2016-03-15

A Hybrid Computer Case Study for Unconventional Virtual Computing

Kirke, Alexis

http://hdl.handle.net/10026.1/6618

International Journal of Unconventional Computation
Old City Publishing

A Hybrid Computer Case Study for Unconventional Virtual Computing

[Abbreviated Title: Hybrid Unconventional Virtual Computing]

Alexis J. Kirke[1], Peter Shadbolt[2,3], Alex Neville[2], Eduardo R. Miranda[1]

[1]Interdisciplinary Centre for Computer Music Research, School of Humanities, Music and Performing Arts, University of Plymouth, Drake Circus, Plymouth, PL4 8AA, UK

[2]Centre for Controlled Quantum Dynamics, Imperial College London, London SW7 2AZ, UK

[3]Centre for Quantum Photonics, H. H. Wills Physics Laboratory, University of Bristol, Tyndall Avenue, Bristol, BS8 1TL, UK

Alexis.Kirke@Plymouth.ac.uk

Abstract. *Improvements in computer efficiency are not always due to increasing computation speed. The mouse and GUI approach to OS's actually slowed down computation, but sped up computing. This paper highlights the concept of Unconventional Virtual Computation (UVC). With the increasing virtualization of computers, and the recognition that this year's virtual computers are as fast as the hardware computers of 10 years ago, it becomes clear that we are only limited in our modes of computation by our imagination. A form of UVC is presented called Pulsed Melodic Affective Processing, which utilizes melodies to perform affective computations. PMAP makes computation more human-friendly by making it audible – a PMAP data stream sounds like the emotion it represents.*

A hybrid computation system is presented combining UVC PMAP with a Photonic Quantum Computer, in which the PMAP musico-logic circuit keeps the QC in a state of entanglement.

Keywords: Unconventional Virtual Computing, Human-Computer Interaction, Music, Affective Computing, Quantum Computing, Entanglement

1 INTRODUCTION

The concept of the Virtual Machine has been around for many decades [1]. The best known virtual machines are probably the Java Virtual Machine [2] (there are also virtual machines available for PHP and Tcl), those which allow Apple users to run Microsoft Windows (Parallels and VMware [3]) and Docker for Linux [4]. These virtual machines allow the execution of software which was either not designed for the supporting hardware or is not designed for any particular supporting hardware. Some of the most pervasive virtual systems are the server virtualization used in cloud computing services; this allows one computer to think it is multiple computers, each with their own OS [5]. Some of the implementations of this approach use up server processing in the virtualization process [6], but this is felt to be outweighed by the advantages it brings. One recent and successful example of a research virtual machine is the neural network spread across multiple machines in Stanford's Deep Learning system [7]. The purpose of this virtual machine is to speed up operations on the neural network training at a lower cost.

The virtual machine referred to in this paper is not aimed at reducing computation time. It is more in the philosophy of the Mouse and Graphical User Interface – also known as

Window, Icon, Mouse and Pointer (WIMP). WIMP did not increase processing power on computers when they were added. In fact they reduced it [8] because of the requirements for bitmapped screens and windows and so forth. However there are many tasks now that would be unfeasibly slow for us without WIMP. Furthermore there are some tasks which would be unimaginable without WIMP, for example modern digital photo manipulation. Similarly high level programming languages at first seemed much slower than machine code. However they opened up the world of software to powerful new ways of thinking which could never have been achieved by the machine code programming approach. Changing the mode of Human Computer Interaction opens up opportunities for increasing the usefulness of computers, even though it can on the lowest level slow it down.

Given the growth in virtual computing, unconventional computing has the opportunity to greatly expand its possible modes, limiting computation only by imagination; hence the field of Unconventional Virtual Computation (UVC). There has been a significant amount of work using simulation to run unconventional computation; however these have been designed to simulate a hardware or wetware system [9, 10, 11].

In this paper we will briefly introduce a UVC approach called PMAP and report on an implementation of a hybrid unconventional computation system: a virtual PMAP processor linked to a Photonic Quantum system. The PMAP processor will be used to control the photonic quantum system, driving it towards a maximally entangled output state.

2 EXAMPLE OF UVC: PMAP

One mode of UVC that could provide benefits is Human-Computer Interaction by Replacement (HCI By replacement, or HBR). HBR is an approach to unconventional virtual computing that combines computation with HCI, a complementary approach in which computational efficiency and power are more balanced with understandability to humans. Rather than ones and zeros in a circuit, have the user-interface object itself; e.g. if you want data to be audible, replace the computation basis by melodies. This form of HBR has been reported on previously (Pulsed Melodic Affective Processing - PMAP) [12, 13]. Some forms of HBR may not be implementable in hardware in the foreseeable future, but current hardware speeds could be matched by future virtual HBR machines.

The focus here will be on forms of HBR in affective computation or in computation that has an affective interpretation. It has been shown that affective states (emotions) play a vital role in human cognitive processing and expression [14]. As a result, affective state processing has been incorporated into robotics and multi-agent systems [15]. A further reason in Human-Computer Interaction studies is that emotion may help machines to interact with and model humans more seamlessly and accurately [16]. So representing and simulating affective states is an active area of research.

