1. Introduction

Navigation through the World Wide Web (WWW) has nowdays become an indispensable way to obtain information in everyday life. A dramatically growing number of webpages that contain various information has revealed the urgent demand of an effective tool to sort and rank the webpages for effective information searching. The PageRank raised by Google is one of the most representative and successful examples to accomplish such tasks [1,2].

The key concept for Google PageRank is to treat the entire internet as a directed graph, where each website can be regarded a node of the graph, and each hyperlink that directs website navigators from one website to another is treated as an edge. After long-time stochastic navigation in the website network which is essentially a classical random walk process, there would eventually be a stable probability distribution. By sorting the websites according to their probability values, one would get the ranking of significance for these websites. PageRank has also been widely applied to a larger diversity of networks. For instance, evaluating the impact of a scientist through his connections in the academic network [3], studying species within an ecosystem [4], and finding key neurons in the neural network of the worm C. elegans [5] can all be reduced to the model of element ranking that PageRank manages to do.

In recent years, quantum physicists have introduced a quantum protocol of PageRank and mainly focused on the implementation on a classical computer with consideration of quantum mechanics [6,7]. Quantum PageRank has demonstrated a few advantages over classical PageRank [7,8], for instance, to generate more accurate ranking by reducing degeneracy from elements of the same probability, and to have better notification of the significance of secondary-hubs in the network, etc.

While Szegedy quantum walk was initially proposed for quantum PageRank [6,7], a currently more popular proposal uses quantum stochastic walks [8] to replace classical random walks in Google PageRank. Quantum stochastic walk [9,10] is a flexible blending of classical random walk and quantum walk to have more complex matrix element transition options (see Fig. 1a). It yields stationary results that can be considered for the ranking, instead of the other models that cannot.
of averaging the dynamic evolution results in time as the Szegedy quantum walk approach works [6,7]. Apart from PageRank, quantum stochastic walk has wide applications such as analyzing the energy transport in biological open quantum systems [11], simulating associative memory of Hopfield neural networks [12], modelling learning agents [13] or decision making [14], making Haar random unitaries [15], and modelling quantum state discrimination [16], etc. So far, the proof-of-principle demonstrations of quantum stochastic walks have been reported for various hardware platforms [17–21], such as integrated photonic lattice [17,18] and programmable Mach-Zehnder interferometer array [19] (see details in the Supplementary materials Note 1). Given the high cost, small size and limited flexibility of these physical quantum systems, numerical calculation of quantum stochastic walks have been reported for various hardware platforms [17–21], such as integrated photonic lattice [17,18] and programmable Mach-Zehnder interferometer array [19] (see details in the Supplementary materials Note 1). Given the high cost, small size and limited flexibility of these physical quantum systems, numerical calculation of quantum stochastic walks is still highly demanded, especially for quantum PageRank that normally tackles large-scale networks.

However, numerical calculations of large-scale quantum stochastic walks will not be easy inherently. Quantum stochastic walk is based on a Lindblad master equation [22] and involves lots of Kronecker products in its form. For a network with $N$ nodes, its Hamiltonian and Lindblad matrix are both in $N \times N$ dimension, and then the numerical calculation for Kronecker products would involve a dimension of $N^4$. Therefore, a network of 100–150 nodes can already cause an explosion of required memory and exhaust a laptop, not to mention the far more intricate networks in real life. There have currently been a number of solvers for quantum stochastic walks [23–26], including Qutip [23,24], a Python package, QSwalk, a Mathematica package that’s specifically tailored for tasks on directed graphs [25], and QSwalk.jl, an adapted package using Julia [26]. However, these packages have not employed enough optimization methods to level up the numerical capability for large networks. See a brief review of relevant software solutions in the Supplementary materials Note 1.

Additionally, there have been alternative attempts to use the sampling method based on the quantum stochastic Schrödinger equation [8] to approximate quantum stochastic walks. These involve a memory of $O(N^2)$ instead of $O(N^4)$, and a time cost that is proportional to the sampling rate $m$. For networks with a large node number, these sampling results would be inaccurate since the probability at each node is too small to be distinguished, and the sampling error would have a strong influence. Increasing the sampling rate $m$ may improve the accuracy, but that increases the computing time as well. Therefore, the limitations in hardware, available solvers and approximating methods strongly call for an efficient numerical solver for large-scale quantum stochastic walks.

