NONCONSTRUCTIVE UPPER BOUNDS ON QUANTUM QUERY COMPLEXITY

ALASDAIR B. PRICE[*]

April 10, 2015

CONTENTS

1 INTRODUCTION

The field of quantum computation promises to revolutionise our society across a wide range of fronts. By utilising a new set of algorithms, which exhibit a significant improvement in efficiency over their highly infeasible classical equivalents, it is possible to find solutions to problems which are inaccessible at present. However it is crucial that we have a means of establishing whether or not a quantum speed-up is present and if a particular algorithm is optimal. To this end, we often choose to calculate query complexities, which come from the number of times an oracle must be "queried" to find some function $f(x)$. Yet there is a problem, in that previous attempts to upper bound the query complexity have been focused solely around finding new algorithms [1, 2].

In this essay, we will examine two very different methods for finding a nonconstructive upper bound. The main focus will be Lin & Lin's "bomb query complexity", which can be used for improving upper bounds on maximum bipartite matching and single source shortest paths for unweighted graphs [1]. This was heavily inspired by application of the quantum Zeno effect to an Elitzur-Vaidman bomb tester [3, 4, 5], so appendices A, B and C have been included to assist in understanding the relevant concepts.

The second technique to be talked about will be that put forward by Kimmel in 2013 [6]. Previous efforts to establish lower bounds have utilised something known as the adversary method [7], and Kimmel's work looks at whether this can also be used to find an upper limit.

[*] *Quantum Engineering Centre for Doctoral Training, Centre for Nanoscience and Quantum Information, University of Bristol, UK.*

2 BOMB QUERY COMPLEXITY

In this section and the one which directly follows, we will consider the main results presented by Lin[2] in their 2014 *arXiv* paper *"Upper bounds on quantum query complexity inspired by the Elitzur-Vaidman bomb tester"* [1]. As the title suggests, the methods they present are heavily rooted in techniques underpinning quantum bomb detection (see Appendix A). By formulating a "bomb oracle", that is a quantum oracle which triggers a failure condition when a controlled query returns 1, it is possible to determine the "bomb query complexity". From this we can find a nonconstructive upper bound for the quantum query complexity, which will be discussed in section 3.

It is imperative that whenever a quantum algorithm is developed, we understand how its run time compares to those of competing algorithms. Finding the time complexity is tricky, however the query complexity, $Q_\delta(f)$, can be equally useful. We define this as the minimum number of calls to an oracle

$$O_x |r, i\rangle = |r \oplus x_i, i\rangle \tag{1}$$

such that an unknown function $f(x)$, which takes some arbitrary Boolean input string, x, can be found with error $\leqslant \delta$. Note that $|r\rangle$ corresponds to a one-qubit record register, and $|i\rangle$, an N-dimensional index register. Later, when discussing the bomb query complexity, we will require $|r\rangle = |0\rangle$, therefore the oracle can be rewritten in the following form:

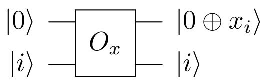

Figure 1: *The quantum oracle which takes input* $|0, i\rangle$ *and returns* $|0 \oplus x_i, i\rangle$*. Based on reference* [1].

Note that δ is often chosen to be 0.01, such that $Q_\delta(f) = Q(f)$. This choice stems from the realisation that it affects our query complexity only by a factor of $\log(1/\delta)$, which can be verified by considering the general form of an algorithm used for calculating $f(x)$ (see figure 2) and performing gap amplification[1].

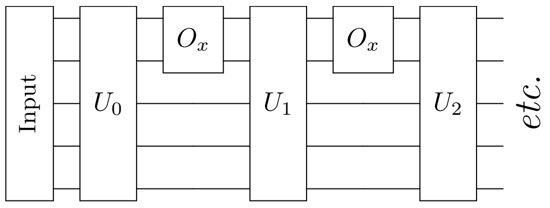

Figure 2: *The general form of an algorithm used in calculating* $f(x)$*. Based on reference* [1].

We now move to examine the Elitzur-Vaidman bomb detector, summarised in Appendix A. If the photon has already passed through the first beam splitter, we can represent the remainder of the experiment as a controlled probe:

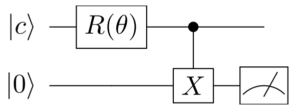

Figure 3: *Quantum bomb detector circuit with Zeno rotations. Based on reference* [1].

Setting the initial state of the control qubit to be $|0\rangle$ and defining

$$R(\theta) = e^{i\theta X} \tag{2}$$

means it will take $\frac{\pi}{2\theta}$ iterations if we wish to rotate all the way through to $|1\rangle$. The probe only operates if the bomb is live, and the probability of explosion at each step is $\mathcal{O}(\theta^2)$. Therefore we can identify

1 Gap amplification can be any one of a number of methods, however a simple example is to take a majority vote over multiple iterations of the circuit.

whether or not the bomb is live after $\mathcal{O}(1/\theta)$ repetitions, with only an $\mathcal{O}(\theta)$ chance of detonation.

From this, we can start to assemble a circuit in the bomb query model. Combining elements of figures 1 and 3, the basic configuration is illustrated below:

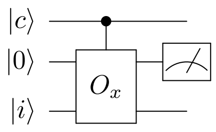

Figure 4: *A quantum query circuit, restricted by the bomb query model. Based on reference* [1].

Figure 4 represents the operation

$$CO_x |c, r, i\rangle = |c, r \oplus (c \cdot x_i), i\rangle \tag{3}$$

such that the algorithm will terminate (that is, the bomb will explode) if our measurement of $c \cdot x_i$ returns 1. Here, $\cdot$ can be regarded as logical AND.

