Opportunistic Routing in Quantum Networks

Ali Farahbakhsh and Chen Feng
School of Engineering
University of British Columbia (Okanagan Campus)
Kelowna, British Columbia, Canada, V1V 1V7
Emails: ali.farahbakhsh@ubc.ca, chen.feng@ubc.ca

Abstract—Unlike classical routing algorithms, quantum routing algorithms make use of entangled states—a type of resources that have a limited lifetime and need to be regenerated after consumption. In a nutshell, quantum routing algorithms have to use these resources efficiently, while optimizing some objectives such as the total waiting time. Current routing algorithms tend to keep a routing request waiting until all of the resources on its path are available. In this paper, we introduce a new way of managing entanglement resources in an opportunistic fashion: a request can move forward along its path as soon as possible (even if some resources on its path are not ready). We show that this opportunistic approach is fundamentally better than conventional approaches. In particular, our results indicate that this new approach achieves a 30-50% improvement in the average total waiting time and average link waiting time compared with several state-of-the-art routing algorithms. As a by-product of this work, we develop a new simulator for quantum routing, which can be used to evaluate various design choices under different scenarios.

I. INTRODUCTION

Quantum systems are able to offer superior advantages for certain applications (e.g., [1]–[6]). These applications either do not have a classical solution, or their classical solutions have significantly worse performance. The spectacular laws of quantum mechanics allow quantum systems to be inherently more suitable for certain complex problems.

An indispensable part of most of these quantum systems is a quantum network. Quantum networks are used to transfer quantum information between quantum computers, and this information is conveyed through qubits, which are the quantum counterparts of bits. The promising performance and applications of quantum networks serve as the building ground for a future quantum Internet [7].

Unfortunately, the same laws of quantum mechanics also create spectacular challenges specific to quantum systems (see, e.g., [8]). These challenges make quantum networks fundamentally different than their classical counterparts, and require new methods to deal with them. Technological limitations, noise, and interaction with the environment are among the main reasons for these challenges [9]–[12]. Specifically, the state of a quantum network is constantly changing. This is partly due to the fact that the links of a quantum network have finite lifetimes, and partly because their generation is probabilistic.

The highly dynamic nature of quantum networks makes routing in such networks highly non-trivial. Traditional (classical) routing algorithms rely on the robustness of the network to a great extent, and then design recovery procedures in which changes in the state of the links and the nodes are effectively handled. In quantum networks, the state is constantly changing, so we need routing algorithms that deal with this inherent dynamism as the main issue rather than a side case. The basis of such algorithms would be an efficient approach towards allocating the limited and temporary resources of a quantum network to the incoming requests, while managing the constant regeneration of the resources.

Most of the existing quantum routing algorithms, such as [9], [13], [14], make a design choice that adversely affects the overall performance. Basically, existing routing algorithms wait for all of the resources (i.e., links) a request needs to be ready and then start to forward the request on its selected path. This design choice is conservative and leads to inefficient usage of the network resources. We argue that a more opportunistic approach better suits the unique characteristics of quantum networks. In particular, we believe that a request should be forwarded along the path as soon as it is possible to move forward, even if it is a single hop. As we will show in this paper, such opportunism increases the efficiency of any routing algorithm by decreasing the average total time required for a group of requests to reach their destinations.

To the best of our knowledge, we are the first to demonstrate and analyze opportunism in quantum networks. In particular, we have implemented three existing routing algorithms and demonstrated the fundamental superiority of opportunism by using both theoretical analysis and simulations. Moreover, we have shown that opportunism can be added to almost any routing algorithm as an add-on. Finally, we have designed and implemented a simulator that can serve as a common ground for implementing several routing algorithms with various design choices. In fact, all three of the mentioned algorithms have been implemented with this simulator. We will open-source our simulator, making it a useful tool for quantum routing research. Our simulator has integrated the essential design choices of quantum routing, such as multipath routing [14], [15], having recovery paths [13], and having contention-free paths.

In summary, our contributions are as follows:

- We introduce and analyze opportunism as a new perspective for managing quantum networks, and demonstrate its superiority with both theoretical analysis and simulations.
- We provide a simulator that can be used to implement different quantum routing protocols since it provides
common functionalities and many design choices. It can also be used to compare state-of-the-art protocols through different metrics and setups.

The rest of the paper is organized as follows. Section II provides an overview of quantum networks and quantum routing. Section III delves into opportunistic routing. Section IV describes our system model. Section V presents our theoretical analysis. The simulation results, together with a brief explanation of our simulator, are presented in Section VI. Finally, the related work, and conclusion and future work are presented in Sections VII, and VIII, respectively.

II. QUANTUM NETWORKS

A. Components

Similar to many other networks, a quantum network consists of nodes and links. The nodes of a quantum network are quantum computers, which are connected through both classical and quantum channels (such as fiber optics). The links of a quantum network are entangled qubit pairs (between two nodes), because they play a crucial role in quantum communication as explained below.

The quality of qubit transmission over physical links degrades exponentially as the length of the link increases [10], [11], [17]. Therefore, quantum teleportation [18] is used to “transfer” quantum information. More specifically, during quantum teleportation, an entangled qubit pair between two nodes is consumed to teleport a qubit between them. This qubit is not sent via a physical link but is obtained at the destination through a series of measurements and classical communication.