Fortunately, there are ways to make a difference. Firstly, the Runge-Kutta method has long been used to solve ordinary differential equations [27–31]. It essentially reduces the matrix dimension to improve the calculation, which can also be applied to solve the quantum stochastic walks with a much smaller memory requirement. Besides the numerical method, the emerging TensorFlow framework and GPU parallel computing can also be utilized. TensorFlow is an open-source software library developed by Google for dataflow programming [32]. TensorFlow is useful for operating machine learning applications such as neural networks on multidimensional data arrays (tensors), and now it is also used to solve physics problems [33–37]. TensorFlow can be run on not only...
CPU, but also on GPU, the graphics processor unit, and operates GPU parallel computing. GPU computing in the last decade has been developed rapidly. Through efforts by companies like NVIDIA, GPUs improve much faster than CPUs, reaching a large capacity of billions of transistors, and in the meantime, the work mechanism of GPUs to quickly create, run and retire multi-threads makes parallelism an inherent advantage for GPUs. GPU parallel computing has benefited many general purpose scientific computational problems by bringing up significantly speed-up performances [38–40].

Therefore, in this article, we present an efficient solver for large-scale quantum PageRank using the Runge-Kutta method to reduce the matrix dimension to \( O(N^2) \) and employing TensorFlow to conduct GPU parallel computing. We demonstrate its performance by solving quantum stochastic walks on Erdős–Rényi graphs using an RTX 2060 GPU. The test on a graph of 6000 nodes requires a memory of 5.5 GB and time of 223 s, and that on the graph of 10000 nodes requires 226 MB and 3.6 s. Compared with QSWalk, a currently prevalent Mathematica solver, our solver reduces the required memory and time to only 0.2% and 0.05% for the same graph of 1000 nodes. We apply the solver to quantum PageRank for the USA major airline network with up to 922 nodes. We further demonstrate quantum fast hitting as a case for solving general quantum stochastic walks with flexible Hamiltonian and Lindblad matrix inputs. This efficient solver for large-scale quantum PageRank and quantum stochastic walks would greatly facilitate the study of quantum information in real-life applications.

### 2. Method

As has been mentioned, Google PageRank uses a very straightforward model with classical random walk and quantum PageRank essentially replaces it with quantum stochastic walk [8]. The detailed model for both PageRank protocols have been explained in the Supplementary materials Note 2. The task for improving this solver can boil down to the problem of solving the quantum stochastic walk that is normally expressed in the Lindblad master equation:

\[
\frac{d\rho}{dt} = -(1 - \omega)\mathcal{I}\mathcal{H}\rho(t) + \omega \sum_{k=1}^{N^2}(L_k\rho(t)L_k^\dagger - \frac{1}{2}L_kL_k^\dagger\rho(t) + \rho(t)L_k^\daggerL_k),
\]

where \( \rho \) is the density matrix that needs to be solved since it works out the element ranking. The parts with Hamiltonian \( \mathcal{H} \) and Lindblad terms \( L \) describe the quantum walks and classical random walk, respectively.

For this equation, we can rewrite it in such a form: \( \frac{d\rho}{dt} = \mathcal{L}\cdot\rho(t) \), where \( \mathcal{L} \) is the transpose of the matrix \( \rho \). Then \( \rho \) can be solved by matrix exponential method: \( \rho(t) = e^{\mathcal{L}t} \cdot \rho(0) \). The expression for \( \mathcal{L} \) reads as follows:

\[
\mathcal{L} = -(1 - \omega)(\mathcal{I}\mathcal{H} + \mathcal{H}^T \mathcal{I}K) + \omega \sum_{k=1}^{N^2}(L_k^\dagger L_k - \frac{1}{2}(L_k^\dagger L_k^\daggerL_k + L_kL_k^\dagger \mathcal{I})�K),
\]

where \( \otimes \) is the Kronecker product.

This suggests, for a network with \( N \) elements, the Hamiltonian \( \mathcal{H} \) and Lindblad matrix \( L \) are all always of a size \( N \times N \), and then the Kronecker product would result in a size of \( N^2 \times N^2 \), as illustrated in Fig. 1b. If we calculate Eq. (2) directly, the memory we will need is approximately \( 2N^4 \), making it difficult to calculate a network with over 150 nodes in a common laptop with a memory of 8 GB.