Another way of viewing the above is to consider the controlled version of the operator

$$P_{x,0} = \sum_{x_i=0} |i\rangle |i\rangle \tag{4}$$

which we will write as

$$\begin{aligned} CP_{x,0} &= \sum_{i=1}^{N} |0, i\rangle \langle 0, i| + \sum_{x_i=0} |1, i\rangle \langle 1, i| \\ &= I - \sum_{x_i=1} |1, i\rangle \langle 1, i| \end{aligned} \tag{5}$$

This can be illustrated more compactly in circuit notation

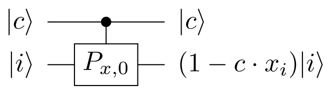

Figure 5: *A "controlled-P" gate, which is equivalent to the bomb oracle in figure 4. From reference* [1].

and so figure 2 can be redrawn as follows

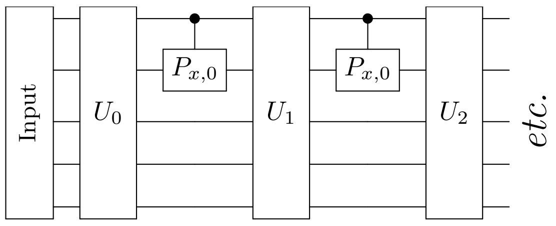

Figure 6: *General form of an algorithm in the bomb query model, used for calculating* $f(x)$. *Defining CP as the "bomb oracle" allows for easy comparison with figure 2. Based on reference* [1].

Finally, this framework allows us to introduce the bomb query complexity, $B_{\epsilon,\delta}(f)$, which we define as:

Definition: *The minimum number of calls which an algorithm must make to the bomb query circuit, such that axioms 1 and 2 hold for all* x.

Axiom 1: *At the end of the algorithm, the probability that* $c \cdot x_i \neq 1$ *is greater or equal to* $1 - \epsilon$.

Axiom 2: *The probability of failure (i.e. that the bomb explodes* or *that* $f(x)$ *is returned incorrectly) is at most* δ*, where* $\delta \geqslant \epsilon$.

Of course this model is simply a means to an end, and we must not lose sight of the fact that we require a nonconstructive upper bound for the query complexity. Therefore our next step will be to look into how $B_{\epsilon,\delta}(f)$ can help us achieve such a goal.

3 UPPER BOUNDING THE BOMB QUERY COMPLEXITY

It is possible to simulate O_x using the following circuit,

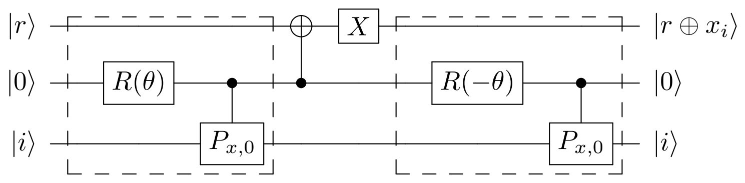

Figure 7: *Circuit with which we can simulate* O_x*. Based on reference* [1].

We define the Zeno rotations

$$R(\varphi) = \begin{pmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{pmatrix} \tag{6}$$

where $\varphi = \pm\theta$, and

$$\theta = \frac{\pi}{2n} \tag{7}$$

n is the number of times the highlighted sections of figure 7 will be iterated over, such that $2n$ calls are made to the bomb oracle. Upon reaching the end of the circuit, the $|0\rangle$ ancilla can be discarded and, after taking this into account, we find that inputting the general state $\sum\limits_{r,i} A_{r,i}|r,i\rangle$ returns

$$\sum_{r,i} A_{r,i}\cos^{2nx_i}\left(\frac{\pi}{2n}\right)|r \oplus x_i, i\rangle \tag{8}$$

Note that since $|r,i\rangle \to |r \oplus x_i, i\rangle$, this is of a similar form to equation 1. Therefore our earlier statement, that this circuit simulates the quantum oracle, must be true.

We show equation 8 to be correct by examining outcomes for all possible values of x_i. First, if $x_i = 0$,

$$P_{x,0}|i\rangle = \left(\sum_{x_i=0}|i\rangle\langle i|\right)|i\rangle = |i\rangle \tag{9}$$

This means that, for a single iteration, the first highlighted part of figure 7 corresponds to a $\frac{\pi}{2}$ rotation of the ancilla, such that the bit-flip on $|r\rangle$ cancels out the effect of the CNOT. The second highlighted section rotates $|1\rangle \to |0\rangle$, so our final state will be the same as our initial. Once the ancilla has been discarded, the overall effect of the circuit is

$$\sum_{r,i} a_{r,i}|r,i\rangle \to \sum_{r,i} a_{r,i}|r,i\rangle \tag{10}$$

When $x_i = 1$,the situation is slightly less straightforward. We observe that

$$CP_{x,0}|0,i\rangle = \left(I - \sum_{x_i=1}|1,i\rangle\langle 1,i|\right)|0,i\rangle = |0,i\rangle \tag{11}$$

and

$$CP_{x,0}|1,i\rangle = \left(I - \sum_{x_i=1} |1,i\rangle\langle 1,i|\right)|1,i\rangle = |1,i\rangle - |1,i\rangle = 0 \tag{12}$$

Therefore the two highlighted sections can be represented as follows,:

$$CP_{x,0}R(\varphi)|0,i\rangle = CP_{x,0}\left(\cos\varphi|0\rangle + \sin\varphi|1\rangle\right)|i\rangle = \cos\varphi|0,i\rangle \tag{13}$$

where, unlike in the original paper, the rotations have been generalised to be in terms of φ.

This state is clearly different to that which we had initially, so we must extend our result to cases where $n > 0$. In other words, we find

$$\cos\varphi \to \cos^{2n}(\varphi) \tag{14}$$

Since the ancilla is projected into 0 by $CP_{x,0}$, the CNOT does nothing, so applying X will send $|r\rangle \to |r \oplus 1\rangle$. As a result, once we have removed the ancilla, the final state will be

$$\cos^{2n}\theta\,|r \oplus 1, i\rangle \tag{15}$$

By linearity, equations 10 and 15 can be combined to return equation 8.