Generalizing quantum teleportation, entanglement swapping [19] enables qubit transfer between two arbitrarily distant nodes by using a chain of intermediate nodes which perform a series of measurements and operations. Basically, a simple swapping protocol consumes entangled qubits between two nodes with an intermediate node to establish entanglement between these two nodes. Fig. 1 illustrates a simple swapping scenario. Several protocols have been proposed for entanglement swapping (e.g., [11], [20], [21]).

To sum up, entangled pairs indeed play a crucial role in quantum networks, serving as the basic unit of resource required for successful quantum communication. This explains why they are considered as links of a quantum network. Throughout this paper, we will use the terms “generating” or “establishing” a link to refer to creating a pair of entangled qubit pairs between two adjacent nodes.

B. Limitations

Quantum networks are generally hard to manage due to the following limitations.

1) Nodes: Current prototypes for quantum computers have rather limited storage capacity for qubits [22]. This limits the number of entanglements (i.e., links) they can establish with other nodes in the network.

2) Qubits: Qubits have different quality metrics, such as fidelity [23]. Generally, the quality of a qubit decreases over time (due to noise and interaction with the environment). A qubit may become useless after a certain threshold, which depends on the application (e.g., entanglement distillation [24]). In other words, qubits, and therefore links, have finite lifetimes.

3) Links: As explained before, links of a quantum network have finite lifetimes. Moreover, the process of generating a link is probabilistic [9], [10], [13]. In each attempt, there is a probability of failure.

4) Swapping: Similar to link generation, each swapping attempt is also probabilistic with a certain failure probability [11].

C. Quantum Routing

With quantum teleportation and swapping, quantum communication can be carried out in a line network. However, a quantum network may have a more complicated topology, making the transfer of qubits nontrivial. To cope with such complexity as well as the dynamic nature of a quantum network as explained in Section II-B, a typical quantum routing algorithm contains the following stages [13]:

1) Reception: First, a number of routing requests are generated in the network.

2) Path Selection: Based on certain rules (see, e.g., [25]), several paths are selected for every request, either offline [13] or on the fly. The links of these paths are usually reserved for the corresponding request.

3) Generation: Then, links are generated along the paths for every request.

4) Forwarding: Finally, once all of the required links are ready along a path, the corresponding request is forwarded based on a certain swapping pattern.

Indeed, most existing routing algorithms follow the above order with few exceptions having a little modification. For example, the algorithm proposed in [14] performs the “Path
Selection” stage after the “Generation” stage (whereas many other algorithms do the opposite).

Unlike most of the existing algorithms, our proposed approach is not related to specific details like how the paths are selected, and is instead more related to efficient management of the resources. We achieve this by proposing a new forwarding stage for any type of routing algorithms. Specifically, as explained in Section III and demonstrated in Section VI, our proposed approach, which is opportunistic in nature, can be used as an add-on to many existing routing algorithms.

III. OPPORTUNISTIC ROUTING

In this section, we give a high-level overview of our opportunistic routing. Our key observation is the following: Most existing routing protocols conduct the forwarding stage rather conservatively. More specifically, a typical routing algorithm (e.g., [9], [13], [14]) often waits until all of the links in a path are ready and then forwards the request by initiating the swapping process. We believe that such a conservative approach fails to exploit the highly dynamic nature of quantum networks for the following reasons.

First, since links have finite lifetimes, it is inefficient to wait for all of the links of a particular path to get established. For example, some already established links may time out even before the remaining links get established, leading to a waste of resources. Second, when a request reserves the links of its path, these links cannot be reserved by other requests. Third, since swapping is also probabilistic and it takes time to finish, by waiting for all of the links to get generated, we unnecessarily lose the opportunity to do some of the swappings while the generation is still not finished. Therefore, by being less conservative, we can enable two types of gain. By leveraging the resources more efficiently, we can allow every request to reach its destination faster, and at the same time prevent it from being a hindrance to other requests.

Based on the above explanations, we propose a modified forwarding stage as follows:

4) Forwarding: The request is forwarded along the path as soon as it is possible, even if it is only a single hop.

Fig. 2 shows an example of being opportunistic in a simple scenario. The blue (pink) request is going from the leftmost (rightmost) node to the rightmost (leftmost) node. At each step, a random number of links are generated (green links), and then the requests consume the generated links, restoring them to the non-generated (black) state. Instead of waiting for all of the links to be ready, the requests take the opportunity of moving forward when they can. For simplicity, the intermediate steps in which links consecutively fail to get generated are omitted.

The gain caused by being opportunistic is worth more explanation. To this end, imagine a scenario in which \( N \) requests want to go from the same source to the same destination. Fig. 3 shows such a scenario. With an ordinary, non-opportunistic routing algorithm, each one of these requests has to wait for all of the links of the path to be ready, and then it gets forwarded. This means that the requests have to wait \( N \) times the time needed for a single request, on average. However, by acting opportunistically, we can exploit the generated links as soon as it is possible for a request to move forward, and then regenerate the consumed links for the other requests. This clearly demonstrates the two-fold gain of opportunism: 1) each request reaches the destination sooner because a part of the waiting time for the link generation is used to perform some of the swappings, and 2) resources are freed more quickly, resulting in less waiting time for requests which are waiting for the prior requests to complete transmission. We call the first gain as the swapping gain and the second one as the reservation gain.

