Quantum optimization and the path forward

Quantum computing could revolutionize numerous domains in business and science with optimization frequently identified as prime candidate to profit from such a revolution.

Optimization problem arises almost everywhere, so any improvement over the state-of-art classical algorithm with quantum computers could have huge impact.

Moreover these improvements could occur across multiple dimensions such as solution quality, solution diversity, time-to-solution and cost-to-solution.

It is often stated that quantum computers can evaluate all possible combinations of optimization problem simultaneously. This neglects that there remains an exponential list of possible solution and a quantum computer requires additional effort to find good solution.

For some problem quantum computers may offer a quadratic speedup eg: via Grover's Search but a quadratic speedup over an exponential run time is still an exponential run time.

Quantum Advantage and Complexity Theory

Complexity classes group computational problems together in terms of the amount of resources required to solve them. Naturally the harder the computational problem, the more resources it will require.

Complexity theory is a guiding compass to find classes of problems that are hard for a classical computer to solve, but perhaps less demanding on a quantum computer.

Understanding complexity theory is extremely useful for gauging possible quantum advantage in optimization, but it is not necessary, nor sufficient when seeking a practical quantum advantage.

Our notion of quantum advantage in practice should cast a wider net than formal proofs, but within a reasonable range guided by complexity-theoretic arguments.

Quantum Computational Complexity

Hardness is typically formalized in terms of the resources required by different models of computation to solve a given problem, such as the number of steps of a deterministic Turing machine.

Computational Complexity

$\{0,1\} = \Sigma$

A function $f: \Sigma^* \to \Sigma$ is said to be polynomial time computable if there exist a polynomial-time deterministic Turing machine that outputs $f(x)$ $\forall$ input $x \in \Sigma^*$

Polynomial bounded function : $f: \mathbb{N} \to \mathbb{N}$ if there exists a polynomial time deterministic Turing machine that outputs $1^{f(n)}$ on input 1^n $\forall n \in \mathbb{N}$. Such functions are upper bounded by some polynomial

Promise Problem : Decision problem for which the input is assumed to be drawn from some subset of all possible input strings.