Given a quantum algorithm requiring $Q_{\delta'}(f)$ queries, where $\delta' = \delta - \epsilon$, it is possible to use the circuit in figure 7 to construct a new algorithm, of the form illustrated in figure 6. If we choose

$$n = \left\lceil \frac{\pi^2}{2\epsilon} Q_{\delta'}(f) \right\rceil \tag{16}$$

then the algorithm will require $2nQ_{\delta'}(f) < \frac{\pi^2}{\epsilon}Q^2_{\delta'}(f) + 2Q_{\delta'}(f)$ queries to the bomb oracle. Although the authors do not make it clear as to where the extra $2Q_{\delta'}(f)$ comes from, it seems reasonable to assume that this arises from the error on the circuit in figure 7, which is $\mathcal{O}(1/n)$.

As a result,

$$\begin{aligned} B_{\epsilon,\delta}(f) &< \frac{\pi^2}{\epsilon}Q^2_{\delta-\epsilon}(f) + 2Q_{\delta-\epsilon}(f) \\ &= \mathcal{O}\left(\frac{Q^2_{\delta-\epsilon}(f)}{\epsilon}\right) \end{aligned} \tag{17}$$

It is clear that

$$\begin{aligned} B_{\epsilon,\delta}(f) &< B_{\epsilon/2,\delta} \\ &= \mathcal{O}\left(\frac{Q^2_{\delta-\epsilon/2}(f)}{\epsilon}\right) \end{aligned} \tag{18}$$

and by considering $\frac{\delta}{2} \geqslant \delta - \frac{\epsilon}{2}$, Lin[2] find the upper bound on the bomb query complexity to be

$$B_{\epsilon,\delta}(f) = \mathcal{O}\left(\frac{Q^2_{\delta}(f)}{\epsilon}\right) \tag{19}$$

For the examples given in the introduction, this information is enough to derive a nonconstructive upper bound on $Q(f)$.

4 QUANTUM ADVERSARY BOUND

The general adversary bound is not a new concept. Behaving well under composition, it has found previous uses in probing lower bounds on quantum query complexity. This section is based on the work presented as part of *"Quantum Adversary (Upper) Bound"* by Shelby Kimmel [6], in which she examines how the general adversary bound can be utilised in finding a nonconstructive upper bound for query complexity. Our main purpose here is to broadly illustrate how a radically different approach can reach a goal similar to that which the bomb query complexity strives toward, rather than to recreate the underpinning mathematics in the same level of detail. Hence, the following two statements (which [6] and [8] have shown to be correct) will be made without proof:

$$ADV^{\pm}(f^d) \geqslant (ADV^{\pm}(f))^d \tag{20}$$

$$Q(f) = \Theta(ADV^{\pm}(f)) \tag{21}$$

Note that equation 20 applies only for $d \in \mathbb{N}$ and $f : S \to \{0, 1\}$, where $S \subseteq \{0, 1\}^n$. In the case of equation 21, we require $f : S \to E$, where $S \in D^n$ and $|D|, |E| \in \mathbb{N}$.

Let us now consider the general algorithm for f^d. If it takes $\mathcal{O}(J^d)$ queries to reach a solution within our desired error margins, equation 21 tells us that

$$ADV^{\pm}(f^d) = \mathcal{O}(J^d) \tag{22}$$

By application of 20, we can see that

$$(ADV^{\pm}(f))^d \leqslant \mathcal{O}(J^d) \tag{23}$$

Note that in the original paper, the two sides of this equation are said to be equal. For reasons of clarity, I have amended this to become an inequality.

It is evident, therefore, that the upper bound on the general adversary bound is

$$ADV^{\pm}(f) = \mathcal{O}(J) \tag{24}$$

From here, we can re-apply 21, returning an upper bound on the quantum query complexity,

$$Q(f) = \mathcal{O}(J) \tag{25}$$

We have demonstrated that it is possible to determine whether or not the best known algorithm for f is optimal, without the need to construct a superior algorithm. Utilising the behaviour of the general adversary bound under composition, in addition to its tightness with query complexity, we are able to establish an upper bound with relative ease. While this bears no resemblance to bomb complexity, it does illustrate the power of non-constructive methods in general, and their usefulness should not be underestimated.

5 CONCLUSION

To summarise, this essay has discussed two different methods for establishing an upper bound on quantum query complexity, without having to construct new algorithms in the process. The main focus has been on the bomb query model, and the physics which inspired it. In particular, Lin & Lin found the relationship between bomb and quantum query complexities to be

$$B_{\epsilon,0.01}(f) = \Theta\left(\frac{Q_{0.01}^2(f)}{\epsilon}\right) \tag{26}$$

with upper bound

$$B_{\epsilon,\delta}(f) = \mathcal{O}\left(\frac{Q_{\delta}^2(f)}{\epsilon}\right) \tag{27}$$

By constructing a bomb query algorithm and utilising this relationship, one can find nonconstructive upper bounds on the quantum query complexity for certain problems.

Shelby Kimmel tried a contrasting approach, utilising a technique which was already well-established for other tasks; the general adversary bound. She found that, if $\mathcal{O}\left(J^d\right)$ queries are needed to find f^d, the upper bound on the quantum query complexity will be

$$Q(f) = \mathcal{O}(J) \tag{28}$$

This improves the upper bound for problems different to those in which the bomb query model is applicable, such as the 1-Fault NAND Tree [6, 9]. Unfortunately, Kimmel herself states that she is unsure whether any other examples exist. However even if this is the case, the fact that the quantum adversary bound works in just one situation where the bomb query model does not lends hope that it may be eventually possible to find nonconstructive upper bounds for most, if not all problems. Given the importance of discovering whether or not a particular algorithm is optimal, the value of this should not be underestimated.