The dimensional approach to specifying emotional state is one common approach. It utilizes an n-dimensional space made up of emotion "factors". Any emotion can be plotted as some combination of these factors. For example, in many emotional music systems [17] two dimensions are used: Valence and Arousal. In that model, emotions are plotted on a graph with the first dimension being how positive or negative the emotion is (Valence), and the second dimension being how intense the physical arousal of the emotion is (Arousal). For example "Happy" is high valence, high arousal affective state, and "Stressed" is low valence high arousal state.

A number of questionnaire studies provide qualitative evidence for the idea that music communicates emotions [18]. Previous research [19] has suggested that a main indicator of valence is musical key mode. A major key mode implies higher valence, minor key mode implies lower valence. For example the overture of The Marriage of Figaro opera by Mozart is in a major key; whereas Beethoven's melancholic "Moonlight" Sonata movement is in a minor key. It has also been shown that tempo is a prime indicator of arousal, with high tempo indicating higher arousal, and low tempo - low arousal. For example: compare Mozart's fast overture above with Debussy's major key but low tempo opening to "Girl with the Flaxen Hair". The Debussy piano-piece opening has a relaxed feel – i.e. a low arousal despite a high valence.

In PMAP [12, 13] the data stream representing affective state is a stream of pulses. The pulses are transmitted at a variable rate. This can be compared to the variable rate of pulses in biological neural networks in the brain, with such pulse rates being considered as encoding information (in fact neuroscientists have used audio probes to listen to neural spiking for many years [20]). In PMAP this pulse rate specifically encodes a represention of the arousal of an affective state. A higher pulse rate is essentially a series of events at a high tempo (hence high arousal); whereas a lower pulse rate is a series of events at a low tempo (hence low arousal).

Additionally, the PMAP pulses can have variable heights with 10 possible levels. For example 10 different voltage levels for a low level stream, or 10 different integer values for a stream embedded in some sort of data structure. The purpose of pulse height is to represent the valence of an affective state, as follows. Each level represents one of the musical notes C,D,Eb,E,F,G,Ab,A,Bb,B. For example 1mV could be C, 2mV be D, 3mV be Eb, etc. We will simply use integers here to represent the notes (i.e. 1 for C, 2 for D, 3 for Eb, etc). These note values are designed to represent a valence (positivity or negativity of emotion). This is because, in the key of C, pulse streams made up of only the notes C,D,E,F,G,A,B are the notes of the key C major, and so will be heard as having a major key mode – i.e. positive valence. Whereas streams made up of C,D,Eb,F,G,Ab,Bb are the notes of the key C minor, and so will be heard as having a minor key mode – i.e. negative valence.

For example a PMAP stream of say [C,C,Eb,F,D,Eb,F,G,Ab,C] (i.e. [1,1,3,5,3,4,5,6,7]) would be principally negative valence because it is mainly minor key mode. Whereas [C,C,E,F,D,E,F,G,A,C] (i.e. [1,1,4,5,2,4,5,6,8]) would be seen as principally positive valence. And the arousal of the pulse stream would be encoded in the rate at which the pulses were transmitted. So if [1,1,3,5,3,4,5,6,7] was transmitted at a high rate, it would be high arousal and high valence – i.e. a stream representing 'happy'. Whereas if [1,1,4,5,2,4,5,6,8] was transmitted at a low pulse rate then it will be low arousal and low valence – i.e. a stream representing 'sad'.

Note that [1,1,3,5,3,4,5,6,7] and [3,1,3,5,1,7,6,4,5] both represent high valence (i.e. are both major key melodies in C). This ambiguity has a potential extra use. If there are two modules or elements both with the same affective state, the different note groups which make up that state representation can be unique to the object generating them. This allows other objects, and human listeners, to identify where the affective data is coming from.

In terms of functionality PMAP provides a method for the processing of artificial emotions, which is useful in affective computing – for example combining emotional readings for input or output, making decisions based on that data or providing an artificial agent with simulated emotions to improve their computation abilities. It also provides a method for "affectively coloring" non-emotional computation. It is this second functionality which is more directly utilized in this paper. In terms of novelty, PMAP is novel in that it is a data stream which can be listened to, as well as computed with. The affective state is represented by numbers which are analogues of musical features, rather than by a binary stream of 1s and 0s. Previous work on affective computation has been done with normal data carrying techniques – e.g. emotion category index, a real number representing positivity of emotion, etc.

This element of PMAP provides an extra utility – PMAP data can be generated directly from rhythmic data and turn directly into rhythmic data or sound. Thus rhythms such as heart rates, key-press speeds, or time-sliced photon-arrival counts can be directly turned into PMAP data; and PMAP data can be directly turned into music with minimal transformation. This is because PMAP data *is* rhythmic and computations done with PMAP data are computations done with rhythm and pitch. Why is this important? Because PMAP is constructed so that the emotion which a PMAP data stream represents in the computation engine, will be similar to the emotion that a person "listening" to PMAP-equivalent melody would be. So PMAP can be used to calculate "feelings" and the resulting data will "sound like" the feelings calculated. Though as has been mentioned, in this paper the PMAP functionality is more to emotionally color the non-emotional computations being performed.