There is a subtle point about the forwarding stage, which is worth mentioning. Forwarding a request can be interpreted in two ways. First, the request may actually be forwarded along the path, as a continuation of the swapping process. That is, as soon as a swapping step is successful along the path, the request can be forwarded to the corresponding intermediate node using teleportation. Alternatively, the request can wait until the swapping process is completely finished, and then it can be sent from the source to the destination using a single quantum teleportation step. Both of these approaches can be considered opportunistic in the sense explained in
the previous paragraphs, as long as the swapping process is started as soon as it is possible. Indeed, this perspective suggests that opportunism can be leveraged in two layers: first in leveraging the generation time for a part of swapping, and then in acting opportunistically during the swapping process. Both of these “forwarding” methods have their advantages and disadvantages. With the former method, the request does not need to start from the beginning if a part of the swapping process fails, while on the other hand, it will consume a part of the corresponding intermediate node’s memory, possibly harming the routing of other requests. With the latter method, no extra resources are used for serving the request, but it has to start from the beginning every time a part of the swapping process fails. For a neat illustration, the figures of this paper demonstrate the former method, so that requests are actually moving forward as more links are generated. However, we have chosen the latter method for swapping in the simulations, and requests have to start from the beginning if their swapping process is not successful.

As a final consideration, as explained in Section II-C, opportunism is an approach targeting the general management of quantum networks, and it can be used in any algorithm since it does not depend on specific details. Details like how the resources are allocated and the paths are selected are not important at this level. The point is to treat the links of a network as precious resources and use them as soon as possible. This mindset can even be extended to be an integral part of a quantum network layer protocol (e.g., [26]).

A. k-opportunism

The opportunism discussed in this section is somewhat greedy and extreme: a request moves as soon as it can move. This may not be harmful in networks with a low number of requests, but the more the requests get, the harder it gets for the network to manage the resources. On a large scale, forwarding a request as soon as a link is available may decrease the efficiency of routing for some other request. Additionally, requests may have different degrees of priority. Therefore, the opportunistic approach needs to have a degree of flexibility to provide dynamic routing means for a highly dynamic quantum network.

To this end, we introduce k-opportunism. Recall that the proposed forwarding stage required a request to move forward once it is possible, even for one hop. We can change this stage as follows:

4) Forwarding: The request is forwarded along the path as soon as the first k immediate and consecutive hops are available.

Fig. 4 shows an example of a 2-opportunistic approach, in which a request is going from the leftmost node to the rightmost node. Here, the request does not move forward unless the two immediate and consecutive links are ready. Intuitively, this new opportunism provides us with a spectrum of choices, allowing us to impose a distributed control over every request’s greediness. Indeed, the extreme opportunism introduced at the beginning of Section III is a 1-opportunistic approach. Now, this spectrum of opportunism, obtained by changing k, which we call the degree of opportunism, provides natural means for managing requests with different priorities. That is, we can assign lower values of k for requests with higher priorities. In this way, high-priority requests will be able to leverage the links of the network sooner than the other requests, and the higher k gets for a request, the lower its chances are to use a link for its transmission.

IV. SYSTEM MODEL

In this section, we present the system model as well as the performance metric.

A. Network, Nodes, and Links

We model a quantum network as a graph. The nodes of the graph are quantum computers, and the edges of the graph are the links (i.e., entangled pairs). For simplicity, we assume that at most one link can exist between any two adjacent nodes, because such a simple setup is sufficient to demonstrate the advantage of our opportunistic routing. Unlike [9], [27], we do not use virtual links, i.e., entanglement between non-adjacent nodes, as a primary resource. The reason for this choice is also similar. We want to demonstrate that links are precious resources and have to be consumed opportunistically, and it does not matter if they are between adjacent or non-adjacent nodes.

We use (s, d) to denote a request. Here, s is the source and d is the destination. When it is clear from the context, we may also use r for a single request, and \(\{r_i\}_{i=1}^{\alpha}\) for a group of requests.

We denote the lifetime of an entangled pair (i.e., a link) as L. We assume that quantum nodes have infinite memories.

B. Entanglement Generation and Swapping

We assume that every entanglement generation attempt between adjacent nodes succeeds with probability \(p_{\text{GEN}} = p\). Entanglement generation for a link is continuously attempted until the link is generated. Furthermore, once a link is consumed or timed out, it starts the generation process again. Finally, we assume that every swapping attempt succeeds with probability \(p_{\text{SWAP}}\).
C. Waiting Time

We divide the waiting time of a request into two parts as follows.

- **Swapping:** It is the time required for a swapping process over a given number of links.
- **Generation:** It is the amount of time it takes for all of the links required for the path of the request to get generated.

We consider a discrete-time model, in which every time step is equal to the time it takes to attempt an entanglement generation between two adjacent nodes. Furthermore, we also assume that a single swapping step takes place at the end of each time slot. That is, each request has a chance of one swapping step at the end of each slot. The waiting times are explained in the following sections.

1) **Swapping waiting time:** Different algorithms have been suggested for entanglement swapping (e.g., [20]). Any algorithm can be selected for this process, and each would lead to a certain formula for the waiting time. We denote the total time it takes to swap over \( M \) links as \( t_{\text{swap}}(M) \).