A INTERACTION-FREE MEASUREMENTS

It is of fundamental understanding that when direct measurements are made of a quantum mechanical state, the system will be disturbed. In 1993, Elitzur and Vaidman [3] uncovered a method which circumvented this, allowing specific properties of a system to be determined without interaction. Their ideas were experimentally verified two years later by Kwiat *et al.* [4], who also introduced the idea of utilising the quantum Zeno effect to push the efficiency of the system arbitrarily close to 100%. That which follows is based on the contents of the papers cited herein, along with section 2.1 of reference [1].

Suppose we have a bomb which detonates if and only if a photon is incident upon its surface. We know there is a non-zero probability of the bomb being defective (in this case, the detonator is missing, leaving a hole for photons to pass straight through), and so wish to determine which ones in our stockpile are not going to explode upon exposure to light. Performing a direct measurement will naturally destroy any of live bombs, meaning we must avoid such interactions if any of of these are still to be usable by the end.

We begin by considering a Mach-Zehnder interferometer, as illustrated in figure 8. A single photon enters the system in either of the two modes of the bottom-left 50:50 beam splitter. If the photon is moving up the page, we will consider it to be in the state $|u\rangle$ (as it is initially), whereas going to the right corresponds to $|r\rangle$. Our choice of input port at the beginning is entirely arbitrary, as the transformations which follow are mathematically identical for both.

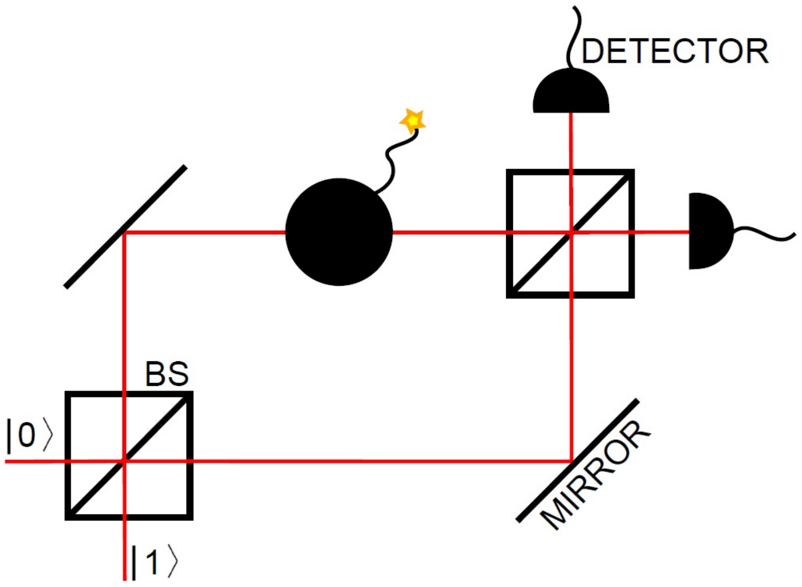

Figure 8: *A schematic of the Elitzur-Vaidman quantum bomb detector in bulk optics. A single photon enters the Mach-Zehnder interferometer at the lower of the two 50:50 beam splitters (BS), where $|0\rangle$ and $|1\rangle$ are number states.*

To understand what is happening here, we first examine the effect of a single beam splitter:

$$\begin{aligned} |u\rangle &\to \frac{1}{\sqrt{2}}\left[|u\rangle + i\,|r\rangle\right] \\ |r\rangle &\to \frac{1}{\sqrt{2}}\left[|r\rangle + i\,|u\rangle\right] \end{aligned} \tag{29}$$

Note the reflected term always experiences a $\frac{\pi}{2}$ phase shift. The mirror transformations behave in a similar fashion, that is

$$\begin{aligned} |u\rangle &\to i\,|r\rangle \\ |r\rangle &\to i\,|u\rangle \end{aligned} \tag{30}$$

From equations 29 and 30 we can see that, where no further objects are present, the state of the photon will evolve as follows:

$$\begin{aligned} |u\rangle &\to \frac{1}{\sqrt{2}}\left[|u\rangle + i\,|r\rangle\right] \\ &\to \frac{1}{\sqrt{2}}\left[i\,|r\rangle - |u\rangle\right] \\ &\to \frac{1}{2}\left[i\,|r\rangle - |u\rangle\right] - \frac{1}{2}\left[i\,|r\rangle + |u\rangle\right] \\ &= -\,|u\rangle \end{aligned} \tag{31}$$

We now consider the case where a live bomb obstructs one of the interferometer arms. Recall that a dud is effectively transparent, so equation 31 represents this adequately.

In the presence of an opaque body, photons passing through the Mach-Zehnder will be scattered after the second mirror, such that

$$\begin{aligned} |u\rangle &\to \frac{1}{\sqrt{2}}\left[|u\rangle + i\,|r\rangle\right] \\ &\to \frac{1}{\sqrt{2}}\left[i\,|r\rangle - |u\rangle\right] \\ &\to -\frac{1}{\sqrt{2}}\,|u\rangle + \frac{1}{\sqrt{2}}\,|0\rangle \\ &\to -\frac{1}{2}\left[i\,|r\rangle + |u\rangle\right] + \frac{1}{\sqrt{2}}\,|0\rangle \end{aligned} \tag{32}$$

Unlike in equation 31 where the photon passed through the hole for the detonator undisturbed, we can no longer be certain as to which detector will click when a photon exits the interferometer. If we have a live bomb, we can determine it is live in 25% of all cases, as one of the detectors which fires cannot do so if the bomb is a dud. In a further 25% of cases we will be unsure, and in 50% of cases the photon will be scattered, so the bomb will explode. While the chances of success are against us, this shows that, surprisingly, it is possible to perform an interaction free measurement in a quantum environment.

An important point of note is that if we have a dud, we can never be entirely certain that it is defective. However if, after repeating the experiment a large number of times, the bomb has not exploded and we have only ever seen clicks in the detector corresponding to $|u\rangle$, it is reasonable to assume the bomb is not live.