PMAP has been applied and tested in a number of simulations. As there is no room here to go into detail, these systems and their results will be briefly described. They are [12, 13, 21]:

a. A security team multi-robot system
b. A musical neural network to detect textual emotion
c. A stock market algorithmic trading and analysis approach

The security robot team simulation involved robots with two levels of intelligence: a higher level more advanced cognitive function and a lower level basic affective functionality. The lower level functionality could take over if the higher level ceased to work. A new type of logic gate was designed to use to build the lower level: musical logic gates. PMAP equivalents of AND, OR and NOT were defined, inspired by Fuzzy Logic.

The PMAP versions of these are respectively: MAND, MOR and MNOT (pronounced "emm-not"), MAND, and MOR. So for a given stream, a PMAP segment of data can be summarized as $m_i = [k_i, t_i]$ with key-value k_i and tempo-value t_i. The definitions of the musical gates are (for two streams m_1 and m_2):

$$\text{MNOT}(m) = [-k, 1-t] \quad (1)$$

$$m_1 \text{ MAND } m_2 = [min(k_1,k_2), min\ (t_1,t_2)] \quad (2)$$

$$m_1 \text{ MOR } m_2 = [max\ (k_1,k_2), max\ (t_1,t_2)] \quad (3)$$

It was shown that using a circuit of such gates, PMAP could provide basic fuzzy search and destroy functionality for an affective robot team. It was also found that the state of a three robot team was human audible by tapping in to parts of the PMAP processing stream.

As well as designing musical logic gates, a form of musical artificial neuron was defined. A simple two layer PMAP neural network was implemented using the MATLAB MIDI toolbox. The network was trained by gradient descent to recognise when a piece of text was happy and when it was sad. The tune output by the network exhibited a tendency towards "sad" music features for sad text, and "happy" music features for happy text. The stock market algorithmic trading and analysis system involved defining a generative affective melody for a stock market based on its trading imbalance and trading rate. This affective melody was then used as input for a PMAP algorithmic trading system. The system was shown to make better profits than random in a simulated stock market.

3 PHOTONIC QUANTUM COMPUTING

The quantum computer set up utilized here exactly simulates a quantum CNOT gate. The CNOT gate acts on two quantum bits (qubits). A qubit is the quantum-mechanical analog of a classical bit. A CNOT gate flips ($|0\rangle \leftrightarrow |1\rangle$) the state of the target qubit if and only if the state of the control qubit is $|1\rangle$. Various physical platforms for quantum computing have been proposed, including ion traps [22] and superconducting qubits [23]. Here we consider a photonic quantum computer [24] - a scheme for efficient quantum computation with linear optics.in which information is represented in the quantum state of optical-frequency photons.

In the hardware photons are obtained by focusing a 404nm laser on to a piece of nonlinear crystal (Bisumuth Borate). This causes the crystal to probabilistically spit out 808nm photon pairs, in a process known as Type I spontaneous parametric down conversion. The chip, which performs several experiments that would each ordinarily be carried out on an optical bench the size of a large dining table, is 70 mm by 3 mm. It consists of a network of tiny channels which guide, manipulate and interact single photons. Waveguides are made with a higher refractive index than their surroundings, so that photons can propagate along them by total internal reflection. The waveguides in the integrated optical device are made from silica and sit in a wafer of silicon, which allows things to be kept on a relatively small scale – the chip is 70mm x 3mm.

Using eight reconfigurable electrodes embedded in the circuit, photon pairs can be manipulated. A schematic is shown in Figure 1. The circles with numbers in them are known as phase shifters – and will be discussed later. They are able to change the phase of the photons. The points where the lines meet are called beam splitters, which will be explained later and also enable further quantum effects to be added to the calculation.

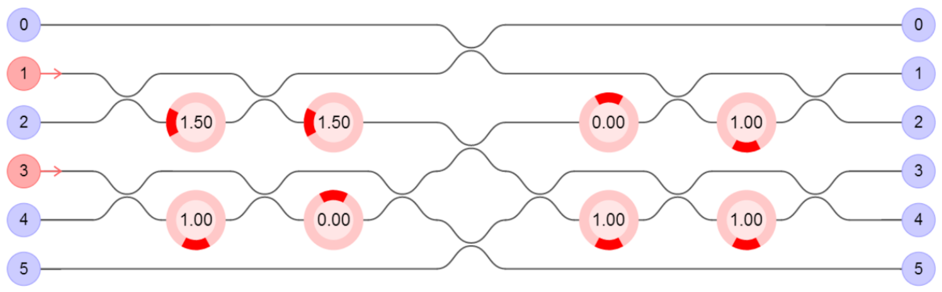

Figure 1: Schematic of the Photon Quantum Computer showing photons being input on (1) and (3) and various phase shifter settings in the pathways.

