A Quantum Algorithm for Assessing Node Importance in the *st*-Connectivity Attack

Iain Burge[1], Michel Barbeau[2], and Joaquin Garcia-Alfaro[1](✉)

[1] SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, 91120 Palaiseau, France
joaquin.garcia_alfaro@telecom-sudparis.eu

[2] School of Computer Science, Carleton University, Ottawa, Canada

Abstract. Problems in distributed security often naturally map to graphs. The centrality of nodes assesses the importance of nodes in a graph. It is used in various applications. Cooperative game theory has been used to create nuanced and flexible notions of node centrality. However, the approach is often computationally complex to implement classically. This work describes a quantum approach to approximating the importance of nodes that maintain a target connection. In addition, we detail a method for quickly identifying high-importance nodes. The approximation method relies on quantum subroutines for st-connectivity and approximating Shapley values. The search for important nodes relies on a quantum algorithm to find the maximum. We consider st-connectivity attack scenarios in which a. malicious actor disrupts a subset of nodes to perturb the system functionality. Our methods identify the nodes that are most important in minimizing the impact of the attack. The node centrality metric identifies where more redundancy is required and can be used to enhance network resiliency. Finally, we explore the potential complexity benefits of our quantum approach in contrast to classical random sampling.

Keywords: Quantum computing · Distributed system · Graph analytics · Game theoretic node centrality · st-Connectivity

1 Introduction

Quantum science and related technologies hold significant potential for global innovation in several domains, quantum-enhanced information networks being only one of them. With recent promising results in quantum computing for combinatorial optimization problems, quantum-enhanced information networks are a promising evolution of classical distributed systems where the use of quantum technologies is expected to foster significant new paradigms [14]. This includes the development of quantum sensor networks and the enhancement of Key Distribution (QKD) technologies [9]. The integration of quantum computing within these new environments must address traditional security problems, including defense and resilience.

In the realm of graph analytics, node centrality metrics quantify properties such as the utility of a node, whether a node is critical in keeping the graph connected, or if the node is vulnerable to attack. These metrics help to determine whether a network is secure and resilient. They can guide structural changes to improve these properties. Traditional node centrality metrics look at individual nodes; however, some properties cannot be easily measured without considering the coalitions of nodes.

This paper builds upon the flexible notion of game-theoretic node centrality measures. Specifically, we describe a node centrality metric based on the connection of two critical nodes. We use the metric to handle the following two properties (relevant to distributed systems' security): resilience and remediation degree. The former refers to the ability of a communication network to maintain functionality and carry out its mission, even in the face of adversarial events. An adversarial event can either occur naturally or result from deliberate actions. The latter can be used to quantify the capacity to provide restoration and mitigation capabilities after an attack on the system occurs.

We aim at addressing the aforementioned properties in the presence of adversaries in a distributed system perpetrating a given type of attack (the *st*-connectivity attack). We build a methodological solution to assess the node's importance. Quantifying the importance of nodes can be used to guide modifications to the network topology so that the level of resilience is improved. We also explore the advantage of a quantum version of our solution, compared to a baseline classical computing solution. We present a practical implementation of our approach and provide a rough estimation and comparison of the complexity of our solution w.r.t. classical Monte Carlo methods. The code is available in our companion Github repository[1].

Paper Organization—Section 2 provides motivation and preliminaries. Section 3 presents our contribution and its complexity analysis. Section 4 surveys related work. Section 5 provides the conclusion and perspectives for future work.

2 Motivation

We assume a quantum distributed system that offers, for example, quantum key expansion, entanglement swapping, and error mitigation services [9,14], in which an adversary aims to disrupt the connectivity of the service from node s to node t. By assuming a classical abstraction of the problem and focusing only on the information-gathering stage of the attack, we aim to anticipate adversarial strategies to disconnect. s from t (i.e., we assume that the adversary can successfully sabotage the services in those intermediate nodes from s to t, hence avoiding any possible functionality between both nodes). Our goal is to identify the most important nodes in keeping node t accessing the services of s, allowing

[1] https://github.com/iain-burge/quantum_st-attack.

us to increase the resilience of the network. We accomplish this by using the game-theoretic concept of centrality of nodes as a metric to quantify the degree of remediation associated with the attack scenario.

Before moving forward, we provide some needed preliminaries and definitions on the use of cooperative games on graphs. We start with some background concepts on which our approach is based.

Definition 1 (Network Graph). *Define a graph* $H = (N, E)$ *as a pair of the set of network nodes* N *and the set of edges* $(u, w) \in E$*, with* $u, w \in N$ *and* $u \neq w$*.*

Remark 1 (Graph Representation). Let us index each node by some integer in $\mathbb{Z}_{|N|}$. Each edge is indexed by an integer in $\mathbb{Z}_{\binom{|N|}{2}}$ that is mapped to the set of pairs $\{(a, b) : a, b \in \mathbb{Z}_{|N|}, a \neq b\}$, with a bijection. We write the edge index (u, w) as uw. We may represent the adjacency matrix of the graph with a binary string $x \in \{0, 1\}^{\binom{|N|}{2}}$, where x_{uw} is one if $(u, w) \in E$, otherwise x_{uw} is zero.

Definition 2 (Cooperative Games on a Network Graph [18]). *We define a cooperative game on graph* $H = (N, E)$ *to be the pair* $G_H = (F, V)$*, where* $F \subseteq N$ *and* V *is a valuation function from the subsets of* F *to the reals, i.e.,* $V : \mathcal{P}(F) \to \mathbb{R}$*. With the restriction that* $V(\emptyset) = 0$*.*

This definition allows us to treat the nodes in F as players in a game. Given a subset of nodes $R \subseteq F$, we can treat it as a binary graph coloring where the colors correspond to the inclusion (or exclusion) of the node in R. $V(R)$ represents the value of that particular graph coloring.

