Cody Jordan
COT6410

Understanding the "Oracle Separation of BQP and PH"

Raz and Tal's recent work on the "Oracle Separation of BQP and PH" is an important step towards establishing exactly where quantum computers lie in the realm of complexity classes (Raz 2018). Scott Aaronson[1] provides a good history of the efforts on this problem, and some context to the development of the field. In order to understand this result there are a number of important questions to answer: what are the role of oracle separations? how are bounded-error quantum polynomial time (BQO) and the polynomial hierarchy (PH) defined? what techniques have been employed in Raz and Tal's new proof? what are the implications for research in this area?

An oracle separation is a proof which uses the technique of "relativization" in order to establish results about complexity classes. Terry Tao[2] provides an intuitive explanation of what this means as a good introduction. In short, an oracle is a theoretical construct which can solve a particular problem in a single timestep. For instance this could answer if a particular SAT instance has a satisfiable assignment or not. Because many problems can be shown to be equivalent under appropriate reductions, such as SAT being NP-Complete[3], it often does not matter what particular problem the oracle can solve, but rather what complexity class that problem belongs to. An exception to this would be a case where the reduction is of a greater complexity than the original class. For instance DLOGTIME, the class of all problems solvable by a deterministic random-access Turing machine in logarithmic time, equipped with a SAT oracle would not be equivalent to a generic NP oracle because a polynomial time reduction is required to reduce SAT to other NP-Complete problems. In

[1] https://www.scottaaronson.com/blog/?p=3827
[2] https://terrytao.wordpress.com/2009/08/01/pnp-relativisation-and-multiple-choice-exams/
[3] https://complexityzoo.uwaterloo.ca/Complexity_Zoo:N#npc

most cases however this is how we can have an "NP oracle" which would be able to solve any NP problem.

Equipping a Turing machine, or some other fixed model of computation, with an oracle yields a new, possibly more powerful, machine with this extra operation. A Turing machine which has access to an NP oracle can trivially solve all NP problems, but would this help the machine solve problems which would take a regular Turing machine exponential time? In essence this forms a new complexity class, EXP^{NP}, all problems which can be solved in an exponential amount of time with a machine that has access to an NP oracle.

Relativization is a method of approaching difficult questions of complexity theory in a new way. P=NP appears to be very difficult, but perhaps there is a particular oracle A "relative to which" the problem may be more approachable. In essence this asks if $P^A=NP^A$, however, even if a proof could be found for the equivalence or nonequivalence of this new problem, what does this say about P=NP? Unfortunately in this case not much, which reveals some of the limitations of these relativized proofs. Baker et. al. have constructed an oracle A such that $P^A=NP^A$, and simultaneously an oracle B where $P^B\neq NP^B$ (Baker 1975). Meaning that a relativized approach will not be able to settle the question of P=NP.

Despite this relativization remains a useful tool. In the case of P=NP the result that the problem cannot be solved by relativized methods means that any proof technique which is known to relativize can immediately be ruled out without wasting a decade or so pursuing that angle. In addition these oracle results can be thought of as a reasonable problem in determining the query complexity of some proposed computational model (Aaranson 2009). Results such as these have preceded the discovery of Shor's algorithm, based on an earlier result on a black-box model for period finding, and are an important part of using Grover search as a bound on unstructured search problems. Query complexity can be a valid

question in its own right as well. Instead of looking at space as the constraining resource, such as with PSPACE, or time, as in P or EXP, one can treat the amount of queries to some long input as the basis for this complexity.

Having established what an oracle separation is the next question is to understand what BQP and PH are, something that the Complexity Zoo[4] has a brief overview of. Generally speaking BQP is the class of problems which a quantum can solve efficiently, while PH is a generalization of P, NP, and co-NP. As a concrete example QSAT is a natural problem in PH, where each alternating quantifier corresponds to an additional level up the hierarchy. Alternatively the second level can be viewed as languages which are decidable in nondeterministic polynomial time where the machine has access to an NP oracle, and the corresponding co-NP^{NP}. Each successive level up the chain is constructed from having an oracle for the previous level, and the union of all such levels is PH itself. Lastly, alternating Turing machines are a natural way of describing problems in PH. The main significance of choosing PH as the target of this proof is that it would mean that even in the strange case that P=NP, and the polynomial hierarchy collapses, then the oracle separation result on BQP would still be meaningful as a comparison between quantum and classical models of computation.