Control	Target	Inputs to send photon in to
0	0	1, 3
0	1	1, 4
1	0	2, 3
1	1	2, 4

Table 1: Setting up inputs on the quantum C-NOT

The key elements are the inputs marked 1 to 4 in Fig. 1. In this C-NOT the inputs are each represented by two photons. These allow the inputs of the quantum C-NOT to be specified, as shown in Table 1.

The hardware and simulation systems are located at the University of Bristol and can be accessed with only a few seconds lag over the internet. A JSON web API is provided which gives full access to the CNOT. It can use any modern programming language (Mathematica, Python, Javascript, MATLAB ...) to talk to the Bristol servers through this API and get data. Below is an example API call, getting counts from the chip with all phases set to zero (i.e. the circles with floating point numbers in Figure 1 all set to 0). This is the Python code to make the call:

```
counts = urllib2.urlopen("http://cnotmz.appspot.com/experiment?
phases=0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0&accessToken=XXXXXXXXXXXXXXX").read()
```

The above calls will return data in the form:

```
{"counts": {"2,3": 0, "1,3": 80, "1,4": 0, "2,4": 28}, "max": 80, "sum": 108}
```

which gives the number of photons detected across the output groups at the far right of Figure 1. It can be seen how these relate to qubit values using Table 1. In this example outputs 1 and 3 had a count of 80 simultaneous photons in the time segment. Changing the phase of the photons causes them to interfere destructively or constructively with each other.

We will now give a brief introduction to how photonic quantum computers function, as they are different to some of the traditional forms of quantum computing. Consider a subset of the sort of paths contained in the chip, as shown in Figure 2. The far left shows the inputs for the qubit – putting in a photon into (0) gives a qubit of value 0, an input in (1) gives a qubit of value 1. In the centre is a beam splitter which splits the photon, and at the far right are the photon detectors which count the number of photons arriving in each path.

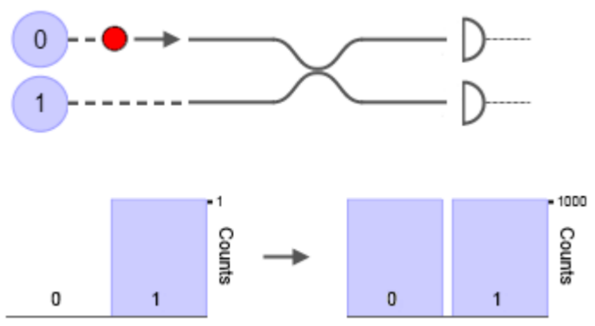

Figure 2: A simplified photon path and the result

If a single photon is put in through (0) or (1) 2000 times, then we would expect to detect the photon half the time at the top detector and half the time at the bottom detector. Between the beam splitter and the detectors, the photon is in what is known as a superposition state, it is “blurred” across paths 0 and 1. Adding another beam splitter gives Figure 3. If a photon is sent into (0) then as a result of the extra beam splitter is will always be detected at the lower detector for the following reason. At the first beam splitter it blurs across both paths, and at the second beam splitter these blurred paths interfere with each other behaving like light waves. This interference causes the probability of the particle being detected at the top detector to become zero. Thus the particle is always detected at the bottom detector. Technically this interference is happening to the spatial wave function.

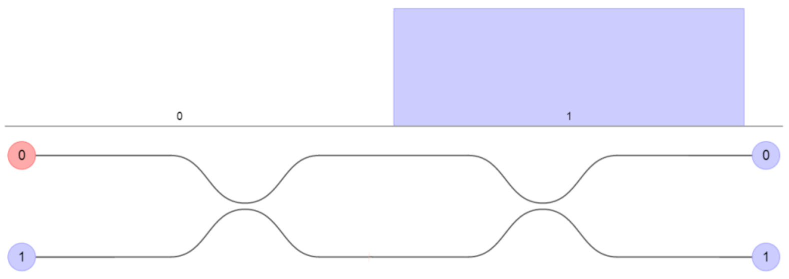

Figure 3: Photon system with an additional beam splitter.

This interference effect can thus be manipulated using the phase shifters in the wave guides. Figures 4 and 5 show what happens when a phase shifter is added. Figure 4 applies a phase shift of 0.5π radians to the “part” of the blurred photon in that wave guide (hence the number 0.5 in the circle). This causes interference effects at the second wave guide leading to photon detection happening at top and bottom detectors with equal probability.

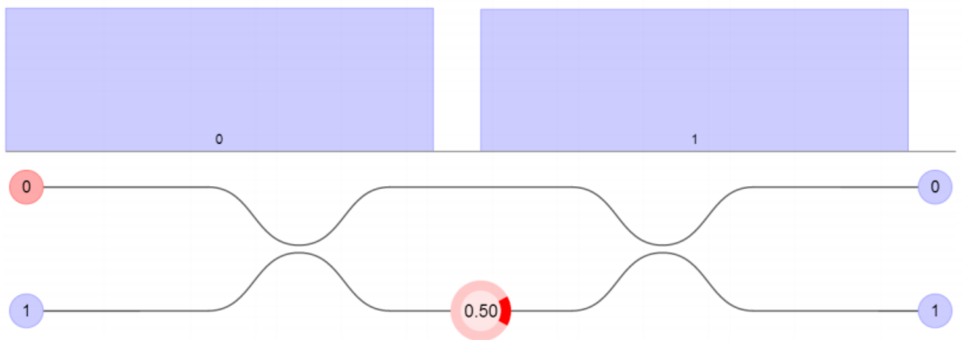

Figure 4: Photon system with an additional beam splitter.