Though it is useful to have a value for coalitions of nodes, or their colorings, the number of combinations grows exponentially with respect to graph size. Thus, it is useful to have a metric that can condense this vast amount of information into a utility for each node. We adapt the Shapley value solution concept to our current situation.

Definition 3 (Node Shapley Value [16]). *Given a game* $G_H = (F, V)$ *on graph* $H = (N, E)$*, with* $F \subseteq N$*. The ith node's Shapley value* Φ_i *is,*

$$\sum_{R \subseteq F \setminus \{i\}} \gamma\left(|F \setminus \{i\}|, |R|\right) \cdot \left(V(R \cup \{i\}) - V(R)\right)$$

where $\gamma(n, m) = \left(\binom{n}{m}(n+1)\right)^{-1}$*.*

In this work, we proceed with the narrow concept of graph coloring. If node $a \in F$ is in the subset $Q \subseteq F$, it is considered *enabled*, otherwise, if a is not in Q, a is considered *disabled*.

Definition 4 (subgraph H_Q). *We define the subgraph* $H_Q = (Q, E_Q)$ *of the graph* $H = (N, E)$*, such that* $Q \subseteq N$*.* $E_Q \subseteq E$ *is the subset of all edges* $(a, b) \in E$ *where* $a, b \in Q$*.*

In the context of node centrality, we consider the value function $V(Q)$ that indicates whether H_Q maintains a particular property.

2.1 The *st*-Connectivity Attack

Definition 5 (*st*-connectivity). *Consider a graph $H = (N, E)$, with nodes $s, t \in N$. The graph H is st-connected if there exists a path from node s to node t. Formally, H is st-connected if there exists a sequence of nodes $s = u_0, u_1, u_2, \ldots, u_{r-1}, u_m = t$ such that $(u_k, u_{k+1}) \in E$ for $k \in \{0, \ldots, m-1\}$. We define the value function $V_{st} : \mathcal{P}(F) \rightarrow \mathbb{R}$,*

$$V_{st}(R) = \begin{cases} 1 & \textit{if } H_{R\cup\{s,t\}} \textit{ is st-connected,} \\ 0 & \textit{otherwise,} \end{cases}$$

where $R \subseteq F = N \setminus \{s, t\}$, and H_R is described in Definition 4.

In the context of our scenario, the adversary aims to remove *st*-connectivity (source-target-connectivity). The value function returns 0 when the set of enabled nodes H_R is no longer able to keep the target connected to the source and 1 when it maintains that property. Hence, the Shapley values (Definition 3) of each node reflect how critical it is to maintain that connection. A high Shapley value means that the node is a valuable target, while a low Shapley value means that the node is not of interest.

Definition 6 (*st*-connectivity attack). *Given a graph $H = (N, E)$, an st-connectivity attack is a malicious action perpetrated by an adversary. The adversary can turn off a subset of nodes $Q \subseteq F = N \setminus \{s, t\}$. The adversary's goal is to transform the graph H into a subgraph $H_{N\setminus Q}$ that is not st-connected. Equivalently, the adversary's goal is to minimize $V_{st}(F \setminus Q)$.*

3 Quantum Approach

In this section, we present our quantum algorithm for *st*-connectivity assessment. To begin, we define a simplified version of span programs, detailed in [4,7].

Definition 7 (Span Program Decision Problem). *A* span program *$P(|\tau\rangle, \mathcal{W}, x)$ takes as input a unit* target vector *$|\tau\rangle \in \mathbb{C}^d$, a set of* input vectors *$\mathcal{W} = \{|\mu_{k,0}\rangle : k \in \mathbb{Z}_r\} \cup \{|\mu_{k,1}\rangle : k \in \mathbb{Z}_r\} \subset \mathbb{C}^d$, and a binary vector selection string $x = x_{r-1} \cdots x_0 \in \{0, 1\}^r$. Given x, the* available vectors *are $A = \{|\mu_{k,x_k}\rangle : k \in \mathbb{Z}_r\}$. The span program P outputs 1 if the target $|\tau\rangle$ is in the span of the available vectors $\mathtt{Span}(A)$. Equivalently, P outputs 1 if there exists a complex vector $c \in \mathbb{C}^r$ such that,*

$$|\tau\rangle = \sum_{k=0}^{r-1} c_k |\mu_{k,x_k}\rangle .$$

Otherwise, the program returns 0.

We now reformulate the problem of *st*-connectivity as a span program decision problem [4].

Theorem 1 (Span Program for *st*-Connectivity). *Consider graph $H = (N, E)$. We detail a span program that determines, given $s, t \in N$, if H is st-connected. Let $|v\rangle$ be a basis vector that represents a node $v \in N$. Define $P(|\tau\rangle, \mathcal{W}, x)$, where $|\tau\rangle \in \mathbb{C}^{|N|}$,*

$$\mathcal{W} = \left\{|\mu_{uw,0}\rangle : uw \in \mathbb{Z}_{\binom{|N|}{2}}\right\} \cup \left\{|\mu_{uw,1}\rangle : uw \in \mathbb{Z}_{\binom{|N|}{2}}\right\} \subset \mathbb{C}^{|N|},$$

and x is the binary string representation of the adjacency matrix for H (Remark 1). So, x_{uw} is 1 if $(u, w) \in E$, otherwise, x_{uw} is 0. The target vector is,

$$|\tau\rangle = \frac{|t\rangle - |s\rangle}{\sqrt{2}}, \quad s, t \in N.$$

The input vectors are $|\mu_{uw,0}\rangle = 0$, and, $|\mu_{uw,1}\rangle = (|u\rangle - |w\rangle)/\sqrt{2}$, for all $u, w \in N$, and edge indices $uw \in \mathbb{Z}_{\binom{|N|}{2}}$. Thus, our available vector span is,

$$\texttt{Span}(A) = \texttt{Span}\left\{\frac{|u\rangle - |w\rangle}{\sqrt{2}} : x_{uw} = 1, uw \in \mathbb{Z}_{\binom{|N|}{2}}\right\}.$$

If the span program outputs 1, H is st-connected, otherwise, H is not st-connected.