2) **Generation waiting time:** We denote the amount of time required for a path with \( M \) links to be generated as \( W(M, p) \). We also denote the time required for link \( i \) of this path to get established by \( T_i \). Note that \( \{T_i\} \) are i.i.d. and they follow a geometric distribution with parameter \( p \). Note also that \( W(M, p) \) is a random variable, and we denote \( R(M, p) = E\{W(M, p)\} \). It is shown in [10], [28] that, if \( L = \infty \),

\[
R(M, p) = \sum_{k=1}^{M} \binom{M}{k} \frac{(-1)^{k+1}}{1 - (1 - p)^k}.
\]

Also, note that \( W(M, p) = \max_i \{T_i\} \).

D. Performance Metric

Our main focus is on the average total waiting time, which is the average total time required for a group of requests to reach their destinations. We show this quantity with \( Q(S, M, p) \), in which \( S \) is the group of requests. For simplicity and brevity, we will use \( Q(S) \) instead of \( Q(S, M, p) \). Furthermore, we will use \( Q_{\text{opt}}(S) \) to show the average total waiting time in the opportunistic approach. In addition to \( Q(S) \), we also introduce the average link waiting time, which is the average time links wait to get consumed, after their generation.

V. ANALYSIS

We focus on the swapping and reservation gains in this section. Our goal is to show that the opportunistic approach improves performance by decreasing the average total waiting time. We also show the spectrum property of opportunism by analyzing \( k \)-opportunism.

To make our analysis tractable, we mainly focus on a line network here (which will be relaxed in the simulation section). More specifically, we assume a number of quantum nodes in a single line, in which every node has two neighbors (except the endpoints, which have one). We will call the endpoints of this network as \( A \) and \( B \) throughout the paper. We assume that this line network has \( M \) links.

In addition, we introduce several simplifying assumptions for our analysis. First, we assume \( L = \infty \). That is, a generated link remains intact forever, until a request consumes it; after which the link has to be regenerated. Second, following [9], [11], [28] we assume that \( p_{\text{swap}} = 1 \). Finally, we have used a simple swapping algorithm in which the swapping operations are carried out on a one-by-one basis, starting from the source towards the destination. For this simple scheme, we have \( t_{\text{swap}}(M) = M - 1 \).

A. A Single Request

When sending a single request, the opportunistic approach helps to reduce the total waiting time by mixing the swapping and generation waiting times. In the conventional approach, the request has to wait for the generation to finish, then it can begin the swapping process. The opportunistic approach enables swapping and generation to happen simultaneously, leveraging the time the request has to wait for link generation to do a part of swapping. This is the swapping gain mentioned in Section III.

We consider a single request \( r = (A, B) \). With the opportunistic approach, \( r \) will be somewhere in the middle of the network\(^2\) once all of the links have been generated (i.e., after \( W(M, p) \)). The number of links \( r \) has been able to swap along the way until all of the links are generated is how opportunism shows itself, and indeed this number can be interpreted as the swapping gain caused by opportunism. Let us denote the position of \( r \) after \( W(M, p) \) by \( K \). We are interested in \( E\{K\} \).

Let us use \( W \) to show \( W(M, p) \), for brevity. Furthermore, let us define \( W_i = \max_{j<i} \{T_j\} \). To calculate \( E\{K\} \), we have \( E\{K\} = \sum_{k=1}^{M} P(K \geq k) \). Note that \( K \geq k \) implies that the first \( k - 1 \) links were not the last links to get generated, and \( W \geq k \) since every swapping attempt takes one time step. We thus have \( P(K \geq k) = P(W > W_{k-1}, W \geq k) \)

Let us define \( W = \max_{i,k} \{T_i\} \). We can decompose this event as follows \( P(W > W_{k-1}, W \geq k) = \sum_{i=k}^{\infty} P(W \geq W_{k-1}, W = i) \), which can further be expressed as \( P(W > W_{k-1}, W \geq k) = \sum_{i=k}^{\infty} P(W_{k-1} < i, W = i) \).

Now, since \( W_{k-1} \) and \( W \) are related to two disjoint sets of links, they are independent of each other, and we have \( P(W > W_{k-1}, W \geq k) = \sum_{i=k}^{\infty} P(W_{k-1} < i) P(W = i) \).

We can now evaluate each part separately. First, we have \( P(W_{k-1} < i) = P(\cap_{j=1}^{k-1} \{T_j < i\}) = \prod_{j=1}^{k-1} P(T_j < i) = P(T_1 < i)^{k-1} \). Here we have used the fact that the \( T_j \) are i.i.d. It is easy to see that \( P(T_1 < i) = 1 - (1 - p)^{i-1} \), and letting \( q = 1 - p \), we have \( P(W_{k-1} < i) = (1 - q^{i-1})^{k-1} \).

Now, we have \( P(W = i) = P(W \leq i) - P(W < i) \), which can be calculated very similar to the procedure above, and one would have \( P(W = i) = (1 - q^i)^{M-k+1} - (1 - q^{i-1})^{M-k+1} \).