However, we notice that for a Kronecker product \( C_{x,y} = A_{x,x} \otimes B_{y,y} \), each element in \( C \) can be calculated separately as follows (with an example shown in Fig. 1b):

\[
C_{ij} = A_{i,j}B_{i,j} \mod n \times B_{i,j} \mod n \times B_{i,j} \mod n.
\]

where \( \mod n \) means floor, and \( (i - 1)\mod n \) gets the largest integer value less than or equal to \( (i - 1)/n \). mod means modulo, and \( (i - 1)\mod n \) gets the remainder after dividing \( i - 1 \) by \( n \).

Therefore, although \( \mathcal{L} \) is a matrix consisting \( N^2 \) elements, they each can be obtained from information of \( O(N^2) \). When solving the equation \( \frac{d\rho}{dt} = \mathcal{L}\cdot\rho(t) \), we make use of the Runge-Kutta method, which does not require the storage of the whole matrix \( \mathcal{L} \). By using only one row of \( \mathcal{L} \) each time, we reduce the memory requirement down to \( O(N^2) \). We use the Runge-Kutta method with adaptive step length, termed as RK45 method [30]. Details for Runge-Kutta method are given in the Supplementary materials Note 3.

In the meantime, as the calculation for each element of a matrix is independent, the calculation sequence for elements has no influence on the numerical results. Such a feature is exactly suitable for parallel computing, mapping the calculation for each element into one processing unit and calculating a large number of elements simultaneously. The GPU parallel computing is conducted using TensorFlow and we also run CPU computing in TensorFlow as a comparison. The framework of our solver of using Runge-Kutta numerical method and TensorFlow GPU parallel computing is illustrated in Fig. 1c.

### 3. Performance

In this part, we further investigate the performance of this quantum stochastic walk solver in terms of memory and time consumption.

As mentioned before, our method uses \( O(N^2) \) memory. We state that it is the least memory consumption a general-purpose quantum stochastic walk routine may achieve, since the storage of the Hamiltonian requires the memory of \( N^2 \) elements. We have also employed some new engineering techniques from TensorFlow 2, e.g., the fine-grained constant folding and TensorFlow XLA, to further reduce memory overhead of our algorithm.

In practice, for a network with 1000 nodes, the built-in MatrixExp routine in the Mathematica solver QSWalk [25] will take over 100 GB of main memory, which is not feasible even for a high-end PC nowadays. An average laptop with 8 GB RAM is merely capable for a graph with 200 nodes. In contrast, as shown in Fig. 2a, our approach requires around 200 MB for \( N = 1000 \).

It is a bit more complex when it comes to the time consumption. Like general calculations for matrix production [41], each step of the Runge-Kutta iteration requires \( O(N^2) \) work. However, the actual amount of steps required depends on the \( H \) and \( L \) matrices received, and the error bound \( \epsilon \) requested. The analysis is quite complicated, so we provide general analysis on the parameters in the Supplementary materials Note 3a.

In our implementation, the worst case time complexity is \( O(N^3/e^{2\epsilon}) \). It is reached when the graph is complete, which is seldom the case in real-world graphs. A graph of Erdős–Rényi type with \( O(N\log N) \) edges requires \( O(N^2\log N/e^{2\epsilon}) \) work. Also, it is worth noting that on the special case of quantum walk (\( \omega = 0 \)), the time complexity is reduced to \( O((EN)^2/e^{2\epsilon}) \), where \( E \) is the exact number of non-zero entries in the Hamiltonian matrix. This makes our method also competitive when simulating quantum walks. See explanation on computational time cost for different cases in the Supplementary materials Note 3b.

On the other hand, each step of our method is defined by a combination of vector operations, which is suitable for parallel
computing. As is shown in Fig. 2b, time consumption of our algorithm on an RTX 2060 GPU is of an order between $N^{1.3}$ and $N^2$, and is far less than the proven asymptotic complexity order. It is mainly due to the growth of parallelism on around 3000 GPU cores as the problem scales.