Proof. Suppose $H = (N, E)$ is st-connected, then there exists a sequence of nodes $s = u_0, \ldots, u_{r-1}, u_m = t$, such that $(u_k, u_{k+1}) \in E$, $k \in \{0, \ldots, m-1\}$. As a result, for our span program $P(|\tau\rangle, \mathcal{W}, x)$, the set of available vectors A includes every

$$\frac{|u_{k+1}\rangle - |u_k\rangle}{\sqrt{2}}, with \quad k \in \mathbb{Z}_r.$$

Simultaneously, we have,

$$|\tau\rangle = \sum_{k=0}^{r-1} \frac{|u_{k+1}\rangle - |u_k\rangle}{\sqrt{2}},$$

since the right-hand side is a telescoping sequence. As a result, the span program accepts the input as expected. In [7] a proof shows that the span program rejects H when it is not st-connected. □

Theorem 2 (Quantum *st*-Connectivity Algorithm [4,7]). *There exists a quantum algorithm to decide whether a graph $H = (N, E)$, with nodes $s, t \in N$, is st-connected. The algorithm uses $\mathcal{O}(\log|N|)$ space and takes $\tilde{\mathcal{O}}\left(|N|^{\frac{3}{2}}\right)$ queries to the adjacency matrix up to polylogarithmic factors (where $\tilde{\mathcal{O}}$ ignores slow-growing factors). The routine is successful with a probability of at least 9/10. The best possible classical algorithm takes at least $\Omega\left(|N|^2\right)$ time.*

Formally, we have a unitary quantum transformation U_{st} which acts on an auxiliary register of $\mathcal{O}(\log|N|)$ qubits **aux** *and an output register of one qubit* **out**. *Performing the algorithm and tracing out the auxiliary register results in,*

$$\texttt{tr}_{aux}\left(U_{st}\,|0\rangle_{aux}^{\otimes\mathcal{O}(\log|N|)}\,|0\rangle_{out}\right) = ((1-p)\,|\neg y\rangle\langle\neg y| + p\,|y\rangle\langle y|)_{out}$$

where y is one if H is st-connected and zero otherwise, and p is in range $[9/10, 1)$*. Measurement of the output bit returns the correct output with probability p.*

Proof. We proceed with a rough sketch of the algorithm. A full algorithm and proof are provided in [7]. The algorithm is based on the span program for *st*-connectivity. We perform phase estimation on the unitary matrix $U = (2\Lambda - I)(2\Pi_x - I)$ with the input vector $|0\rangle$ using precision $\mathcal{O}(|N|^{-3/2})$. Thus U is queried $\mathcal{O}(|N|^{3/2})$ times. If the phase estimation outputs zero, the algorithm claims that the graph H is *st*-connected and outputs 1. Otherwise, if the phase estimation outputs a non-zero answer, the algorithm claims that H is not *st*-connected, and outputs 0. It is correct with probability 9/10. We assume, for the sake of simplicity, that $(s,t) \notin E$, this can be checked in $\mathcal{O}(|N|)$ time. We also give edge (s,t) the index $st = 0$.

U is the product of two reflections, a reflection about Λ, and a reflection about Π_x. Λ represents a projection onto the kernel of,

$$\tilde{M} = \mathcal{O}\left(\frac{1}{\sqrt{|N|}}\right) |\tau\rangle \langle 0| + \sum_{uw \in \mathbb{Z}_{\binom{|N|}{2}} \setminus \{0\}} |\mu_{uw,1}\rangle \langle uw| .$$

$\tilde{M}$ represents a transformation from the indices of edges to their respective vectors in the span program for *st*-connectivity. The reflection, $(2\Lambda - I)$, is implemented using a Szegedy-type quantum walk [7,17]. The walk is implemented in logarithmic space and time with respect to $|N|$, and is input-independent. Π_x is the projection onto available vector indices and onto the target vector index,

$$\Pi_x = |0\rangle \langle 0| + \sum_{(u,w) \in E} |uw\rangle \langle uw| . \tag{1}$$

Thus, $(2\Pi_x - I)$ represents a reflection where all the indices of unavailable edges are negated. This reflection can be performed with a single query to the adjacency matrix.

Intuitively, the quantum phase estimation extracts the spectral qualities of U. The reflections $(2\Lambda - I)$ and $(2\Pi_x - I)$ are constructed such that the spectral qualities of U correspond to whether $|\tau\rangle$ is linearly independent of the available vectors. □

Remark 2 (Span Program for st-Connectivity Node Centrality). Consider the graph $H = (N, E)$. Suppose that we wish to determine the *st*-connectivity of a subgraph $H_R = (R, E_R)$, $R \subseteq N$. Equivalently, we wish to compute $V(R)$. We proceed similarly as in Theorem 1. Define the span program $P(|\tau\rangle, \mathcal{W}, x^R)$, where $|\tau\rangle$ and $\mathcal{W}$ are described in Theorem 1. Let x^R_{uw} be one if $uw \in E_R$, otherwise x^R_{uw} is zero. Equivalently, we can define x^R_{uw} to equal one if and only if x_{uw} is one and nodes $u, w \in R$.