The phase shifter in Figure 5 is set to 1π radians (hence the value 1 in the circle). This creates an interference effect in the second beam splitter that leads to the waves cancelling out for the bottom detector. So the photon will always be detected at 0. Applying different phase shifts causes different probabilities of detecting the photon at different detectors. The demonstration of these interferences is a mathematical task which – although not highly advanced – would require lengthy mathematical expansions – thus they will not be shown in this paper.

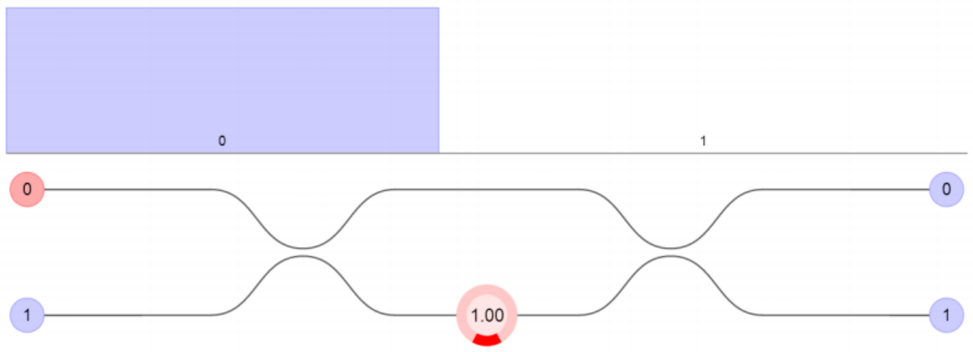

Figure 5: Photon system with an additional beam splitter.

However looking at Figure 1, and the brief description of the JSON Web API earlier, it can be seen that the phases can be set in various paths dynamically and photon counts returned, over the internet.

4 BELL'S INEQUALITY AND ENTANGLEMENT

The phenomenon of entanglement is at the heart of quantum computing, and relates to instantaneous statistical correlations between measurements, even when they are physically separated and have no causal connection. One methodology used to quantify incidences of entanglement is by Bells Inequality [26]. What will be partially explained here is the CHSH inequality [27], a more practical form of Bell's ground-breaking work. The CHSH methodology defines four methods of simultaneously measuring photon events across the detectors 1 to 4 in Figure 1: A, A', B and B'. CHSH says that if reality is local then for the two measurements with settings A, A' and B, B':

$$CHSH = E(A, B) + E(A, B') + E(A', B) - E(A', B') <= 2 \tag{4}$$

where E is the quantum correlation, defined for the qubits in Table 1 as:

$$E = (N_{00}-N_{01}-N_{10}+N_{11})/ N_{Total} \tag{5}$$

where N is the count at the detectors of the detected qubits. To investigate this with the photonic quantum chip, the four measurement configurations in Equation 4 are activated by setting the last four phase shifters on the right hand side of the schematic in Figure 1. A and A' are the two settings for the top two, B and B' the settings for the bottom two. The quantum correlation in equation 5 is then given by:

$$E = 9[P(1, 3)+P(2, 4)-P(1, 4)-P(2, 3)] \qquad (6)$$

where:

$$P(x, y) = N(x, y) / [N(1, 3)+N(2, 4)+N(1, 4)+N(2, 3)] \qquad (7)$$

Where $N(x,y)$ is the number of coincident photons counted at x and y detectors in Figure 1, in the same time segment. The reason for multiplying by 9 is “post-selection”. The chip has 9 more output states that the qubit states, so we throw these away and multiply up the output states of interest.

Most phase values for A, B, A’ and B’ will lead to the CHSH value in equation 4 being less than or equal to 2, thus satisfying local realism. However the following settings violate classical local reality (in other words lead to instantaneous correlations between physically separated and non-communicating qubits). First set the left hand side phases as shown in figure 6: i.e. to 1.5, 1.0, 1.5 and 0.0 (i.e. 1.5π, π, 1.5π and 0 radians).

Then for A set the top two phase shifters on the right hand side to 0.5 and 1.75, for A’ set them to 0.5 and 1.25. For B set the lower two on the left hand side to 0.5 and 1.5, and for B’ to 1.0 and 1.0. The settings for [A,B], [A,B’], [A’,B] and [A’,B’] are shown in Figures 7 to 10 respectively. Having set these up, if you actually run sufficient experiments on the photonic computer and take the average value of the CHSH, it will be closer to 2.6 than to 2. The theoretical limit can be shown to be 2*sqrt(2).

In fact these detector settings give the maximally entangled states for the photons i.e. qubits. Figure 11 shows how adjust the phases on the right hand side of the schematic can move the CHSH values between the classical limit and outside the classical limit.