To verify our analysis involving the special case of $\omega = 0$, we carried out simulations on a graph of size 200 on an i7-4700MQ CPU. The results are 39.6 and 1.4 s respectively for $\omega = 0.8$ and $\omega = 0$. For the general quantum stochastic walks on Erdős-Rényi graphs, the test using an RTX 2060 GPU for a graph of 6000 nodes requires a memory of 5.5 GB and time of 223 s, and for a graph of 1000 nodes, it requires 226 MB and 3.6 s (Fig. 2). Compared with QSWalk, a currently prevalent Mathematica solver, our solver for the same graph of 1000 nodes reduces the required memory and time to only 0.2% and 0.05%.

For a brief power analysis, we compare the required power for GPU and CPU for the same task, which essentially reduces to the task general matrix multiplication. According to an NVIDIA report, the energy efficiency (images/s/Watt) for GPU (Tegra X1 FP16) is 10 times higher than CPU (Intel Core i7 6700 K). Considering the huge size operation in our solver, the power saved by GPU can be impressive. Additionally, comparing to the HPC system, the GPU implementation is much more accessible. The household GPU setups nowadays help to ensure the practicability of this solver.

4. Case study for quantum PageRank

Having constructed the solver, we would demonstrate it for quantum PageRank on a large-scale network in real-life: the highly developed USA airlines among 922 main USA airports, i.e., a network of 922 nodes. The data of airline and airport information is retrieved from the USA Department of Transportation. As shown in Fig. 3, we plot some airports and airlines in the USA map according to their real longitude and latitude information. For visual effects, we just plot 80 airports that have largest number of airlines and their airlines, instead of all 922 airports and over 14,000 airlines that would otherwise make the figure a mess. Even Fig. 3 just shows partial network, we can still see the network covers all states across the USA. Therefore, fights are important modes of transport in USA, and information of significance ranking for these airports would be useful for passengers.

We use the solver to import the raw data that shows all airlines departing from an airport and arriving at another airport as the connection profile. Note that for many other networks, for instance, the hyperlinks for a company website, the connection profile cannot be directly downloaded in a spreadsheet. Instead, we can use the web scraper code to get the information, which we provide separately in the Supplementary materials Data I to facilitate users. Now the solver generates the Hamiltonian and Lindblad matrix from the connection profile. The solver then employs the Runge-Kutta RKF45 method [30] and loads TensorFlow GPU parallel computing. The outcome is the probability distribution at all elements. The ranking of these probability suggests the ranking of significance of these elements, in this case study, the ranking for all 922 USA airports. The whole code for this solver and some sample routines are given in the Supplementary materials Data II and III, respectively.

We can now analyze the obtained quantum PageRank results. The solver gives the top-10 airports, as listed in Fig. 3. It shows that these airports indeed serve as the hubs to link to many airports through airlines. The result has also well spotted the “secondary-hubs”, those in the tier-two ranks with still many connections in certain regions. A quantitative definition of the “secondary-hub” has been given [6]: a node with its probability larger than 1/N and smaller than c/N, where c is a constant set to be 10, and a “main-hub” is a node with a probability larger than c/N. In the case of USA airport ranking, for comparative studies, we conduct classical PageRank using Gephi, an open-source network analysis software. As shown in inset of Fig. 4, classical PageRank does not spot the main hubs and regards up to 217 airports as the secondary hubs, which seems not very realistic. On the other hand, our quantum PageRank solver gives a better classification of the main hubs and secondary hubs: 11 nodes are above c/N and 66 nodes between 1/N and c/N. A reason for the sharper identification of secondary hubs by quantum PageRank can be found from the obtained probability distribution by the two approaches (Fig. 4).
Classical PageRank has more averaged distribution over all nodes, and does not differentiate nodes of higher ranking with a probability separation as clear as the quantum PageRank.

The case of USA airline network is highly representative. For any other networks that users want to rank, they only need to import the connection profile of that network, which records a column of the starting node id and a column of the destination node id, so that the two ids in a row represent a connected edge. Even if they do not know the connection profile for some website networks, they can use our web scraper code to easily get it. Then our solver conveniently reads the connection profile to give the rank results.

5. Case study for a general quantum stochastic walk

Generally, apart from quantum PageRank, a large variety of fields can be modeled by open quantum systems [42] using quantum stochastic walks. We provide a slightly adapted solver that takes user-defined Hamiltonian and Lindblad matrices as inputs instead of the node connection profile, so that it can work for various quantum stochastic walks. The whole code for this solver and some sample routines are given in the Supplementary materials Data II and III, respectively.