Despite all of this PH itself is mentioned very little in the proof, which instead focuses on AC^0, the set of all constant-depth polynomial-sized circuits with unlimited fan-in. More precisely it is only those circuits composed of NOT, AND, and OR gates, adding a gate capable of calculating PARITY describes a significantly different complexity class. The inability of AC^0 to calculate the parity of inputs, which requires an exponential number of gates in a constant-depth circuit, is an important result which lead to the current connection between

[4] https://complexityzoo.uwaterloo.ca/Complexity_Zoo:B#bqp
https://complexityzoo.uwaterloo.ca/Complexity_Zoo:P#phcc

AC^0 and PH. Established by Sipser in 1984 this result is one of the main motivations behind the formalization of AC^0 as a circuit class, because proving lower bounds on this very limited class of circuits is one of the few known proof techniques for addressing this manner of problem (Sipser 1984).

Intuitively this connection can be seen by imagining trying to solve some QSAT instance. The first quantifier states that "there exists some x_1, x_2,...x_n such that," which is expressed as a single OR gate at the top of a large tree. The output of this gate corresponds to the solution of the decision problem, and has children for each of the 2^n possible values of all variables which are qualified by that qualifier. Feeding in to each of these children is an AND gate, corresponding to a quantifier which ensures that "for all" such variables the predicate must be true. These continue down for a constant depth, corresponding to the number of quantifiers, and end with input variables which are the values of the predicate evaluated for a particular setting of its variables. Unfortunately a minor typo in the paper confused this slightly, something fortunately rectified in the original work by Sipser and an excellent talk by Tal himself in an excellent talk on his work[5]. AND gates correspond to universal quantifiers, and OR gates to existential quantifiers!

Based on this intuitive picture a more formal relationship between AC^0 and PH can be established, which in turn is used by Tal to convert his bounds on AC^0 circuits to a statement about the relativized version of PH. Understanding the quantum half requires understanding some of what makes quantum computation work. BQP is also typically defined in terms of circuits, due to an easy correspondence with the fact that quantum processes in physics can be described by unitary operations. These naturally can be broken down to unitary matrices operating on 1 or 2 qubits to make a universal family of gates. BQP

[5] https://www.youtube.com/watch?v=LUGr7mO-tM0

limits this to polynomial time uniform circuits for which the probability of returning a wrong answer is bounded to some constant below ½. The constant chosen is arbitrary because the process can be repeated a polynomial number of times, and a majority vote of taken, in order to reduce the probability of error to some exponentially small value. Uniformity as well is an important concept, which generally means that these circuits are not allowed to be wildly different for different input lengths. Were one to allow this the resulting complexity class would be one which allowed a given number of "advice bits," such as BQP/poly, something explored in an entertaining way in *Quantum Computing Since Democritus*[6], and in a more formal way by Aaranson (Aaranson 2010). The most popular example is Shor's algorithm, but the exact place which BQP occupies is still open.

Understanding these definitions we can talk about exactly what it is that Raz and Tal have done, with the help of a good introduction by Barak[7]. At a high level their proof constructs a problem requiring a classical computer and a quantum computer to distinguish between two probability distributions by sampling from them. From this black box model they connect their result to a general oracle separation, and also establish the relationship between the bounds they prove on the classical computation and PH.

Their first step is to establish an explicit distribution which fools AC^0, in the sense that it appears pseudorandom, but against which BQP machines still have high advantage. On a simple level random numbers are chosen from a normal distribution with a low standard deviation. A second set of numbers is computed by applying the Hadamard transformation to these random numbers, concatenated with the original list, and then both are probabilistically rounded to either -1 or 1. This is equivalent to drawing from a multivariate gaussian distribution with a covariance matrix with Hadamard matrices as

[6] https://dl.acm.org/citation.cfm?id=2487754
[7] https://windowsontheory.org/2018/06/17/on-the-raz-tal-oracle-separation-of-bqp-and-ph/

sub-matrices, in essence a distribution which largely appears random but has subtle correlations.

Establishing the existence of an efficient quantum algorithm which can detect these correlations is straightforward, as most of the work was performed earlier in the paper which introduced the idea of the correlation problem (Aaranson 2018). This algorithm operates by performing a single query to the oracle, receiving a superposition of the distribution, applying a Hadamard to the latter half, and then measuring to search for correlations. This relies on the fact that the Hadamard transformation is very easy for a quantum computer to compute, and that it does not need access to the actual values of the distribution, but rather only needs to search for correlations. Because it runs in logarithmic time this can be easily amplified to provide a near-certain result on the decision problem.