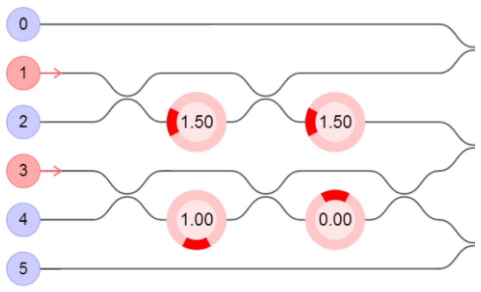

Figure 6: Fixed settings of first four phase shifter.

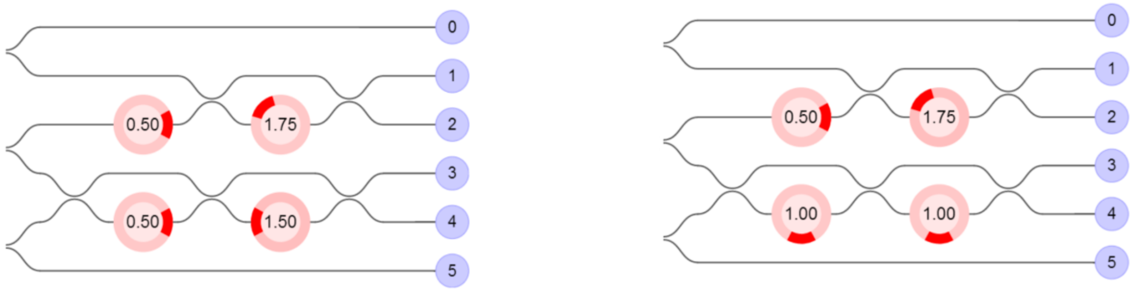

Figure 7 / 8 : Phase shifter settings for E(A,B) and E(A,B’) in equation 4.

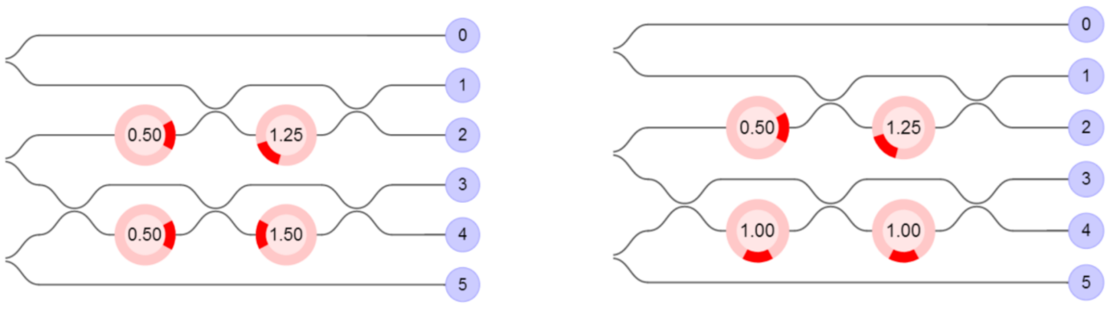

Figure 9 / 10: Phase shifter settings for E(A’,B) and E(A’,B’) in equation 4.

There are a number of simulated quantum computers available on the internet [28]. However the uniqueness of the Bristol Computer is that it is a photonic quantum computer (which fits well with PMAP, as explained in the next section), that the simulator is actually designed - by hardware experiment and theory - to accurately reflect the hardware system, and it uses the same internet calling structure as the Bristol hardware computer:

counts = urllib2.urlopen("http://cnotmz.appspot.com/simulate?

chipName=cnot_mz&quantumClassical=quantum&noiseMode=false&sortMode=false&inp uts=0,1&phases=0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0 ").read()

returning in the form:

{"counts": {"2,3": 0, "1,3": 80, "1,4": 0, "2,4": 28}, "max": 80, "sum": 108}

Thus by designing the hybrid system using their online simulator, then once the hardware system has been re-commissioned in mid-2015, the hybrid code can simply have its call structure replaced and it will become a PMAP / Hardware QC Hybrid. The hybrid set-up will now be described.