\(^2\)The request is actually in the source, but the swapping process has reached the middle of the network. See Section III.
Combining these results leads us to $P(K \geq k) = \sum_{i=k}^{\infty} (1 - q^{i-1})^{k-1} M^{k-1} M^{-k+1}$. Therefore, we have

$$E[K] = \sum_{k=1}^{M} \sum_{i=k}^{\infty} (1 - q^{i-1})^{k-1} \left[ (1 - q^{i})^{M-k-1} - (1 - q^{i-1})^{M-k+1} \right].$$

(2)

Now, we have $P(K > 0) > 0$. Furthermore, $r$ has to swap along $M - K$ links after $W(M, p)$, and $t_{swap}(M - K) < t_{swap}(M)$. We have $Q_{opt}(S) = R(M, p) + E\{t_{swap}(M - K)\}$. Thus it is easy to see that $Q_{opt}(S) < Q(S) = R(M, p) + t_{swap}(M)$. This proves the following theorem.

Theorem 5.1: For a single request $S = \{(A, B)\}$, we have $Q_{opt}(S) < Q(S)$.

B. A Group of Requests in the Same Direction

Here, we demonstrate the reservation gain caused by opportunism. To do this, we will assume $N$ requests going from $A$ to $B$. Note that to show opportunism decreases the total waiting time in this scenario, all one has to do is to leverage the swapping gain, as explained in Section V-A, for every request. However, as discussed in Section III, opportunism enables another gain, which we call the reservation gain, and this gain has a different nature than the swapping gain.

To fully demonstrate the reservation gain, we will assume that swapping takes no time (i.e., $t_{swap}(M) = 0$). This way, a request can perform as many swapping steps as it can in a single time step. In this new setting, the swapping gain will vanish, and in the case of a single request, opportunism will not improve the performance. However, when having multiple requests, we can show that opportunism decreases the total waiting time, thus demonstrating the existence of what we call the reservation gain.

Let us use $T_{ij}$, where $i \leq M$ and $j \leq N$, to show the time it takes to generate the $i$-th link for the $j$-th time (i.e., for the $j$-th request), once it has been consumed for the $j-1$-th request. Each $T_{ij}$ follows the geometric distribution with parameter $p$, and $T_{ij}$ are independent. In the non-opportunistic approach, each request has to wait for the previous request to reach the destination. Since swapping takes no time, this is equal to the generation time of $M$ links. That is, the $j$-th request has to wait for the previous $j-1$ requests, and then an additional amount equal to $W_j := \max_{i \leq M} T_{ij}$. Therefore, by induction, the total waiting time is equal to $\sum_{j=1}^{N} W_j$. We thus have $Q(S) = E\{\sum_{j=1}^{N} W_j\}$.

In the opportunistic setting, we need a recursive approach to formulate $Q_{opt}(S)$. Let us define $W_{ij} = \max_{k \leq i} (W_{k, j-1} + T_{kj})$, and $W_{i1} = \max_{k \leq i} T_{k1}$.

Lemma 5.2: For $S = \{r_j\}_{j=1}^{N}$, in which for every $j$ we have $r_j = (A, B)$, the total waiting time is equal to $W_{MN}$, and we thus have $Q_{opt}(S) = E\{W_{MN}\}$.

Proof: We prove this lemma by induction. First, we show that the lemma holds for $N = 2$. Link $i$ will start generation for $r_2$ once it has been generated and consumed by $r_1$. It will be consumed once all of the first $i$ links are generated for $r_1$, so that $r_1$ can take the opportunity (remember that swapping takes no time). Hence, link $i$ can start its generation for $r_2$ at time $W_{i1} + T_{i1}$. Since this generation has taken place, now link $i$ takes $T_{i2}$ time steps to get generated for $r_2$, which means $r_2$ should wait a total amount of $W_{i1} + T_{i2}$ time slots for the $i$-th link to get generated. Since there is no swapping gain, we can conclude that the total waiting time is equal to $\max_{i \leq M} (W_{i1} + T_{i2})$, which is the formula for $W_{MN}$. This means that $Q_{opt}(S) = E\{W_{MN}\}$.

We now assume that the lemma is true for $i \leq N - 1$, and prove it is also true for $N$. The proof is similar. Request $r_N$ has to wait for $r_{N-1}$ to consume link $i$, and then link $i$ can start the generation process for the $N$-th time. Request $r_{N-1}$ can consume link $i$ once it can consume all of the first $i$ links, and this takes $T_{iN}$ time slots for the $N$-th generation of link $i$ to succeed. Thus, link $i$ is ready for $r_N$ at time $W_{i1} + T_{i1} + T_{i2} + T_{i3} + \cdots + T_{iN}$, which means $r_N$ will have all of the links ready and reach the destination at time $\max_{i \leq M} (W_{i1} + T_{i1} + T_{i2} + T_{i3} + \cdots + T_{iN})$. This is exactly the definition of $W_{MN}$, and leads us to $Q_{opt}(S) = E\{W_{MN}\}$.

Based on this lemma, we can show that $Q_{opt}(S) < Q(S)$.

Theorem 5.3: for $S = \{r_j\}_{j=1}^{N}$, in which for every $i$ we have $r_i = (A, B)$, we have $Q_{opt}(S) < Q(S)$.

Proof: We will prove this theorem by induction. First, we will prove for the case $N = 2$. We have $W_{M2} = \max_{i \leq M} (W_{i1} + T_{i2}) \leq \max_{i \leq M} W_{i1} + \max_{i \leq M} T_{i2}$.

