, Distributed and Network-based Processing, 2002. Proceedings. 10th Euromicro Workshop on*, pp. 403–410.

[26] Buchegger, S. and Le Boudec, J.-Y. (2002) Performance analysis of the confidant protocol. *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pp. 226–236.