Clearly our next move must be to consider whether or not we can improve on our chances of finding a live bomb. To do this, we must consider what happens when we repeat the experiment.

It has already been established that the probability of correctly identifying a working device in a single iteration of the circuit is $\frac{1}{4}$, independent of the number of times a bomb has been probed and the same as the chance of being unsure. If we cannot tell whether or not the bomb is a dud after inputting one photon, we must try again. The odds of success are now determined by the probability that we correctly identify the bomb as being live given we could not do so in the first round, i.e.

$$\begin{aligned}\text{Prob}(\textit{success in round 2}) &= \text{Prob}(\textit{unsure in round 1}) \times \text{Prob}(\textit{know bomb is live with certainty}) \\ &= \qquad\qquad \frac{1}{4} \qquad\qquad \times \qquad\qquad \frac{1}{4}\end{aligned} \tag{33}$$

Similarly, the chances that we succeed in a third iteration of the Mach-Zehnder, given we were unable to do so in both the first and second rounds, are

$$\begin{aligned}\text{Prob}\begin{pmatrix}\textit{success in}\\ \textit{round 3}\end{pmatrix} &= \text{Prob}\begin{pmatrix}\textit{unsure in}\\ \textit{round 1}\end{pmatrix} \times \text{Prob}\begin{pmatrix}\textit{unsure in}\\ \textit{round 2}\end{pmatrix} \times \text{Prob}\begin{pmatrix}\textit{know bomb is}\\ \textit{live with certainty}\end{pmatrix} \\ &= \qquad \frac{1}{4} \qquad \times \qquad \frac{1}{4} \qquad \times \qquad \frac{1}{4}\end{aligned} \tag{34}$$

We now assume the bomb can no longer be tested after the third cycle. If this is the case, the probability that we discover the bomb is live without exploding it will be

$$\begin{aligned}\text{Prob}\,(\textit{overall success}) &= \text{Prob}\begin{pmatrix}\textit{success in}\\ \textit{round 1}\end{pmatrix} + \text{Prob}\begin{pmatrix}\textit{success in}\\ \textit{round 2}\end{pmatrix} + \text{Prob}\begin{pmatrix}\textit{success in}\\ \textit{round 3}\end{pmatrix} \\ &= \qquad \frac{1}{4} \qquad + \qquad \frac{1}{4^2} \qquad + \qquad \frac{1}{4^3}\end{aligned} \tag{35}$$

Choosing to terminate the sequence here makes it easy to identify a pattern. Yet in reality, $n = 3$ is not the maximum number of iterations, so we find

$$\text{Prob}(\textit{overall success}) = \sum_{n=1}^{\infty} \frac{1}{4^n} = \frac{1}{3} \tag{36}$$

A method of improving this probability was first given by Elitzur and Vaidman themselves. By modifying the two 50 : 50 beam splitters, such that the first has close to 100% transmissivity and the second is almost totally reflective, we can re-run the mathematics to find our overall chances of success increase to $\frac{1}{2}$. However Kwiat *et al.* managed to go one better.

The quantum Zeno effect is a phenomena first explained by Misra and Sudarshan in 1977 [5]. Through continuous observation of a physical system, it is possible to prevent evolution away from an initial state. In reality, only discrete measurements can be made, however by minimising the time between each, the probability of decay will be close to zero (see Appendix B).

Let us first consider a series of Mach-Zehnder interferometers, such as those in figure 9(i). Assume a suitably large number of beam splitters, N, each with reflectivity

$$R = \cos^2\left(\frac{\pi}{2N}\right) \tag{37}$$

One should note that while the outcome of this first part will be unaffected by our choice of N, we will later require it not be too small.

Like in the original scheme, we now inject a single photon into our setup. If no object is present (or alternatively, if our bomb is a dud), we will observe an evolution of the photon's state from the lower path to the upper, meaning a detector placed in the top-right of our diagram will fire with certainty. To aid in the visualisation of this, a simulation has been included in Appendix C, demonstrating two- and three-beam splitter arrangements.

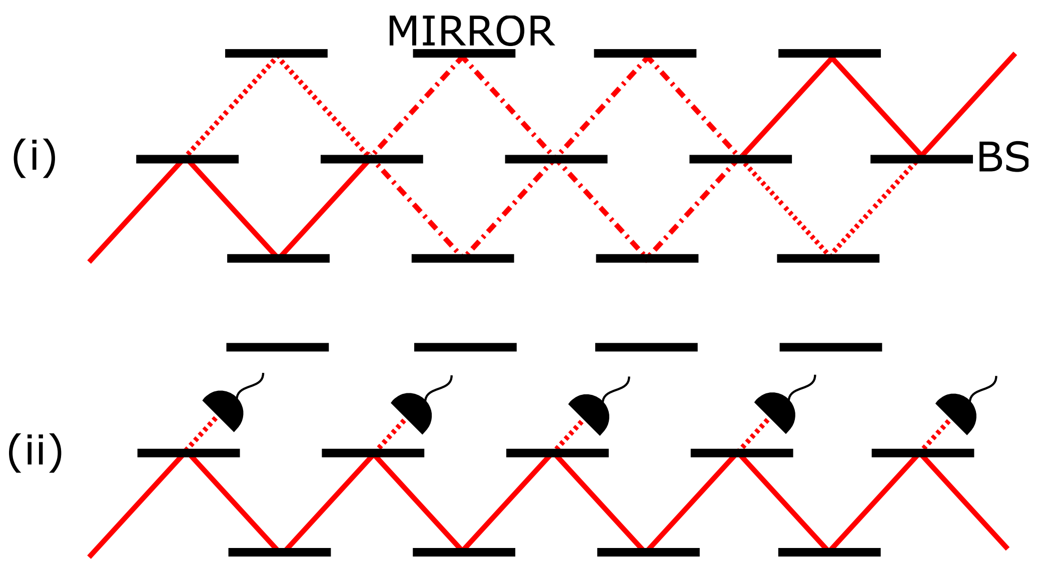

Figure 9: *Multiple quantum bomb testers in series, utilising the quantum Zeno effect. Grouped together as a single entity, the detectors represent a live bomb. Based on figure 1 in reference [4].*