Now, since $W_{i1} = \max_{k \leq i} T_{k1}$, we have $\max_{i \leq M} W_{i1} = \max_{i \leq M} T_{i1}$. This leads us to $W_{M2} \leq \max_{i \leq M} T_{i1} + \max_{i \leq M} T_{i2} = W_{1} + W_{2}$, and by taking the expected value, to $Q_{opt}(S) \leq Q(S)$. To show that the inequality is strict, notice that we have equality when $\max_{i \leq M} T_{i1}$ and $\max_{i \leq M} T_{i2}$ are coincident on some $i \leq M$. Let us call such an event $E_1$. It can be shown that $0 < P(E_1) < 1$. We now have $Q_{opt}(S) = E\{W_{M2, 1 \{E_1\}}\} + E\{(W_{1} + W_{2})\cdot 1\{E_1\}\}$, which leads to $Q_{opt}(S) < E\{W_{1} + W_{2}\} \cdot 1\{E_1\} + E\{(W_{1} + W_{2})\cdot 1\{E_1\}\} = E\{W_{1} + W_{2}\}$, which is $Q(S)$, since $0 < P(E_1) < 1$.

Now we assume the theorem is true for $N - 1$, and prove it is also true for $N$. We have $W_{MN} = \max_{i \leq M} (W_{i, N-1} + T_{iN}) \leq \max_{i \leq M} W_{i, N-1} + \max_{i \leq M} T_{iN}$. Now, the recursively defined nature of $W_{i, N-1}$ leads us to $\max_{i \leq M} W_{i, N-1} = W_{MN}$, since the set of outcomes $W_{MN}$ takes a maximum on contains the corresponding sets for all $W_{i, N-1}$. We thus have $W_{MN} \leq W_{MN-1} + \max_{i \leq M} T_{iN}$. Taking expectation leads us to $Q_{opt}(S) \leq E\{W_{MN-1}\} + E\{W_{N}\}$. The correctness of the theorem for $N - 1$ gives us $E\{W_{MN-1}\} < E\{\sum_{i=1}^{N} W_i\}$, and summing this with the previous inequality results in $Q_{opt}(S) = E\{W_{MN}\} < E\{\sum_{i=1}^{N} W_i\} = Q(S)$, proving the theorem correct for $N$.

C. k-opportunism

Finally, we provide a theorem about k-opportunism, and show the spectrum property it provides. In the following theorem, we show the average total waiting time of a group $S$ of requests in the k-opportunistic setting with $Q_k(S)$, and omit the opp subscript for simplicity.
Theorem 5.4: For $S = \{(A, B)\}$ we have $Q_k(S) < Q_{k+1}(S)$.

Proof: Let us call the $k$-opportunistic and $(k+1)$-opportunistic requests as $r_k$ and $r_{k+1}$, respectively. We consider two line networks, with each one of the requests traversing one of the networks. We also assume that link generation in both networks is the same, meaning that whenever a link is generated, it is generated in both networks. This allows us to analyze both requests in completely similar outcomes of the probabilistic link generation process, and show that when taking the average on all of the possible outcomes, we have $Q_k(S) < Q_{k+1}(S)$.

Both of the requests have to wait for $W(M, p)$ time steps until all of the links are established. We assume that this time has passed. Then there are two possibilities: Either both of the requests are at the same place somewhere along the path, or $r_k$ is ahead of $r_{k+1}$. Note that it is not possible for $r_{k+1}$ to be ahead of $r_k$, since at each time slot in which $r_{k+1}$ can move, $r_k$ can move too. Let us show the event in which $r_k$ is ahead of $r_{k+1}$ with $E$.

We denote the number of remaining links (for swapping) for $r_k$ and $r_{k+1}$ as $N_k$ and $N_{k+1}$, respectively. Notice that $N_k$ and $N_{k+1}$ are random variables. If $E$ has not happened, then we use $Z = N_k = N_{k+1}$ to show the remaining links for both of the requests, and otherwise we have $N_k < N_{k+1}$. We can formulate the average total waiting time of $r_k$ as follows

$$Q_k(S) = P(E)(R(M, p) + E\{t_{\text{SWAP}}(N_k)\}) + (1 - P(E))(R(M, p) + E\{t_{\text{SWAP}}(Z)\}).$$

(3)

Similarly for $r_{k+1}$ we have

$$Q_{k+1}(S) = P(E)(R(M, p) + E\{t_{\text{SWAP}}(N_{k+1})\}) + (1 - P(E))(R(M, p) + E\{t_{\text{SWAP}}(Z)\}).$$

(4)

When $E$ has not happened, we have $N_k < N_{k+1}$, which means $E\{t_{\text{SWAP}}(N_k)\} < E\{t_{\text{SWAP}}(N_{k+1})\}$. Therefore, based on Equations (3) and (4) we conclude that $Q_k(S) < Q_{k+1}(S)$.

VI. SIMULATIONS

We will demonstrate our experimental results in this section. Our simulation setup is much more general than our analysis setup in the previous section. For instance, we use a grid topology for simulation, which automatically covers many complex routing scenarios. We have also relaxed the assumptions of the analysis section, except for the swapping method. Before showing the evaluation results, we will first provide a brief explanation of the simulator we have developed, QNS-Lite [29].