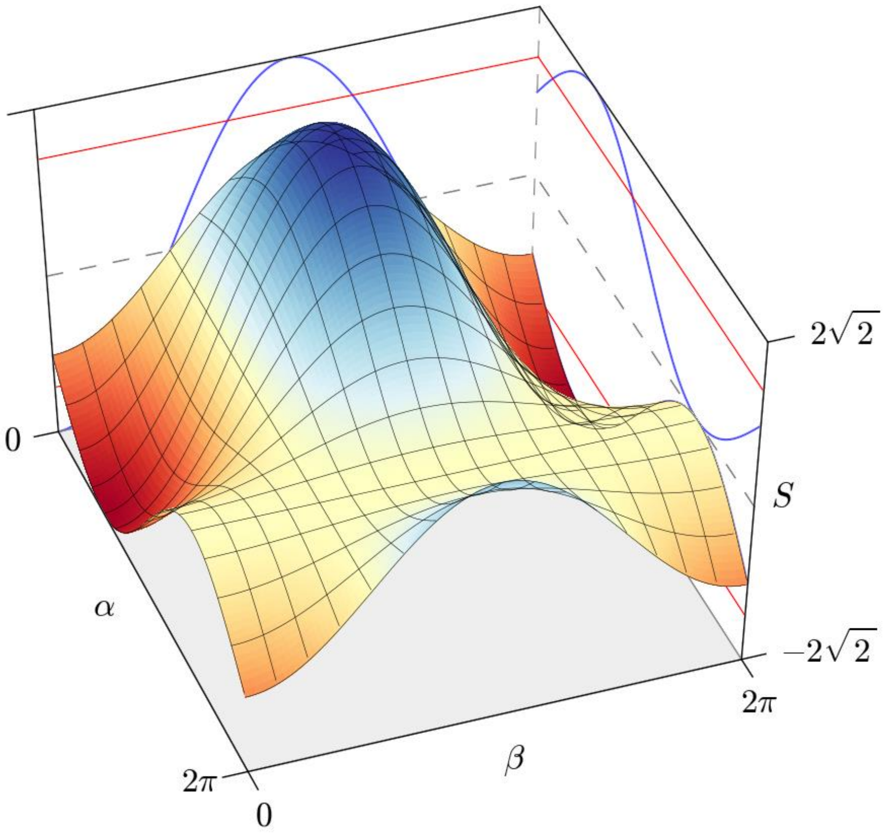

Figure 11: How phase changes effect the Bell CHSH number calculated

5 HYBRID UVC AND QC

5.1 Design

It has been shown previously that the Quantum Computer in the Cloud can achieve entanglement in the CNOT gate [25] and also that PMAP can be used to control non-musical adaptive processes [12, 13, 21]. The interest here is to examine how the two can be combined to create a form of Hyrbid Computer which is part Unconventional Virtual Computation and part Unconventional/Quantum Computation. It is particularly of interest utilizing PMAP and a photonic QC. Being essentially spike-based, PMAP is particularly suited to dealing with data which is rhythmic and has a tempo. Simultaneous photon arrival counts have some of these properties. To examine this link in the first place, the simplest possible process will be created. A PMAP circuit will be designed which attempts to move the QC towards entanglement and keep it there. Although this in itself has no explicit computational usage, it would be a demonstration of the ability of UVC (PMAP) to interface with work with a quantum computer. The circuit in Figure 12 is the basis for this. The design process behind this circuit will be explained.

The input of the QC is held at simultaneous photons on 1,3. The output counts of photon simultaneous arrivals are sampled every second and these become the counts S_{13}, S_{23}, S_{14}, and S_{24} seen in Figure 12. These arrival rates are converted by a simple linear transform into tempos for a PMAP stream. So the higher the output count, the higher the music tempo. The pitches of the PMAP streams do not vary in this calculation, and so are simply just a repeating figure consisting of middle C and middle E. So S13, S23, S14, and S24 will be PMAP streams of tempos proportional to C13, C23, C14, and C24 respectively, each with a pitch form [C,E,C,E,C,E, …] or [1, 4, 1, 4, 1, 4, …].