Here we use this adapted code to run an example: to test the robustness of quantum fast hitting against the environment decoherence. Fast hitting, a task that a particle enters the entry site and reaches the exit site of the glued tree, has been suggested as a typical task to show quantum advantages [43], and has been recently experimentally demonstrated in integrated photonic lattice [44,45]. It is of interest to know how the advantageous hitting efficiency would persist when decoherence is introduced [9], which can be quantitatively described by the weighting parameter $\omega$ given in Eq. (1).

However, the number of nodes in such glued tree structure increases exponentially with the tree depth. For one with a branch rate of 3 and tree depth of 6, it has a total number of 2186 nodes (Fig. 5a). While there are different one-dimensional equivalences for pure quantum and pure classical hitting respectively [45], the mixed hitting has to use the complete two-dimensional graph. Furthermore, one needs to calculate the hitting scenarios for different evolution lengths to get the evolution dynamics, which requires a usual solver to repeat calculations many times, each time for one evolution length at a certain $\omega$.

Using our solver, we get the evolution dynamics for different values of $\omega$. Hitting efficiency, the calculated probability at the exit node, is found to be sensitive to the decoherence (Fig. 5b). What’s remarkable in this case is that, we only calculate once for each $\omega$ and the solver outputs dynamics at all evolution lengths up to the set maximal length. This is due to the mechanics of Runge-Kutta method that iterates in time, so that the intermediate states can always give the full probability distribution for various evolution lengths. This is very advantageous for studying the evolution dynamics of quantum stochastic walks.

Many reported works on quantum stochastic walks can also be extended to studies on a larger-scale network. For instance, the simulation of “associative memory” of Hopfield networks using quantum stochastic walks [12] can be applied to pattern recognition with a color cube of $256 \times 256 \times 256$ nodes; the simulation for energy transport in M13 virus [11] can be improved by
increasing the donor-to-acceptor ratio from 100:1 all the way to 1000:1, using 1001 nodes for quantum stochastic walks.

6. Discussion and conclusion

In this work, we have demonstrated a highly efficient quantum PageRank solver that can easily conduct the ranking for large-scale networks on a normal computer within several minutes, which previously takes a hyper-performance workstation several hours or days. This is of great meaning. Without overcoming this hurdle, we will not be able to apply quantum PageRank to practical ranking problems for real-life networks such as Internet, bacteria groups, and potentially the future networks in 5G or the Internet of things (IoT) [46], which always comprises a large number of nodes. Note that the classical PageRank only has a complexity of $O(N^2)$, our effort of reducing the complexity of quantum PageRank from $O(N^4)$ to $O(N^2)$ would substantially boost the comparative study on quantum and classical PageRank algorithms for various applications. Meanwhile, the solver can also benefit studies on rich scenarios of large-scale quantum stochastic walks.

Furthermore, we anticipate that Runge-Kutta method will continue to serve an important role in reducing memory cost when solving quantum physics problems via classical computers, since matrix calculation has always been a main task in quantum mechanics. Besides, the method of using TensorFlow GPU parallel computing to reduce the time is a powerful approach with strongly growing popularity. It is highly suggested to combine this emerging powerful tool with advanced numerical methods to together contribute to various complicated quantum tasks [47]. Such multidisciplinary research would always be helpful for the research of quantum information sciences.

Conflict of interest

The authors declare that they have no conflict of interest.

Acknowledgments

The authors thank Jian-Wei Pan for helpful discussions. This work was supported by the National Key R&D Program of China (2019YFA0308700, and 2017YFA0303700), the National Natural Science Foundation of China (61734005, 11761141014, 11690033), the Science and Technology Commission of Shanghai Municipality (STCSM) (17JC1400403), and the Shanghai Municipal Education Commission (SMEC) (2019SHZDZX01, 2017-07-00-02-E00049). Hao Tang is supported by the National Natural Science Foundation of China (11904229), and China Postdoctoral Science Foundation (2019T120334). Tian-Yu Wang acknowledges support from Thiyuan Innovative Research Center of Shanghai Jiao Tong University. Xian-Min Jin acknowledges additional support from a Shanghai Talent Program.

