Compilation of Fault-Tolerant Quantum Heuristics for Combinatorial Optimization

Yuval Sanders[1], Dominic Berry[1], Pedro Costa[1], Louis Tessler[1], Nathan Weibe[2], Craig Gidney[3], Hartmut Neven[3], and Ryan Babbush[3]

[1] *Department of Physics and Astronomy, Macquarie University, Sydney, NSW 2109, Australia.*
[2] *Department of Computer Science, University of Toronto, Toronto, ON M5S, Canada.*
[3] *Google, Los Angeles, CA 90291, United States.*

We compile explicit circuits and evaluate the computational cost for heuristic-based quantum algorithms for combinatorial optimization. We consider several variants of quantum-accelerated simulated annealing as well as adiabatic algorithms, quantum-enhanced population transfer, the quantum approximate optimization algorithm, and other approaches. We provide novel methods for executing the bottleneck subroutines for these heuristics, and our methods can easily be applied to other algorithms where numerical performance matters. We estimate how quickly the subroutines could be executed on a modestly sized superconducting-qubit-based quantum computer with surface code error correction. We conclude that quadratic speedups for heuristic-based quantum optimization algorithms are insufficient for early quantum computers to beat present day classical computers.

Much corporate interest in quantum technology is driven by the prospect of faster combinatorial optimization on a quantum computer. Combinatorial optimization problems are in general NP-hard, so it is not reasonable to expect quantum computers to provide efficient solutions in the worst case. Instead, the hope is that a quantum computer may deliver significantly better performance in the typical case, where performance is measured in terms of runtime or expected accuracy of an approximate solution. Ideas for such approches include the quantum approximate optimization algorithm (QAOA) [FGG14], the adiabatic algorithm [Far+00], and quantum-simulated annealing (QSA) [Som+08]. Previous work has focussed on high-level designs for which there is a likely asymptotic speedup over classical approaches, but this prior work does not provide explicit gate counts that allow for direct comparision between quantum and classical computers.

We compare the cost of implementing bottleneck subroutines for various quantum heuristics for combinatorial optimization. We consider four kinds of combinatorial optimization problem: the L-term spin model, Quadratic Unconstrainted Binary Optimization (QUBO), the Sherrington-Kirkpatrick (SK) model, and the Low Autocorrelation Binary Sequences (LABS) problem. In each case we compile several bottleneck primitives: amplitude amplification using the problem oracle, steps for QAOA, the Hamiltonian walk operator for the adiabatic algorithm, QSA via qubitized Metropolis-Hastings update [Lem+20], and QSA via adiabatic evolution of a spectral-gap-amplified Hamiltonian [BOS15]. We then estimate how many of these steps could be executed during a set period of time (either one hour or one day) on a surface-code quantum computer for SK and LABS; our results are summarised in Table 1.

Problem	Algorithm Primitive	Steps/Day	Physical Qubits	Toffoli Cost
SK	Amplitude Amplification	4.8×10^3	8.1×10^5	$2N^2 + N \quad +\mathcal{O}(\log N)$
	QAOA / 1st order Trotter	4.7×10^3	8.6×10^5	$2N^2 + 4N \quad +\mathcal{O}(1)$
	Hamiltonian Walk	3.3×10^5	8.0×10^5	$6N \quad +\mathcal{O}(\log^2 N)$
	QSA / Qubitized	3.3×10^5	8.4×10^5	$5N \quad +\mathcal{O}(\log N)$
	QSA / Gap Amplification	3.9×10^5	8.4×10^5	$5N \quad +\mathcal{O}(\log N)$
LABS	Amplitude Amplification	3.3×10^3	8.0×10^5	$5N^2/2 + 7N/2 \quad +\mathcal{O}(\log N)$
	QAOA / 1st order Trotter	3.4×10^3	8.4×10^5	$5N^2/2 \quad +\mathcal{O}(N)$
	Hamiltonian Walk	4.9×10^5	8.0×10^5	$4N \quad +\mathcal{O}(\log N)$
	QSA / Qubitized	1.7×10^3	8.8×10^5	$5N^2 \quad +\mathcal{O}(N)$
	QSA / Gap Amplification	1.7×10^3	8.8×10^5	$5N^2 \quad +\mathcal{O}(N)$

Table 1: A snapshot of our results for SK and LABS. The numerical values are based on a problem size of $N = 256$, a surface code cycle time of one microsecond, and an error rate of 10^{-3} per physical gate. Here N refers to the number of bits needed to specify a candidate solution for the combinatorial optimization problem. We assume all arithmetic is performed to constant precision; our technical manuscript [San+20] has results for variable precision arithmetic.

When compiling the bottleneck primitives, we make a large number of improvements to previous approaches. Underlying these improvements, we frequently find the need to manage the amount of arithmetic the quantum computer needs to do. As we focus on *heuristic* methods of optimisation, we expect that it will be preferable to sacrifice precision in coherent function evaluation on a quantum computer if it means we find a significant computational cost savings in the execution of a heuristic step. To take the most thoroughly investigated example [San+20, Fig. 5], we expect that we do not need high precision when calculating the probability of transitioning between candidate solutions during simulated annealing. We can improve performance by sacrificing precision.

Our most impactful contribution is therefore our technique for approximate function evaluation [San+20, Sec. II.E]. The idea is an old one. Instead of requiring the quantum computer to calculate, say, a high degree polynomial in order to approximate the exponential function, we provide the quantum computer with a precomputed lookup table stored as a QROM array [Bab+18]. The quantum computer would then approximate a function f by testing whether its argument, x, is between $k\delta$ and $(k+1)\delta$ for some predetermined grid spacing δ and an integer k. Depending on the value of k, the lookup table provides the data needed to calculate a linear approximation of f:

$$f(x) \approx f(k\delta) + m_k(x - k\delta). \tag{1}$$

Given k, the lookup table would provide the values for $f(k\delta)$ and m_k. Note that QROM is able to return an unboundedly large amount of data for no additional Toffoli cost because the readout is done with CNOTs. The computational cost of calculating $x - k\delta$ can be made zero by setting δ to be a power of two and encoding x in binary.