The more challenging result in the paper is establishing a lower bound on the advantage which an AC^0 circuit can have for distinguishing the two distributions. At a high level this relies on Tal's earlier results about the bounds on higher-order terms in the multilinear extension of boolean functions. Establishing these particular properties about polynomials is the leverage that they need in order to prove the bound on their capabilities.

Understanding this requires a few conceptual leaps to keep track of. As before the class PH is related to the class of AC^0 circuits, in turn these circuits are treated as total computable boolean functions of a particular form, mapping $\{-1, 1\}^n$ to $\{-1, 1\}$. In particular these can be extended to unique multilinear functions on the real numbers, corresponding to some polynomial. The coefficient of each term in this polynomial is called the "Fourier coefficient," and there are a number of good resources[8] which provide an introduction to these concepts, including one by Raz himself[9] and a survey paper by de Wolf (de Wolf 2008).

[8] http://www.contrib.andrew.cmu.edu/~ryanod/?p=253#propfourier-coeff-formula
[9] https://www.youtube.com/watch?v=FmisseepEys

The specific relied upon is that the functions representing AC^0 circuits have strict bounds on the coefficients of their higher order terms (Tal 2014). Using these techniques the paper is then able to establish that the expected value for these functions is not significantly different on values chosen from a random uniform distribution, and from the pathological distribution they designed.

In order to bring this together we can walk through a high-level description of the path to these results. First they establish an explicit distribution designed to fool classical computers with near-certainty. This is shown by treating these computers as a particular subset of multilinear functions, one with bounded L1 norms on coefficients, and showing that the expectation values do not significantly differ. As a result of these functions describing the capabilities of AC^0, and the correspondence between AC^0 and PH, this establishes that this problem is difficult for PH machines. Simultaneously quantum computers do not experience the same difficulty, and they reference an explicit algorithm which puts the problem in to BQP. Combining these they conclude from this black-box model that there exists an oracle relative to which BQP is not contained within PH.

Open Questions

Reading through this there are a number of open questions which seem to present themselves.

- ❏ Is there an oracle relative to which P=NP but P≠BQP?
- ❏ AC^0 circuits must be total computable, while a quantum algorithm may be "ambivalent" on many inputs, is this fundamental difference important to this separation?
- ❏ Tal established an L1 bound on fourier coefficients, an important detail because efficient quantum algorithms are also described by low-degree polynomials, do similar bounds exist on other related circuit families?
- ❏ Is there a non-relativizing separation proof? Although this is a very large question
- ❏ Is BPP = P? Would derandomization results have any impact on this work?

References

Aaronson, Scott, and Andrew Drucker. "A Full Characterization of Quantum Advice." Proceedings of the 42nd ACM Symposium on Theory of Computing - STOC 10, 2010. doi:10.1145/1806689.1806710.

Aaronson, Scott. "BQP and the Polynomial Hierarchy." Proceedings of the 42nd ACM Symposium on Theory of Computing - STOC 10, 2010. doi:10.1145/1806689.1806711.

Aaronson, Scott, and Andris Ambainis. "Forrelation: A Problem That Optimally Separates Quantum from Classical Computing." SIAM Journal on Computing 47, no. 3 (2018): 982-1038. doi:10.1137/15m1050902.

Baker, THEODORE, John Gill, and Robert SOLOVAY. "RELATIVIZATIONS OF THE P = NP QUESTION." SIAM J. 4, no. 4 (1975).

Buhrman, Harry, and Ronald De Wolf. "Complexity Measures and Decision Tree Complexity: A Survey." Theoretical Computer Science 288 (2002).

De Wolf, Ronald. "A Brief Introduction to Fourier Analysis on the Boolean Cube." THEORY OF COMPUTING LIBRARY GRADUATE SURVEYS, 2008.

Furst, Merrick, James B. Saxe, and Michael Sipser. "Parity, Circuits, and the Polynomial-Time Hierarchy." Mathematical Systems Theory 17 (1984): 13-27.

Raz, Ran, and Avishay Tal. "Oracle Separation of BQP and PH." Electronic Colloquium on Computational Complexity 107 (2018).

Tal, Avishay. "Tight Bounds on The Fourier Spectrum Of AC0." Electronic Colloquium on Computational Complexity, 2014.