Author contributions

Xian-Min Jin conceived and supervised the project. Hao Tang and Ruoxi Shi designed the scheme of numerical method and parallel computing for quantum stochastic walks. Ruoxi Shi wrote the TensorFlow code, did comparative studies on related software solutions and conducted the complexity analysis for Runge-Kutta method. Tian-Shen He conducted the conventional theoretical model of quantum stochastic walks and the case study of quantum fast hitting. Hao Tang reviewed the hardware implementations. Yan-Yan Zhu, Marcus Lee, and Tian-Yu Wang analyzed the case study of quantum PageRank. Hao Tang, Tian-Shen He and Tian-Yu Wang processed the figures. Hao Tang wrote the paper, with input from all the other authors.

Appendix A. Supplementary materials

Supplementary materials to this article can be found online at https://doi.org/10.1016/j.scib.2020.09.009.

References

[1] Brin S, Page L. The anatomy of a large-scale hypertextual web search engine. Comput Netw ISDN Syst 1998:33:107–17.
[2] Page L, Brin S, Motwani R, et al. The PageRank citation ranking: bringing order to the web [Technical Report]. Stanford InfoLab; 1999.
[3] Radicchi F, Fortunato S, Markines B, et al. Diffusion of scientific credits and the ranking of scientists. Phys Rev E 2009;80:056103.
[4] Allesina S, Pascual M. Googleing food webs: can an eigenvector measure species importance for coextinctions? PLoS Comput Biol 2009;5:e1000494.
[5] Watts DJ, Strogatz SH. Collective dynamics of ‘small-world’ networks. Nature 1998;393:440–2.
[6] Paparo GD, Martin-Delgado MA. Google in a quantum network. Sci Rep 2012;2:444.
[7] Paparo GD, Müller M, Comellas F, et al. Quantum Google in a complex network. Sci Rep 2013;3:2773.
[8] Sánchez-Burillo E, Duch J, Gómez-Gardeñes J, et al. Quantum navigation and ranking in complex networks. Sci Rep 2012;2:605.
[9] Whitfield JD, Rodríguez-Rosario CA, Aspuru-Guzik A. Quantum stochastic walks: a generalization of classical random walks and quantum walks. Phys Rev A 2010;81:022323.
[10] Bianmonte J, Faccin M, De Domenico M. Complex networks from classical to quantum. Commun Phys 2019;2:53.
[11] Park H, Heldman N, Rebentrost P, et al. Enhanced energy transport in genetically engineered excitonic networks. Nat Mater 2015;15:211–6.
Hao Tang received a B.S. degree in Electronic Science and Technology from Huazhong University of Science and Technology in 2010, and a Ph.D. degree in Optoelectronics from Sheffield University (UK) in 2014. She also took an executive program on quantitative finance at London Business School in 2013. She worked as a postdoctoral researcher at Shanghai Jiao Tong University during 2015–2019 and turned to an assistant professor in 2020. Her research interest includes quantum computing and quantum simulation on integrated photonics, and application of quantum computing on machine learning, finance and optimization tasks.

Ruoxi Shi is currently an undergraduate student at Shanghai Jiao Tong University. He has been working with Prof. Xian-Min Jin and Dr. Hao Tang at the Center for Integrated Quantum Information Technologies (IQIT), Shanghai Jiao Tong University since he was in Shanghai High School. His current research interest is quantum computation and machine learning.

Tian-Shen He is currently an undergraduate student at Fudan University. He has been working with Prof. Xian-Min Jin and Dr. Hao Tang at the Center for Integrated Quantum Information Technologies (IQIT), Shanghai Jiao Tong University since he was in Shanghai High School. His current research interest is quantum computation, quantum information and quantum simulation for biology and chemistry.

Xian-Min Jin is currently a professor at the School of Physics and Astronomy, Shanghai Jiao Tong University. He received his Ph.D. degree at University of Science and Technology of China in 2008. After two-year postdoctoral research in Hefei National Laboratory for Physical Sciences at the Microscope, he joined the Clarendon Laboratory and University of Oxford as a research associate, and became Marie Curie Fellow in 2012. He was promoted as a full professor and the director of the Center for Integrated Quantum Information Technologies (IQIT) in Shanghai Jiao Tong University in 2019. His current research interest is quantum computation, quantum information and quantum simulation for biology and chemistry.