The main computational cost in terms of Toffolis comes from the multiplication, which can be performed to limited precision and is considerably simpler than the original function. The other cost comes from the QROM, which has a Toffoli cost corresponding to the number of points used in the interpolation. We are able to make that cost remarkably small by using variable spacing in the interpolation, with larger regions where the function is more slowly varying.

We also highlight our techniques for executing QSA, which demonstrate the application of a low-arithmetic method for performing state preparation based on inequality testing [San+19]. For example, we compile [San+20, pp. 37-38] the Szegedy walk suggested by Somma et al. [Som+08] using the inequality testing technique. In step 7, which is the key step, we show how to prepare the proper amplitudes (the square roots of the transition probabilities p) using an inequality test between the computed transition probabilities and a register prepared in an even superposition of possible values z. We save a great deal of arithmetic by testing the inequality $z^2 < p$ rather than $z < \sqrt{p}$ and thereby avoid the need to calculate $\sqrt{p}$.

Unfortunately, the Somma et al. walk operator requires us to calculate the probability for each possible transition, which dominates the cost of execution. A better idea is to use a quantum version of Metropolis-Hastings developed by Lemieux et al. [Lem+20], which requires only one coherent calculation of a transition probability per walk step. The method as proposed in Lemieux et al. [Lem+20] would be exponentially inefficient for the optimization problems we consider, because they have high connectivity. We are able to overcome that problem by using our lookup-based function approximation technique [San+20, Eq. (193)], thereby achieving a highly computationally efficient method for combinatorial optimization.

Having compiled these various approaches to heuristic-based combinatorial optimization on a quantum computer, we evaluate [San+20, p. 24] the computational cost of each primitive in terms of the number of non-Clifford gates (usually Toffoli gates) as well as the additional space cost. Finally, we estimate the number of times a surface code quantum computer could execute these algorithm primitives in a given unit of time (one hour or one day) given the physical error rate of the quantum computer (10^{-3} or 10^{-4} per physical gate), as well as the number of physical qubits needed. Under various assumptions about the workings of the surface code (described in [San+20, Sec. IV]), we expect that one Toffoli takes about 170 μs and 150,000 physical qubits [GF19]. We then calculate the runtime of our various algorithm primitives. Again, we present indicative results in Table 1.

Because of the significant overhead incurred by the surface code, we find discouraging estimates for heuristic-based combinatorial optimization on a million-qubit quantum computer. To take one classical benchmark [Isa+15], it is possible to execute 5×10^{11} steps of simulated annealing for an $N = 512$ SK instance on a classical computer in one hour whereas we estimate that the quantum computer could only execute 8×10^3 [San+20, p. 51] in the same time. The primary cause of this discrepancy is the large overheads incurred by performing error correction for physical error rates of 10^{-4} or worse. Without significantly lower error rates or much more efficient error correction, we are forced to conclude that the reported quadratic speedups for existing quantum algorithms are unlikely to allow even a million-qubit quantum computer to outperform today's classical computers at combinatorial optimization.

References

[Bab+18] Ryan Babbush et al. "Encoding Electronic Spectra in Quantum Circuits with Linear T Complexity". In: *Phys. Rev. X* 8 (Oct. 2018), p. 041015. DOI: 10.1103/PhysRevX.8.041015.

[BOS15] S. Boixo, G. Ortiz, and R. Somma. "Fast quantum methods for optimization". In: *The European Physical Journal Special Topics* 224 (2015), pp. 35–49. DOI: 10.1140/epjst/e2015-02341-5.

[Far+00] Edward Farhi et al. *Quantum Computation by Adiabatic Evolution.* Jan. 2000. arXiv: quant-ph/0001106.

[FGG14] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. *A Quantum Approximate Optimization Algorithm.* Nov. 2014. arXiv: 1411.4028 [quant-ph].

[GF19] Craig Gidney and Austin G. Fowler. "Efficient magic state factories with a catalyzed $|CCZ>$ to $2|T>$ transformation". In: *Quantum* 3 (Apr. 2019), p. 135. DOI: 10.22331/q-2019-04-30-135.

[Isa+15] S.V. Isakov et al. "Optimised simulated annealing for Ising spin glasses". In: *Computer Physics Communications* 192 (2015), pp. 265–271. DOI: 10.1016/j.cpc.2015.02.015.

[Lem+20] Jessica Lemieux et al. "Efficient Quantum Walk Circuits for Metropolis-Hastings Algorithm". In: *Quantum* 4 (June 2020), p. 287. DOI: 10.22331/q-2020-06-29-287.

[San+19] Yuval R. Sanders et al. "Black-Box Quantum State Preparation without Arithmetic". In: *Physical Review Letters* 122 (2 Jan. 2019), p. 020502. DOI: 10.1103/PhysRevLett.122.020502.

[San+20] Yuval R. Sanders et al. "Compilation of Fault-Tolerant Quantum Heuristics for Combinatorial Optimization". In: *PRX Quantum* 1 (Nov. 2020), p. 020312. DOI: 10.1103/PRXQuantum.1.020312.

[Som+08] R. D. Somma et al. "Quantum Simulations of Classical Annealing Processes". In: *Physical Review Letters* 101 (Sept. 2008), p. 130504. DOI: 10.1103/PhysRevLett.101.130504.