We now turn our attention to figure 9(ii). For each beam splitter, a "detector" has been introduced on the upper path; grouping these together constitutes our bomb. If N is large, the beam splitter reflectivities will be high, so the chances that the bomb detonates will be small. If the photon fails to cause an explosion, we can be sure that it was reflected, meaning its state will be projected onto the lower path. In effect, this is subjecting our quantum probe to repeated measurements, thereby forcing it to remain in the bottom half of the system with high probability. To put it in another way, if a bomb is present, we can perform an interaction-free measurement with odds corresponding to the likelihood of the photon being reflected at each beam splitter,

$$P = \left[\cos^2\left(\frac{\pi}{2N}\right)\right]^N \tag{38}$$

We require only $N = 4$ to beat Elitzur and Vaidman's maximum efficiency of 50%, and for $N \to \infty$,

$$P \to 1 - \frac{\pi^2}{4N} + \mathcal{O}(N^{-2}) \tag{39}$$

The result of this is a large number of beam splitters will yield close to a 100% success rate, thereby proving it is possible make interaction-free measurements with arbitrarily high probability.

B QUANTUM ZENO EFFECT

The quantum Zeno effect was originally presented as a supposed paradox, whereby an excited particle under continuous observation will never been seen to decay. Not only can this be utilised for forcing a system to remain in a particular quantum state following an initial observation, but it has also been applied to the Elitzur-Vaidman bomb detector, pushing the probability of explosion arbitrarily close to zero [4]. In this section, we will examine the mathematical reasoning for the Zeno effect, based on the original derivation in reference [5], though also deviating away from this in many respects.

If we consider a particle in an initial state ρ, then the probability of it decaying out of said state, over the interval $\Delta = [0, t]$ can be represented as $P(0, t; \rho)$. The probability of there being no decay during the same interval is defined to be $Q(0, t; \rho)$, such that

$$P(0, t; \rho) + Q(0, t; \rho) = 1 \tag{40}$$

We now decide to take $n + 1$ instantaneous measurements at times

$$0, \frac{t}{n}, \frac{2t}{n}, \ldots, \frac{(n-1)t}{n} \tag{41}$$

Whenever a measurement is made, it will trigger a state collapse

$$\rho \to \rho' = E\rho E \tag{42}$$

where E is an orthogonal projection in our Hilbert space onto the undecayed states. If we account for unitary evolution U of the quantum state in between the times as which measurements are taken, then we can define

$$T_n(t) = \left[EU\left(\frac{t}{n}\right)E\right]^n = \left[E\exp^{-iHt/n}E\right]^n \tag{43}$$

such that

$$\rho(n,t) = T_n(t)\rho T_n^*(t) \tag{44}$$

$\rho(n,t)$ is the final state after $n+1$ measurements have been made. If by this point, the particle still has not decayed,

$$\rho(n,t) = \rho \tag{45}$$

This means our probability Q can be written in the form [10]

$$\text{Prob}(m) = \text{Tr}\left[M_m \rho M_m^\dagger\right] \tag{46}$$

giving the expression

$$Q(\Delta, n; \rho) = \text{Tr}\left[T_n(t)\rho T_n^*(t)\right] \tag{47}$$

We now define

$$\rho(t) = \underset{n\to\infty}{\text{s-lim}}\, \rho(n,t) \tag{48}$$

Here, s-lim corresponds to the strong operator topology [11], where any set of self-adjoint operators $\{O_n\}$ will strongly converge on the self-adjoint operator O if $(O_n - \lambda)^{-1}$ strongly converges to $(O-\lambda)^{-1}$ for $\forall\lambda \in \mathbb{C}$, $\text{Im}\,\lambda \neq 0$ [12].

If, for $t \geqslant 0$,

$$T(t) = \underset{n\to\infty}{\text{s-lim}}\, T_n(t) \tag{49}$$

then we are able to evaluate $Q(\Delta;\rho)$ as the limit of $Q(\Delta, n;\rho)$ when $n \to \infty$. Therefore, by equation 40,

$$P(\Delta;\rho) = 1 - \text{Tr}\left[T(t)\rho T^*(t)\right] \tag{50}$$

Using the provable assertion that

$$EU(t) = U(t)E \tag{51}$$

for $\forall t \in \mathbb{R}$, we show

$$T(t)T(s) = [EU(t)E][EU(s)E] = EEU(t)U(s)EE \tag{52}$$

Any projector P must satisfy the relation [10]

$$P^2 = P \tag{53}$$

thus

$$T(t)T(s) = Ee^{-iHt}e^{-iHs}E = Ee^{-iH(t+s)}E = T(t+s) \tag{54}$$

If we now turn back to equation 43, then it is clear that

$$T_n(-t) = \left[EU\left(\frac{-t}{n}\right)E \right]^n \tag{55}$$

Consider also

$$T_n^*(t) = \left[\left[EU\left(\frac{t}{n}\right)E \right]^n \right]^* = \left[EU\left(\frac{-t}{n}\right)E \right]^n \tag{56}$$

Equation 55 converges strongly to $T(-t)$ as $n \to \infty$ and, equivalently [13], equation 56 converges weakly to $T^*(t)$ as $n \to \infty$, therefore

$$T^*(t) = T(-t) \tag{57}$$

It is at this point where we diverge from the final pages of reference [5]. That which follows favours clarity of explanation over the mathematical rigour of the original paper, however our conclusions come out to be the same.