Definition 8 (Majority Vote). *We define the majority function* $\mathtt{MAJ} : \{0,1\}^n \to \{0,1\}$, *where* n *is odd, as,*

$$\mathtt{MAJ}(z) = \begin{cases} 1 & \text{if } \sum_{k=0}^{n} z_k > n/2, \\ 0 & \text{otherwise.} \end{cases}$$

Where $z = z_{n-1} \cdots z_0 \in \{0,1\}^n$. *We also define the quantum version of this function,* $U_{\mathtt{MAJ}}$, *which operates on an* n*-qubit register* $\mathtt{in}$ *and a one-qubit register* $\mathtt{maj}$,

$$U_{\mathtt{MAJ}} |z\rangle_{\mathtt{in}} |0\rangle_{\mathtt{maj}} = |z\rangle_{\mathtt{in}} |\mathtt{MAJ}(z)\rangle_{\mathtt{maj}}.$$

Lemma 1 (Majority Vote Powering). *Suppose that we have a quantum algorithm* U *that outputs a binary value with fixed success probability* $p > 0.5$. *Let the correct value be* $y \in \{0,1\}$. *We can augment the probability of success by repeatedly performing the algorithm and taking the majority output. In particular, suppose our repeated quantum subroutine gave an* n*-qubit output of,*

$$\left((1-p) |\neg y\rangle \langle \neg y| + p |y\rangle \langle y|\right)^{\otimes n}.$$

Then, adding an extra qubit in the form of a $\mathtt{maj}$ *register, the majority vote unitary* $U_{\mathtt{MAJ}}$ *can be applied. Given a desired final failure probability bound* κ, *the* $\mathtt{maj}$ *register stores the correct answer with probability* $1 - \kappa$ *if* n *is of order* $\mathcal{O}\left(\log \kappa^{-1}\right)$. *In other words, we have failure chance* κ *given* $\mathcal{O}(\log \kappa^{-1})$ *applications of the* U *algorithm.*

Proof. Suppose we perform our quantum algorithm n times, where $n \geq 3$ is odd. This outputs a list of n bits. The probability that k bits are correct is,

$$\binom{n}{k} p^k (1-p)^{n-k}. \tag{2}$$

The threshold for a majority is $t = (n-1)/2$. So, the probability that the majority fails is $\sum_{k=0}^{t} \binom{n}{k} p^k (1-p)^{n-k}$. In Eq. (2), for $k \in \{0, 1, \ldots, t\}$, the probability is increasing with respect to k. Thus, the probability of majority failure is bounded by,

$$t \binom{n}{t} p^t (1-p)^{n-t}. \tag{3}$$

By an improved version of Stirling's formula [15],

$$\binom{n}{t} < \sqrt{\frac{n}{2\pi t(n-t)}} \frac{n^n}{t^t (n-t)^{n-t}} < \sqrt{\frac{2}{\pi n}} 2^n,$$

where the latter inequality is the result of replacing t with $n/2$. Plugging the inequality into Eq. (3) and once again replacing t with $n/2$ yields the new bound $\sqrt{\frac{n}{2\pi}} 2^n (p(1-p))^{n/2}$. So long as $\sqrt{p(1-p)} < 1/2$, which holds for $p > 0.5$, the upper bound for majority failure chance shrinks exponentially with respect to n. □

3.1 Quantum Algorithm for Shapley Value Approximation

The quantum algorithm for Shapley value approximation takes an approach inspired by classical random sampling [10]. Each subset of nodes is given a probability amplitude proportional to their γ coefficient in the Shapley equation (Definition 3). Classically, we would randomly sample from the distribution of node subsets, and record how much our target node increases the value of the subset. After many samples, we take the average increase in value and use it as an approximation. By Chebyshev's inequality, the number of samples required scales quadratically with respect to the desired error. The quantum approach can provide a quadratic improvement.

Theorem 3 (Quantum Algorithm for Shapley Value Approximation [5,6]). *Take the cooperative game on graph $H = (N, E)$ to be the pair $G_H = (F, V)$ where $F \subseteq N$ and V is the value function. Suppose we have a quantum implementation of V, U_V, and that we wish to find the Shapley value Φ_i of node i. Then, given a fixed desired probability for success, there exists a quantum algorithm that produces approximation $\tilde{\Phi}_i$ in,*

$$\mathcal{O}\left(\frac{\sqrt{(V_{\max} - V_{\min})(\Phi_i - V_{\min})}}{\epsilon}\right),$$

queries to the value function U_V. Where $V_{\max}, V_{\min}$ are respectively an upper and lower bound for the value function V, and the desired error bound is $\epsilon \geq |\Phi_i - \tilde{\Phi}_i|$.

Proof. We now give a sketch of the algorithm; a complete proof and error analysis is provided in [6]. We can uniquely encode a subgraph H_Q, $Q \subseteq F$, as a binary string of the form: $b^Q = b_0^Q b_1^Q \cdots b_{|F|-1}^Q \in \{0,1\}^{|F|}$, where $b_j^Q = 1$ if $j \in Q$ else $b_j^Q = 0$. We define quantum implementation U_V of V as,

$$U_V \left|b^Q\right\rangle_{\mathtt{Pl}} |0\rangle_{\mathtt{Ut}} = \left|b^Q\right\rangle_{\mathtt{Pl}} \left(\sqrt{1 - \frac{V(Q)}{V_{\max} - V_{\min}}}\,|0\rangle + \sqrt{\frac{V(Q)}{V_{\max} - V_{\min}}}\,|1\rangle\right)_{\mathtt{Ut}}.$$

We begin with a quantum state made of three registers: `Pt`, the partition register, which helps to prepare the γ probability amplitude distribution (Definition 3); `Pl`, the player register, which stores the subgraph encodings; and `Ut`, the utility register, which stores the value of a subgraph. We begin with the quantum state, $|0\rangle_{\mathtt{Pt}}^{\otimes \ell} |0\rangle_{\mathtt{Pl}}^{\otimes |F|} |0\rangle_{\mathtt{Ut}}^{\otimes 1}$, where $\ell = \mathcal{O}(\log((V_{\max} - V_{\min}) \cdot \sqrt{n}/\epsilon))$. Next, prepare the `Pt` register as follows,

$$\frac{1}{\sqrt{2^\ell}} \sum_{k=0}^{2^\ell - 1} |\theta_k\rangle_{\mathtt{Pt}} |0\rangle_{\mathtt{Pl}}^{\otimes |F|} |0\rangle_{\mathtt{Ut}}^{\otimes 1},$$

where θ_k is an ℓ bit binary approximaiton of $\arcsin\sqrt{2^{-\ell}k}$. For notational simplicity, we suppose $i = |F| - 1$. Using the partition register as a control, it is efficient to transform the state to,

$$\frac{1}{\sqrt{2^\ell}}\sum_{k=0}^{2^\ell-1}|\theta_k\rangle_{\mathtt{Pt}}\left(\left(\sqrt{1-2^{-\ell}k}\,|0\rangle+\sqrt{2^{-\ell}k}\,|1\rangle\right)^{\otimes|F|-1}\otimes|0\rangle\right)_{\mathtt{Pl}}|0\rangle_{\mathtt{Ut}}^{\otimes 1}, \tag{4}$$