A. QNS-Lite

QNS-Lite is the simulator developed during our research on opportunism. The main goal behind its design is to provide a set of abstractions that faithfully represent the physical rules of quantum networks, while hiding all the quantum mechanical details. Specifically, quantum phenomena are captured by mathematical models which govern the state of the quantum simulator at any given time.

Following the existing literature on quantum routing (e.g., [9], [13]), we have adopted the practice of dividing time into time slots, and furthermore, based on the structure explained in Section II-C, each time slot is divided into several stages. Because of our focus on demonstrating the essential superiority of opportunism, we have kept some aspects of the design simple, and improving these aspects is a part of our future work. This being said, most of the design choices required for creating and fine-tuning a quantum routing algorithm are crafted into the simulator, hence making it a suitable tool for research and development.

B. Evaluation

In order to provide and explain our experimental results, we first explain our setup, metrics, and methodology, as follows.

1) Setup: In our setup, the source and the destination of each request are randomly selected, based on a uniform distribution on the nodes. Once the requests are generated, the runtime of the simulator starts, and it finishes when all of the requests have reached their destinations. This can be thought of as giving an impulse as the input to a system and measuring its impulse response. All of the metrics are measured during the runtime.

Each configuration of interest for this paper can be shown as the tuple $(p_{\text{GEN}}, p_{\text{SWAP}}, L, N, M, k)$, in which $p_{\text{GEN}}$ is the generation success probability, $p_{\text{SWAP}}$ is the swapping success probability, $L$ is the lifetime of a link, $N$ is the number of requests, $M$ is a parameter determining the size of the network, and $k$ is the opportunism degree. In a grid network, the nodes are arranged in an $M \times M$ mesh.

2) Metrics: The main metric for our evaluations is the average total waiting time for requests, as explained in Section V. The lower this metric becomes, the sooner (on average) the requests reach their destinations, and the higher the average throughput of the network. In addition to this metric, we briefly evaluate the average link waiting time. The lower this metric gets, the more efficient resource consumption becomes. We only demonstrate that in all of the implemented algorithms, opportunism helps reduce these metrics, and we do not provide a full comparison of the algorithms.

3) Methodology: For every $(p_{\text{GEN}}, p_{\text{SWAP}}, L, N, M, k)$ configuration, we first generate $N$ requests in the network and then run each algorithm 100 times. We measure each metric in every runtime and calculate its average across the 100 iterations. Then, we repeat this procedure another 50 times, each time with a different set of requests. Each one of these outer iterations produces an average total waiting time for its corresponding $N$ requests, and an average link waiting time for the links. We calculate the average of these average values, and then compare them across the different algorithms we have implemented. This allows us to assess the essence of the overall effect of opportunism on the performance of a given network, by including many groups of requests in many scenarios.
C. Results

We have implemented three state-of-the-art algorithms alongside their opportunistic versions. These algorithms are Modified Greedy (MG) [9], Nonoblivious Local (NL, we have chosen this name for this algorithm for ease of use) [14], and QPASS (QP) [13]. It is worth mentioning that our version of the QP protocol is slightly modified, and to be more exact, it is a combination of the QPASS and QCAST algorithms from [13]. Furthermore, the original QP algorithm in [13] uses a novel metric, EXT, when finding paths for requests in arbitrary networks. Due to the complete homogeneity of the grid topology, the EXT metric is equivalent to the hop distance, in terms of the preference it induces on different paths. Thus we have used the ordinary hop distance for our path selection, in all of the three algorithms. In all of the figures, NOPP and OPP correspond to the non-opportunistic and opportunistic approaches, respectively.

Opportunism is Better. We start by fixing $M = 5, N = 20, L = 30, k = 1$, and $p_{SWAP} = 1$, and compare the average total waiting time based on $p_{GEN}$. Fig. 5 shows the result of this comparison. The figure on the right shows the ratio of the improvement caused by opportunism to the non-opportunistic approach, which is obtained by dividing their difference by the non-opportunistic values. It is clear that opportunism has improved the performance of the network by almost 30%, and up to 45%. Furthermore, it can be seen that the QP algorithm provides better performance in this setup.

We now change the setup to another configuration, in which $M, N$ and $k$ stay the same, while we have $L = 6$ and $p_{SWAP} = 0.8$. This new setup has much more dynamism than the previous one. The results of comparing the average total waiting time of the three algorithms, based on $p_{GEN}$, is shown in Fig. 6. We can see that in a high dynamism of the quantum network’s state, the opportunistic approach enables significantly better performance, and this improvement is around 50%.

Now we increase the network size and analyze the effect of opportunism in bigger networks. Fig. 7 shows the average total waiting time based on the network size, for the three algorithms, in a grid network. The setup is $N = 20, L = 30, k = 1, p_{SWAP} = 0.8$, and $p_{GEN} = 0.8$. This setup is somehow an intermediate setup that is neither stable nor highly dynamic. We can see that just like small networks, opportunism significantly improves performance in bigger networks as well.

We finally present Table I, in which we have shown the effect of opportunism on the average link waiting time. The bottom row shows the improvement with respect to the non-opportunistic approach. Due to lack of space, we have only included the MG algorithm. Nonetheless, the improvement is similar in the other two algorithms. The results clearly demonstrate the fact that opportunism decreases the average link waiting time, thus increasing the efficiency of resource consumption. The setup in this part is $N = 20, M = 10, L = 30, p_{SWAP} = 1$, and $k = 1$.