By application of equations 43, 49, 52 and 57, we find

$$\begin{aligned} T^*(t)T(t) &= T(-t)T(t) \\ &= T(-t+t) \\ &= T(0) \\ &= \operatorname*{s-lim}_{n\to\infty} \left[EU\left(\frac{0}{n}\right)E \right]^n \\ &= \operatorname*{s-lim}_{n\to\infty} [EE]^n \end{aligned} \tag{58}$$

Using relation 53,

$$T^*(t)T(t) = \operatorname*{s-lim}_{n\to\infty} [E]^n = E \tag{59}$$

This, along with the cyclic properties of the trace, allows us to rewrite equation 50 as

$$P(\Delta;\rho) = 1 - \mathrm{Tr}[\rho E] \tag{60}$$

It is obvious that the quantum Zeno effect is only going to be of relevance if the initial state is undecayed. By this condition,

$$\mathrm{Tr}[\rho E] = 1 \tag{61}$$

As such it must be true that, in the limit of $n \to \infty$, we find $P(\Delta;\rho) \to 0$. Therefore, if we take repeated measurements of a state, only allowing a short amount of time to elapse between each measurement, the probability of decay becomes negligible.

C SIMULATING MULTIPLE BEAM SPLITTERS IN SERIES

The following simulation was written in *Wolfram Mathematica 10.1*, to illustrate the output from figure 9(i). In the cases evaluated below, N was chosen such that it represented two- and three-beam splitter arrangements respectively.

When interpreting states in this model, $1_1 \otimes 0_2$ means the photon is in the upper half of the system, whereas $0_1 \otimes 1_2$ corresponds to the lower half. We see that the final states are both $1_1 \otimes 0_2$ with associated probabilities of 1, therefore if the bomb is a dud, the upper detector will fire with certainty.

Bomb Detector Simulator version 1.0

Written by Alasdair Price, April 2015

Defining Fock Representation

```
In[6]:= fock[n1_, n2_] := α * n1_{1}⊗n2_{2} (* Require non-
         valued α and β for nested creation functions to work correctly *)
        c_{1}[α_ * n1__{1}⊗n2__{2}] := α * √(n1 + 1) * (n1 + 1)_{1}⊗n2_{2}
        (* Defines creation (raising) operator (a^†) for upper rail *)
        c_{2}[α_ * n1__{1}⊗n2__{2}] := α * √(n2 + 1) * n1_{1} ⊗ (n2 + 1)_{2}
        (* Defines creation (raising) operator (a^†) for lower rail *)
        extract1[α_ * n1__{1}⊗n2__{2}] := n1 (* Extracts photon number from upper rail *)
        extract2[α_ * n1__{1}⊗n2__{2}] := n2 (* Extracts photon number from lower rail *)
        sumnumber[α_ * n1__{1}⊗n2__{2}] := n1 + n2
        (* Calculates total photon number across both rails *)
        raise11[input_] := Nest[c_{1}, fock[0, 0], extract1[input]]
        (* Applies the raising (creation) operator to upper rail a number
         of times corresponding to incoming number of photons in rail *)
        r^1_{1}[input_] := If[MatchQ[input, α_ * n1__{1}⊗n2__{2}],
          raise11[input], Map[raise11, input]]
          (* Generalises raise11 to work for an arbitrary superposition of inputs *)
        raise12[input_] := Nest[c_{2}, fock[0, 0], extract2[input]]
        (* Applies the raising (creation) operator to lower rail a number
         of times corresponding to incoming number of photons in rail *)
        r^1_{2}[input_] := If[MatchQ[input, α_ * n1__{1}⊗n2__{2}],
          raise12[input], Map[raise12, input]]
        (* Generalises raise12 to work for an arbitrary superposition of inputs *)
        raise21[input_] := Nest[c_{1}, fock[0, 0], sumnumber[input]]
        (* Applies the raising (creation) operator to upper rail a number of times
         corresponding to total incoming number of photons in both rails *)
        r^2_{1}[input_] := If[MatchQ[input, α_ * n1__{1}⊗n2__{2}],
          raise21[input], Map[raise21, input]]
        (* Generalises raise21 to work for an arbitrary superposition of inputs *)
        raise22[input_] := Nest[c_{2}, fock[0, 0], sumnumber[input]]
        (* Applies the raising (creation) operator to lower rail a number of times
         corresponding to total incoming number of photons in both rails *)
        r^2_{2}[input_] := If[MatchQ[input, α_ * n1__{1}⊗n2__{2}],
          raise22[input], Map[raise22, input]]
```

```
(* Generalises raise22 to work for an arbitrary superposition of inputs *)
c12[α_ * n1_1⊗n2_2] := c1[c2[fock[0, 0]]]
(* Applies the creation operator once to each rail *)
setab := α = 1 (* To be run at the end of the program,
setting α and β equal to 1 so as to normalise input state *)
revab := α =.(* To be run following the return of the solution,
clearing α and β so as to allow the
 program to be executed for a different scenario *)
return[x_] := (setab; outputstates = x; revab; Collect[Collect[
    Collect[Collect[Collect[outputstates, 0_1⊗2_2], 1_1⊗1_2], 2_1⊗0_2], 1_1⊗0_2], 0_1⊗1_2])
Normalising[α_ * n1_1⊗n2_2] := n1_1⊗n2_2
Normalise[input_] := If[MatchQ[input, _ * n1_1⊗n2_2], Normalising[input],
  Map[Normalising[#] &, input]] (*Extracts component states without co-
 efficients. IMPORTANT: This does not normalise per se,
however when used in the context of this code,
the component states are returned individually,
and are therefore normalised*)
Prob[α_ * n1_1⊗n2_2] := Abs[return[α]]^2
(*Returns probability for any state of form α|n1n2⟩*)
Pr[input_] := Module[{input1},
  If[MatchQ[input, α_ * n1_1⊗n2_2], {Prob[input]}, input1 = Apply[List, input];
   Map[Prob, input1]]] (*Extracts probabilities for
  each term of fock state of arbitrary form*)
```