Note that the bit corresponding to node i is zero. Switching to a density matrix representation and tracing out the partition register gives an approximation for the state,

$$\sum_{R\subseteq F\setminus\{i\}}\gamma\left(|F\setminus\{i\}|,|R|\right)\left|b^R\right\rangle_{\mathtt{Pl}}|0\rangle_{\mathtt{Ut}}\left\langle b^R\right|_{\mathtt{Pl}}\langle 0|_{\mathtt{Ut}}.$$

This results from the fact that $\int_0^1(1-t)^{n-m}t^m dt=\gamma(n,m)$ for integer $n\geq 2$, and $m\in\{0,1,\dots,m\}$. Now, applying U_V and measuring the utility bit gives an expected value of,

$$\frac{1}{V_{\max}-V_{\min}}\sum_{R\subseteq F\setminus\{i\}}\gamma\left(|F\setminus\{i\}|,|R|\right)V(R). \tag{5}$$

Using the quantum speedup for Monte Carlo methods [13], the expected value can be approximated quadratically faster than with classical methods.

We can repeat the process with a simple modification, prepare Eq. (4) where the bit corresponding to node i is one, then proceed identically to above. This yields the expected value,

$$\frac{1}{V_{\max}-V_{\min}}\sum_{R\subseteq F\setminus\{i\}}\gamma\left(|F\setminus\{i\}|,|R|\right)V(R\cup\{i\}). \tag{6}$$

Subtracting Eq. (5) from Eq. (6), then multiplying the result by $(V_{\max}-V_{\min})$ gives an approximation for the ith player's Shapley value. Note that we can compute Eq. (5), Eq. (6), and thus the entire Shapley approximation without measurement. As a result, we can approximately perform the transformation,

$$|i\rangle|0\rangle\rightarrow|i\rangle\left|\tilde{\Phi}_i\right\rangle. \tag{7}$$

□

Lemma 2 (Shapley Values and Unreliable Value Functions). *Consider the cooperative game $G_H=(F,V)$ on graph $H=(N,E)$ where $F\subseteq N$. We wish to find the Shapley value Φ_i of node i. Suppose $V:\mathcal{P}(F)\rightarrow\{0,1\}$ is a binary classifier, and that V is monotonic, i.e., if $Q,R\subseteq F$ then $V(Q\cup R)\geq V(Q)$. We define $\hat{V}$, that, given $Q\subseteq F$, fails and outputs $1-V(Q)$ with probability $\kappa\in[0,1]$, or succeeds and outputs $V(Q)$ with probability $1-\kappa$. Note, for simplicity, we assume a perfect implementation of the γ distribution, in reality, the implementation is an exponentially accurate approximation. Applying the Shapley value approximation using $\hat{V}$ as a substitute for V has an expected value*

$$\Phi_i+\xi$$

where ξ is bounded, $|\xi|\leq 2\kappa$.

Proof. We must find the expected value of the following equation,

$$\sum_{R \in F \setminus \{i\}} \gamma(|F \setminus \{i\}|, |R|) \left(\hat{V}(R \cup \{i\}) - \hat{V}(R) \right), \tag{8}$$

By definition, the expected value of $\hat{V}(Q)$, $Q \subseteq F$, is $\kappa \cdot (1 - V(Q)) + (1 - \kappa) \cdot V(Q)$. Rearranging gives, $\mathbb{E}\left[\hat{V}(Q)\right] = V(Q) + \kappa - 2\kappa V(Q)$. Thus, Eq. (8) has expected value,

$$\sum_{R \in F \setminus \{i\}} \gamma(|F \setminus \{i\}|, |R|) \left[(V(R \cup \{i\}) - V(R))(1 - 2\kappa) + 2\kappa \right].$$

Applying Definition 3 and Lemma 1 from [6], the expected value is equal to, $\Phi_i + 2\kappa(1 - \Phi_i)$. Since V is monotonic and the output is in the range $\{0, 1\}$, Φ_i is in the range $[0, 1]$. □

3.2 Combining the Algorithms

In this section, we describe a quantum approach for finding the *st*-connectivity-based node centrality. Consider the cooperative game $G_H = (F, V_{st})$ on graph $H = (N, E)$, where $s, t \in N$ and $F = N \setminus \{s, t\}$. Suppose that we wish to find the Shapley value Φ_i of node $i \in F$. We can represent each subset $Q \subseteq F$ with a binary string $b^Q = b_0^Q \cdots b_{|N|-1}^Q$ where b_j^Q is equal to 1 if $j \in Q$ else b_j^Q is 0. Note that $V_{st}(Q)$ is either 0 or 1. Hence, we can take $V_{\max} = 1$ and $V_{\min} = 0$.