Degrees of Opportunism. In this part, we demonstrate

### Table I: Average link waiting time in the MG algorithm

| $p_{GEN}$ | 0.1 | 0.3 | 0.5 | 0.7 | 0.9 |
|-----------|-----|-----|-----|-----|-----|
| NOPP      | 12.9| 8.8 | 7.18| 6.3 | 5.3 |
| OPP       | 11.5| 6.15| 4.16| 3.6 | 3.13|
| Improv.   | 11% | 30% | 41% | 43% | 41% |

Fig. 7: Average total waiting time in a grid network, with the fixed scenario $N = 20, L = 30, k = 1, p_{SWAP} = 0.8$, and $p_{GEN} = 0.8$. The figure on the right shows the ratio of the improvement caused by opportunism to the non-opportunistic approach.
the effect of changing the degree of opportunism. The results verify our analysis, and as we can see in Fig. 8, opportunism has a spectrum of choices for algorithm design. Due to space limitations, we have only included the MG protocol here. In Fig. 8, the number after OPP is the opportunism degree.

VII. RELATED WORK

Opportunism, as studied in this paper, is related to several lines of work in quantum networks. In terms of fundamental analysis, there are a series of works analyzing the limits of quantum communication, without examining the effect of opportunism. Below are several examples. Pirandola et al. [30] analyze the fundamental limits of repeaterless communications using local operations and classical communication. Bäuml et al. [31] study fundamental limitations on quantum broadcast networks by using multipartite entanglement instead of bipartite entanglement. Patil et al. [21] show that by leveraging n-qubit GHZ projective measurements, entanglement can be generated between two endpoints at a rate independent of the distance between them. Khatri et al. [10] and Bernardes et al. [28] analyze the average total waiting time for a group of links with infinite lifetimes, and Coopmans et al. [32] provide analytical bounds on delivery times of long-distance entanglement in different protocols. Praxmeyer [33] analyzes the average total waiting time for a group of links with finite lifetimes.

Routing in quantum networks has gained significant attention in the past few years. For example, Schoute et al. [27] analyze virtual links (i.e., entanglement between non-adjacent nodes) as primary resources for entanglement routing, in ring and sphere topologies. Chakraborty et al. [9] modify the greedy routing algorithm with regards to quantum networks, while also analyzing on-demand and continuous resource generation schemes. Pant et al. [14] introduce a multipath routing algorithm, utilizing local link information, in a grid topology. Van Meter et al. [25] present several link costs and use Dijkstra’s algorithm for calculating the cost of a path of links. Shi et al. [13] propose using recovery paths in routing, present a nonlinear path cost function, and introduce an extended version of Dijkstra’s algorithm for this function. Dai et al. [34] use linear programming to optimize the entanglement distribution rate between two endpoints, using repeaters with probabilistic swapping. However, none of these algorithms consider opportunism as we did in this paper.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we introduced opportunism in quantum networks and provided theoretical and experimental analysis with regard to its better performance when compared to the non-opportunistic approach. We demonstrated the two inherent mechanisms by which opportunism increases efficiency, namely the swapping and reservation gains. The results indicated that opportunism can decrease the average total waiting time up to 50% in different setups and network dynamism. Furthermore, we showed that opportunism is inherently flexible by introducing k-opportunism and demonstrating the spectrum of choices it provides. Due to the importance of efficient resource allocation in quantum networks, the results of this paper demonstrate the need for a consistent approach towards integrating opportunism in the management layer of such networks (e.g., by including opportunism as a key functionality in the network layer of a quantum networking stack).

Opportunism can be further analyzed in several directions. The effect of opportunism on the fundamental entanglement distribution rate between two endpoints of a repeater chain needs further study. Our analysis of the swapping and reservation gains does not include the finite lifetime and probabilistic swapping regime, and extending the results to include these assumptions would provide useful insight for designing new algorithms. Furthermore, our analysis can be extended to more graph-theoretic results (including different topologies). Finally, our simulator is currently supporting only grid networks and does not include the finite memory assumption. Extending the simulator to cover these areas is also an interesting future work.

ACKNOWLEDGEMENT

This work was supported by Natural Sciences and Engineering Research Council of Canada (NSERC) CREATE 528125-2019 and NSERC Discovery Grant RGPIN-2016-05310.

REFERENCES

[1] C. H. Bennett and G. Brassard, “Quantum cryptography: Public key distribution and coin tossing,” Theor. Comput. Sci., vol. 560, pp. 7–11, 2014.
[2] P. W. Shor, “Algorithms for quantum computation: discrete logarithms and factoring,” Proceedings of 35th annual symposium on foundations of computer science, pp. 124–134, 1994.
[3] M. Fitzi, N. Gisin, and U. Maurer, “Quantum solution to the Byzantine agreement problem,” Phys. Rev. Lett., vol. 87, p. 217901, 2001.
[4] M. Ben-Or and A. Hassidim, “Fast quantum Byzantine agreement,” Proceedings of the 37th Annual ACM Symposium on Theory of Computing, pp. 481–485, 2005.