Beam Splitter

```
(* NB: At present, this has been coded for a maximum of 2 photons *)
```

```
In[3]:=  UDC[η_] := RootApproximant[( √η           i * √(1 - η) )]
                                    ( i * √(1 - η)   √η          )
         (* Directional coupler unitary for a dual rail where η =
          transmission probability *)
         DC[η_, α_ * n1_1⊗n2_2] :=
          DiscreteDelta[n1 - n2] * (α * UDC[η] [[1, 1]] * UDC[η] [[2, 1]] * r^2_1[α * n1_1⊗n2_2] +
              α * UDC[η] [[1, 2]] * UDC[η] [[2, 2]] * r^2_2[α * n1_1⊗n2_2] +
              (UDC[η] [[1, 1]] * UDC[η] [[2, 2]] + UDC[η] [[1, 2]] * UDC[η] [[2, 1]]) * (c_12[α * n1_1⊗n2_2]))
             (*|11⟩_input term*) + DiscreteDelta[n2] * DiscreteDelta[n1 - 1] *
            (α * UDC[η] [[2, 2]] * r^2_1[α * n1_1⊗n2_2] + α * UDC[η] [[1, 2]] * r^2_2[α * n1_1⊗n2_2])
             (*|10⟩_input term*) + DiscreteDelta[n1] * DiscreteDelta[n2 - 1] *
            (α * UDC[η] [[2, 1]] * r^2_1[α * n1_1⊗n2_2] + α * UDC[η] [[1, 1]] * r^2_2[α * n1_1⊗n2_2])
            (*|01⟩_input term*) + DiscreteDelta[n1 - 2] * DiscreteDelta[n2] *
             Expand[Apart[(α * UDC[η] [[1, 1]] * UDC[η] [[2, 2]] * r^2_1[α * n1_1⊗n2_2] + α * UDC[η] [[1, 2]] *
                    UDC[η] [[2, 1]] * r^2_2[α * n1_1⊗n2_2] + α * (UDC[η] [[1, 1]] * UDC[η] [[2, 1]] +
                       UDC[η] [[1, 2]] * UDC[η] [[2, 2]]) * (c_12[α * n1_1⊗n2_2])) / (√2)]]
              (*|20⟩_input term*) + DiscreteDelta[n2 - 2] * DiscreteDelta[n1] *
             Expand[Apart[(α * UDC[η] [[1, 2]] * UDC[η] [[2, 1]] * r^2_1[α * n1_1⊗n2_2] +
                  α * UDC[η] [[2, 2]] * UDC[η] [[1, 1]] * r^2_2[α * n1_1⊗n2_2] +
                  α * (UDC[η] [[1, 1]] * UDC[η] [[2, 1]] + UDC[η] [[1, 2]] * UDC[η] [[2, 2]]) *
                    (c_12[α * n1_1⊗n2_2])) / (√2)]] (*|02⟩_input term*)
         BS[η_, input_] := If[MatchQ[input, α_ * n1_1⊗n2_2], DC[η, input],
            Map[DC[η, #] &, input]]
          (* Generalises DC to work for an arbitrary superposition of inputs *)

In[65]:= BSiterate[η_, input_, N_] := Module[{x, i}, x = BS[η, input];
            For[i = 2, i < N + 1, i++, x = BS[η, x]]; x] (* Iterates beam splitter N times *)

In[57]:= out = BSiterate[(Cos[π / (2 * 2)])^2, fock[0, 1], 2]; (* N=2 *)
         "Output State:"
         Normalise[out]
         "Probability:"
         Pr[out] [[1]]

Out[57]= Output State:

Out[58]= 1_1⊗0_2

Out[59]= Probability:

Out[60]= 1
```

```
In[61]:= out = BSiterate[(Cos[π/(2*3)])^2, fock[0, 1], 3]; (* N=3 *)
         "Output State:"
         Normalise[out]
         "Probability:"
         Pr[out][[1]]
```

```
Out[61]= Output State:
```

Out[62]= $1_1 \otimes 0_2$

```
Out[63]= Probability:

Out[64]= 1
```

REFERENCES

[1] C. Y.-Y. Lin and H.-H. Lin, "Upper bounds on a quantum query complexity inspired by the Elitzur-Vaidman bomb tester," *arXiv:1410.0932v2 [quant-ph]*, 2014.

[2] A. Ambainis, "Quantum walk algorithm for element distinctness," *SIAM Journal on Computing 37*, 2007.

[3] A. C. Elitzur and L. Vaidman, "Quantum Mechanical Interaction-Free Measurements," *Found. Phys.*, 1993.

[4] P. Kwiat *et al.*, "Interaction-Free Measurement," *Phys. Rev. Lett.*, 1995.

[5] B. Misra and E. Sudarshan, "The Zeno's paradox in quantum theory," *J. Math. Phys.*, 1977.

[6] S. Kimmel, "Quantum Adversary (Upper) Bound," *Chicago Journal of Theoretical Computer Science*, 2013.

[7] A. Ambainis, "Quantum lower bounds by quantum arguments," *Proc. 3rd ACM STOC*, 2000.

[8] T. Lee *et al.*, "Quantum query complexity of state conversion," *Integr. Equ. Oper. Theory*, 2011.

[9] B. Zhan *et al.*, "Super-Polynomial Quantum Speed-ups for Boolean Evaluation Trees with Hidden Structure," *Proc. 3rd ACM ITCS*, 2012.

[10] M. A. Nielsen and I. L. Chuang, "Quantum Computation and Quantum Information: 10th anniversary edition," *Cambridge University Press*, 2010.

[11] P. Exner *et al.*, "Zeno Product Formula Revisited," *Proc. IEEE FOCS*, 2007.

[12] M. Takesaki, "Theory of Operator Algebras II," *Springer*, 2003.

[13] M. Takesaki, "Theory of Operator Algebras I," *Springer*, 2002.