Consider a modified quantum algorithm for the *st*-connectivity algorithm based on Remark 2. We define $U_{st}(Q)$, $Q \subseteq F$ as the quantum *st*-connectivity algorithm for graph $H_{Q \cup \{s,t\}}$. This requires a small alteration to the projection Π_x, Eq. (1). We replace Π_x with,

$$\Pi_x^Q = |0\rangle \langle 0| + \sum_{(u,w) \in E_Q} |uw\rangle \langle uw|.$$

This can often be done efficiently. Instead of directly using the adjacency bit x_{uw}, we use the binary value $x_{uw} \wedge b_u^Q \wedge b_w^Q$. Note that this implementation allows us to perform the calculation for all $Q \subseteq F$ in superposition. The modification makes the algorithm easily compatible with the Shapley value algorithm.

The base quantum algorithm for *st*-connectivity only has a success probability of 9/10 (Theorem 2). This is insufficient, as demonstrated by Lemma 2. However, it is possible to improve our accuracy with logarithmic factor increase to time and space complexity; we repeatedly perform the quantum *st*-connectivity algorithm and take the majority answer (Lemma 1). In particular, assuming a desired error κ, we can apply $U_{st}(Q)$ (Remark 2) $n \in \mathcal{O}(\log \kappa^{-1})$ times independently and take the majority vote. We begin with,

$$U_{st}(Q)^{\otimes n} \bigotimes_{k=0}^{n-1} |0\rangle_{\texttt{aux}_k}^{\otimes \mathcal{O}(\log |N|)} |0\rangle_{\texttt{out}_k}.$$

Tracing the auxiliary registers gives us a state of the form required in Lemma 1. Thus, we can take the majority vote $U_{\texttt{MAJ}}$ and output it to a new one-qubit register. If we consider this new register as our utility register `Ut` described in Theorem 3, we can apply the logic of Lemma 2. Specifically, for each possible subgraph $Q \subseteq F$ represented in the `Pl` register, the output, stored in the `Ut` register, holds the correct value $V(Q)$ with probability $1-\kappa$. As a result, we can define U_V as the product of repeatedly computing $U_{st}(Q)$ order $\mathcal{O}(\log \kappa^{-1})$ times, followed by a $U_{\texttt{MAJ}}$ operation on the output. Thus, by Lemma 2, the expected value that we extract, Φ_i, is shifted to $\Phi_i+\xi$, $\xi \leq 2\kappa$. Applying the Monte-Carlo quantum acceleration routine extracts the value $\Phi_i + \epsilon + \xi$. Since both ϵ and ξ can be bounded to arbitrarily small values, the algorithm is asymptotically correct.

3.3 Finding Important Nodes

Suppose that we wish to find the index of a node with a large Shapley value. Let node m have the largest Shapley value Φ_m. We find node j such that their Shapley value Φ_j is greater than or equal to $\Phi_m - \epsilon$.

Lemma 3. *Consider a game $G_H = (F, V)$, where F is a subset of nodes in the graph H, and $V : \mathcal{P}(F) \to \mathbb{R}$ is the value function. Suppose that the player m has the largest Shapley value; $\Phi_m \geq \Phi_j$ for all $j \in F$. Then, the Shapley value of the player m' has the following lower bound,*

$$\Phi_m \geq \frac{V(F)}{|F|}.$$

Proof. By the property of efficiency [6], we have $\sum_{k=0}^{|F|-1} \Phi_k = V(F)$. Suppose that Φ_m is the maximum Shapley value. We proceed by contradiction; let $\Phi_m = (V(F)/|F|) - \epsilon$ for $\epsilon > 0$. It follows that for all k, $\Phi_k \leq (V(F)/|F|) - \epsilon$. Thus,

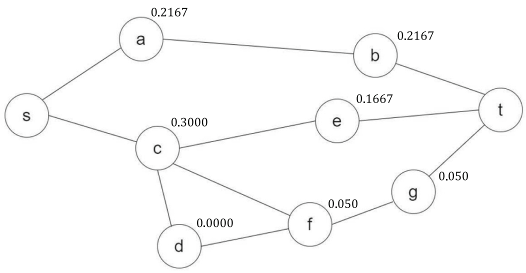
$$V(F) = \sum_{k=0}^{|F|-1} \Phi_k \leq \sum_{k=0}^{|F|-1} ((V(F)/|F|) - \epsilon) = V(F) - |F|\epsilon. \tag{9}$$

A contradiction, therefore Φ_m cannot be less than $V(F)/|F|$. □

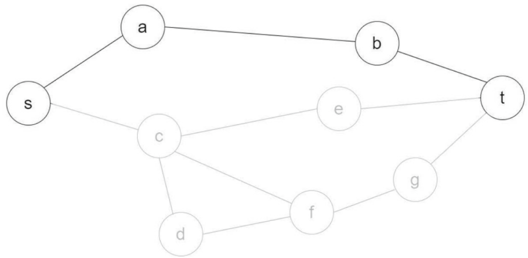
As a result, when looking for an important node, at worst, we need precision that is inversely proportional to $V(F)/|F|$. Thus, to find our importance nodes, we create a uniform superposition of the nodes stored in the `Ind` register, where each has the same probability, $(1/|F|) \sum_{k \in F} |k\rangle_{\texttt{Ind}}$. We perform our combined algorithm to assess the Shapley values in the st-connectivity game, storing the results $\tilde{\Phi}_k \approx \Phi_k$ in a new `Shp` register,

$$\frac{1}{|F|} \sum_{k \in F} |k\rangle_{\texttt{Ind}} \left|\tilde{\Phi}_k\right\rangle_{\texttt{Shp}},$$

where $|\tilde{\Phi}_k - \Phi_k| \leq \mathcal{O}(V(F)/|F|)$. We can find the k so that $\tilde{\Phi}_k$ is maximized in $\mathcal{O}(\sqrt{|F|})$ applications of the combined algorithm using a quantum algorithm to find the maximum [1]. If there are multiple high-value players or the most valuable player is not yet found, the algorithm can be repeated to find multiple high-value players.



(a)



(b)

Fig. 1. Practical example (see our companion Github repository for further details). (a) Shapley Values for the intermediate nodes between s and t. (b) Coalitions of Nodes (in which coalitions of nodes are represented by binary string 1100000).

3.4 Practical Example

Let $H = (N, E)$ be the graph as shown in Fig. 1a. We define cooperative game $G_H = (F, V_{st})$, with $s, t \in N$ and $F = N \setminus \{s, t\}$. Suppose that we wish to find the Shapley value Φ_a of node $a \in F$. We can represent each subset $R \in F$ with binary string $b^R = b_a^R \cdots b_g^R$ where b_j^R is equal to 1 if $j \in R$ else b_j^R is 0. Note that, $V(R)$ is either 0 or 1, so we can take $V_{\max} = 1$ and $V_{\min} = 0$. We define U_V as we did in the previous section. For example, suppose that we apply U_V with input string $|1100000\rangle_{\texttt{P1}}$, this represents the subgraph in Fig. 1b. This subset is st-connected, since the path s, a, b, t is valid. As a result, if we perform $U_V\,|1100000\rangle_{\texttt{P1}}\,|0\rangle_{\texttt{Ut}}$, the `Ut` register stores the correct answer 1, with probability $1-\kappa$. However, if we removed node a, the graph would no longer be st-connected. So, the state $U_V\,|0100000\rangle_{\texttt{P1}}\,|0\rangle_{\texttt{Ut}}$ has the answer 0 stored in the `Ut` register with probability $1 - \kappa$.

To find Shapley value Φ_a, we proceed as follows: (i) craft a quantum state that encodes every possible subset of nodes, that does not include node a, with correct amplitude probability weights corresponding to γ (Definition 3); (ii) perform the unitary U_V outputting to `Ut`, i.e., repeatedly check for st-connectivity leveraging Theorem 2 and take the majority answer; (iii) extract the expected value of the utility register `Ut` using the Monte-Carlo speed-up [13]; (iv) repeat the previous steps where each subset that includes node a is considered and compared outputs. Using this strategy, we can approximate the Shapley values of each node with arbitrary precision (see Fig. 1a). As a result, we can also take advantage of the techniques described in Sect. 3.3, to quickly identify which nodes have the highest Shapley values.

3.5 Complexity Analysis

Baseline Classical Complexity – We now describe a reasonable, though not necessarily optimal method to approximate *st*-connectivity based node-centrality through classical methods. Let $G_H = (F, V_{st})$ be a cooperative game in $H = (N, E)$, where $s, t \in N$ and $F = N \setminus \{s, t\}$. Let us discuss the complexity of approximating the Shapley value of the player *i*', Φ_i. The *st*-connectivity can be assessed using a breadth-first search, with a time complexity of $\mathcal{O}(|N|^2)$. By Chebyshev's inequality, we need to query the *st*-connectivity algorithm $\mathcal{O}(\sigma^2/\epsilon^2)$ times, where ϵ is the desired error, and σ^2 is the variance of V_{st} over the distribution matching the Shapley value Definition 3. Since the only outputs of V_{st} are zero and one, we effectively have a Bernoulli distribution with the expected value Φ_i. Thus, the variance is $\Phi_i(1 - \Phi_i)$. Since nontrivial situations do not allow Φ_i to be close to one, we effectively have a variance of $\mathcal{O}(\Phi_i)$. Thus, given a fixed likelihood of success, the time complexity of approximating the Shapley value Φ_i with an error bounded by ϵ is $\mathcal{O}\left(\Phi_i \epsilon^{-2}|N|^2\right)$.

Next, we briefly consider a method to extract important nodes. In the worst case, the largest Shapley value is of size $\mathcal{O}(V(F)/|F|) = \mathcal{O}(1/|N|)$, and in this case, most values are close together. So, an error bound $\epsilon \in \mathcal{O}(1/|N|)$ and Shapley value $\Phi_i \in \mathcal{O}(1/|N|)$ are appropriate values. Thus, we require $\mathcal{O}(|N|^3)$ operations for sufficient accuracy. Finally, we must find the Shapley value for each node, thus, naively, the worst-case scenario involves about $\mathcal{O}(|N|^4)$ operations.

Quantum Complexity – Finally, let us address the complexity of our quantum approach. Note that in this section, we drop polylogarithmic factors for notational simplicity. We now describe the complexity of approximating the Shapley value of the player *i*', Φ_i, with quantum methods. U_V involves repeating the algorithm from Theorem 2 a logarithmic number of times. Thus, U_V has a time complexity of $\tilde{\mathcal{O}}\left(|N|^{3/2}\right)$. Note that Theorem 2 implicitly requires an easily addressable form of adjacency matrix, which can be implemented with `qRAM`, or possibly through native interactions with a quantum network. In this context, the Shapley value algorithm has complexity $\tilde{\mathcal{O}}\left(\sqrt{\Phi_i}/\epsilon\right)$ (Theorem 3). Thus, the complexity in finding the Shapley value of node *i*' is $\tilde{\mathcal{O}}\left(\sqrt{\Phi_i}\epsilon^{-1}|N|^{3/2}\right)$.

Applying the same rationale as above, we consider the problem of extracting important nodes. Suppose that the largest Shapley value is $\mathcal{O}(1/|N|)$ and as a result we want $\epsilon \in \mathcal{O}(1/|N|)$. Thus, calculating the Shapley values to the required precision takes $\tilde{\mathcal{O}}(|N|^2)$ time. As discussed in Sect. 3.3, we can approximate all Shapley values in superposition, then extract the maximum in $\tilde{\mathcal{O}}\left(\sqrt{|N|}\right)$ queries. Thus, our total complexity for finding important nodes takes $\tilde{\mathcal{O}}(|N|^{5/2})$ operations.

4 Related Work

The work presented in this paper combines quantum computing with distributed system security. Some existing research directions related to our work include (i) the study of potential advantage or speed-up optimizations of quantum computing associated with probing, control, and planning of cyber-physical systems [2], as well as formally verifying properties and providing explainability of the related processes [11]; (ii) use of quantum technologies to secure quantum data communications (e.g., protecting the authenticity of quantum signals when in transit, the detection of adversaries maliciously modifying quantum messages, and analysis of any other threat models affecting the security of entanglement rates to endanger applications built upon distributed quantum networks [3]); (iii) advantages of quantum technologies to build more secure ways to protect classical data with key expansion protocols like QKD, any of its flavors [14]; (iv) risks and threats posed by quantum science to contemporary information security, including the use of quantum annealers or any other quantum-inspired metaheuristics paving the way for new cracking strategies against classical or post-quantum cryptography [8].

Compared to previous work, we provide in this paper a formal approach built on game-theoretic node centrality following in line with [12,18]. Game-theoretic node centrality provides a more flexible and nuanced concept of node centrality. The *st*-connectivity attack, in the context of game-theoretic node centrality, relies on novel methods to quantify the security properties of a graph. As previously shown [5], the Shapley values necessary for our node centrality can be approximated with quadratically fewer value function queries using quantum methods, up to polylogarithmic factors. Moreover, our value function, based on *st*-connectivity, can be assessed faster on a quantum computer by taking advantage of [4]. The combination of these two factors allows for a faster calculation than is possible with a classical Monte Carlo approach to solving the problem. Finally, to find high-importance nodes, we can calculate each node's Shapley value simultaneously using quantum superposition, which yields a database of Shapley values. We then search this database of nodes to find the node with the largest Shapley value; this is quadratically faster than a classical search would allow [1].

5 Conclusion

We have described a quantum approach to approximating the importance of nodes that maintain a target connection, as well as how to quickly identify high-importance nodes. Our methods are built upon multiple subroutines: one for *st*-connectivity, another for the Shapley value approximation, and a final subroutine for finding the maximum of a list. We have considered a formal attack scenario, denoted as the *st*-connectivity attack, in which a malicious actor disrupts a subset of nodes to disrupt the system functionality. Using our methods, we can identify the nodes that are the most important and use this information to guide topological adjustments to increase resilience. Our solution can also be used as a security metric to guide remediation strategies. Perspectives for future work include investigating extended subroutines that leverage node values to improve resilience.

Acknowledgments. Authors acknowledge support from the CyberCNI chair (Cybersecurity for Critical Networked Infrastructures) of Institut Mines-Télécom.

References

1. Ahuja, A., Kapoor, S.: A quantum algorithm for finding the maximum (1999). https://arxiv.org/abs/quant-ph/9911082
2. Barbeau, M., Garcia-Alfaro, J.: Cyber-physical defense in the quantum era. Sci. Rep. **12**(1), 1905 (2022)
3. Barbeau, M., Kranakis, E., Perez, N.: Authenticity, integrity, and replay protection in quantum data communications and networking. ACM Trans. Quant. Comput. **3**(2), 1–22 (2022)
4. Belovs, A., Reichardt, B.W.: Span programs and quantum algorithms for st-connectivity and claw detection. In: European Symposium on Algorithms, pp. 193–204. Springer (2012)
5. Burge, I., Barbeau, M., Garcia-Alfaro, J.: Quantum algorithms for shapley value calculation. In: 2023 IEEE International Conference on Quantum Computing and Engineering (QCE), vol. 1, pp. 1–9. IEEE (2023)
6. Burge, I., Barbeau, M., Garcia-Alfaro, J.: A shapley value estimation speedup for efficient explainable quantum AI. arXiv preprint arXiv:2412.14639 (2024)
7. Cade, C., Montanaro, A., Belovs, A.: Time and space efficient quantum algorithms for detecting cycles and testing bipartiteness. arXiv preprint arXiv:1610.00581 (2016)
8. Campbell, R., Diffie, W., Robinson, C.: Advancements in quantum computing and AI may impact PQC migration timelines (2024). https://doi.org/10.20944/preprints202402.1299.v1
9. Cao, Y., Zhao, Y., Wang, Q., Zhang, J., Ng, S.X., Hanzo, L.: The evolution of quantum key distribution networks: on the road to the QInternet. IEEE Communicat. Surv. Tutor. **24**(2), 839–894 (2022)
10. Castro, J., Gómez, D., Tejada, J.: Polynomial calculation of the Shapley value based on sampling. Comput. Oper. Res. **36**(5), 1726–1730 (2009)
11. Chareton, C., Bardin, S., Bobot, F., Perrelle, V., Valiron, B.: An automated deductive verification framework for circuit-building quantum programs. In: Programming Languages and Systems: 30th European Symposium on Programming, ESOP 2021, pp. 148–177. Springer (2021)
12. Michalak, T.P., Aadithya, K.V., Szczepanski, P.L., Ravindran, B., Jennings, N.R.: Efficient computation of the shapley value for game-theoretic network centrality. J. Artif. Intell. Res. **46**, 607–650 (2013)
13. Montanaro, A.: Quantum speedup of Monte Carlo methods. Proc. Roy. Soc. A: Math. Phys. Eng. Sci. **471**(2181), 20150301 (2015)
14. Noirie, L.: From existing quantum key distribution systems towards future quantum networks. In: 13th International Conference on Communications, Circuits, and Systems (ICCCAS 2024) (2024)
15. Robbins, H.: A remark on Stirling's formula. Am. Math. Mon. **62**(1), 26–29 (1955)
16. Shapley, L.S.: A Value for N-Person Games. RAND Corporation, Santa Monica (1952). https://doi.org/10.7249/P0295
17. Szegedy, M.: Quantum speed-up of Markov chain based algorithms. In: 45th Annual IEEE Symposium on Foundations of Computer Science, pp. 32–41. IEEE (2004)
18. Tarkowski, M.K., Michalak, T.P., Rahwan, T., Wooldridge, M.: Game-theoretic network centrality: a review. arXiv preprint arXiv:1801.00218 (2017)