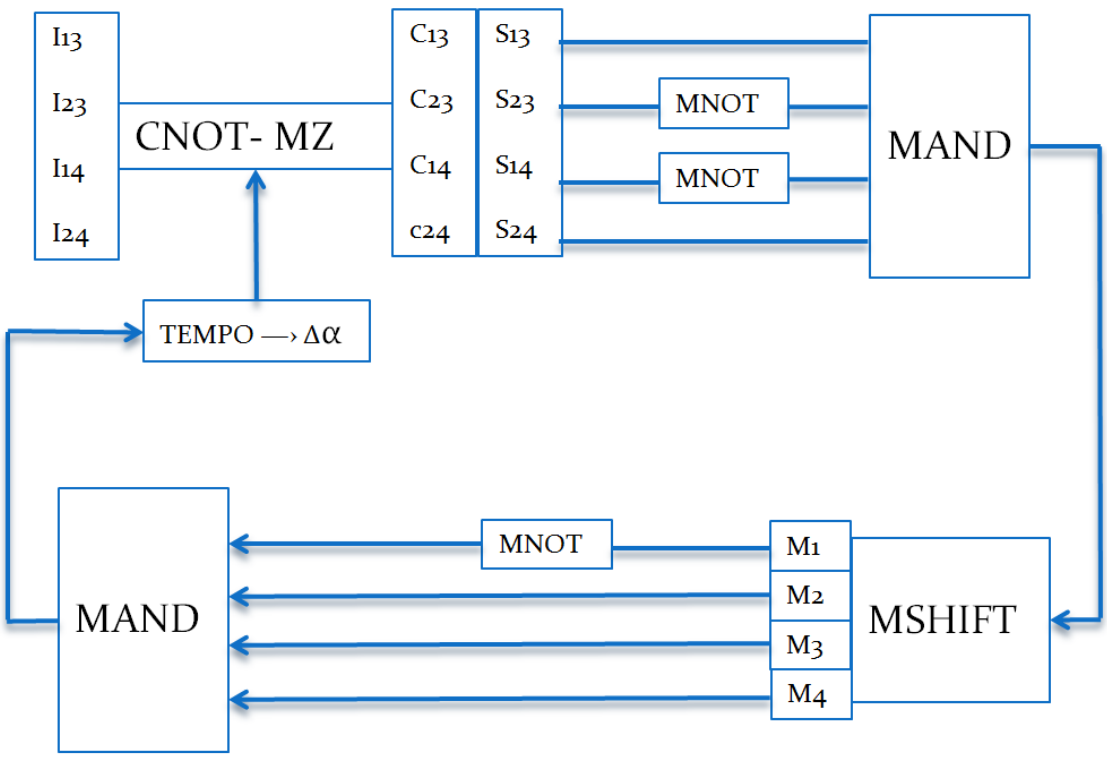

Figure 12: Circuit which is the basis for an example PMAP / QC hybrid process

Looking at equation 6, the combination of counts is replaced by musical ANDs (MANDs) done on the four PMAP streams. The negative signs in the combination are replaced by a musical NOT (MNOT) on the negative input of the MAND gate. Thus equation 6 becomes:

$$E = MAND(S_{13},S_{24}, MNOT(S_{14}), MNOT(S_{23})] \qquad (8)$$

which is what is implemented in the top part of Figure 12. However then the calculation of the entanglement measure requires four such calculations like equation (8) to be performed with the different phase shifter settings in Figures 7-10.

The MSHIFT object is simply a musical shift register. So at each of the four calculation step it stores an output from the MAND gate being input to it. This is synchronized with the phase change process. So for the each of the four phase value sets required to calculate the Bell CHSH number (Figures 7 to 10) the melody from equation 8 is stored in the MSHIFT register. Equation 4 (for calculating Bell CHSH) is approximated using the MAND for combination, and the MNOT for negative values, giving equation 9.

$$\textit{CHSH-PMAP} = \textit{MAND}(\textit{MNOT}(\textit{MSHIFT1}), \textit{MSHIFT2}, \textit{MSHIFT3}, \textit{MSHIFT3}) \quad (9)$$

This produces a PMAP output whose tempo is then used in the "Tempo $\rightarrow$ $\Delta\alpha$" object to calculate a phase shifter setting for each of the four sub-experiments. The phase shifter which is adjusted by the PMAP circuit is that seen in the top left of Figures 7-10. This corresponds to α in Figure 11. It can be seen the optimal value is 0.5 (i.e. 0.5π) to maximize entanglement. So each time CHSH-PMAP is calculated, it will result in a melody whose tempo is converted into a change delta as follows. If the tempo of the current CHSH-PMAP stream is greater than the tempo of the previous CHSH-PMAP stream, then the delta is left unchanged. *If the tempo is less than the previous tempo*, then the delta is adjusted as in equation 10.

$$\textit{delta} \rightarrow -0.5\textit{delta} \quad (10)$$

Then at each iteration we have:

$$\alpha \rightarrow \alpha + \textit{delta} \quad (11)$$

If the PMAP circuit based on equation 9 is somehow representing the CHSH calculation in equation 4, then increases in tempo should be correlated to increases in entanglement and vice-versa. Thus the effect of equations 10 and 11 should be to cause α to converge to the point of maximum entanglement, i.e 0.5π.

To explain this more clearly. Suppose that α is set to 0.6π and *delta* is set to 0.1, and the first calculation (where α = 0.6π) gives a tempo of T1 from equation 9. Then the delta is applied to give α = 0.7π, i.e. (0.6+*delta*)π. Suppose that gives a tempo of T2 from equation 9. If T1 < T2, i.e. if the tempo is increasing, then the next value of α will be α = 0.8π. But if T2 < T1, i.e. if the tempo decreases, then *delta* = -0.5**delta*, i.e. *delta* = -0.5*0.1 = -0.05. So the next value of the phase shifter will be α = (0.7-0.05)π = 0.65. Then if the next tempo T3 is such that T3 > T2, *delta* will remain unchanged, so the next value will be α = (0.65-0.05)π = 0.6. This is the algorithm that is being implemented by the "Tempo → Δα" object in Figure 12. It is claimed that the combination of this object, and the PMAP and the QC system are sufficient to move the qubits to close to maximum entanglement, and to keep them there.

5.2 Results

If the PMAP circuitry consistently approximates the Bell CHSH calculation, then the system will clearly always converge – as it is using a form of gradient descent with decreasing gradient; and Figure 11 shows there are no local maxima. The maximum that the simulation can reach is a Bell Value of approximately 2.54. It was indeed found that the QC / PMAP system always converges to near a Bell CHSH value of 2.54, with the α values close to 0.5 + 2.N for N = 0,1,2,…, because the maxima in Figure 11 actually occur at α = (0.5 + 2.N) π. Thirty examples were run, each starting at a random α between 0 and 2π, and with a random initial delta between 0 and 1. Examples of 4 such runs are shown in Figure 13.

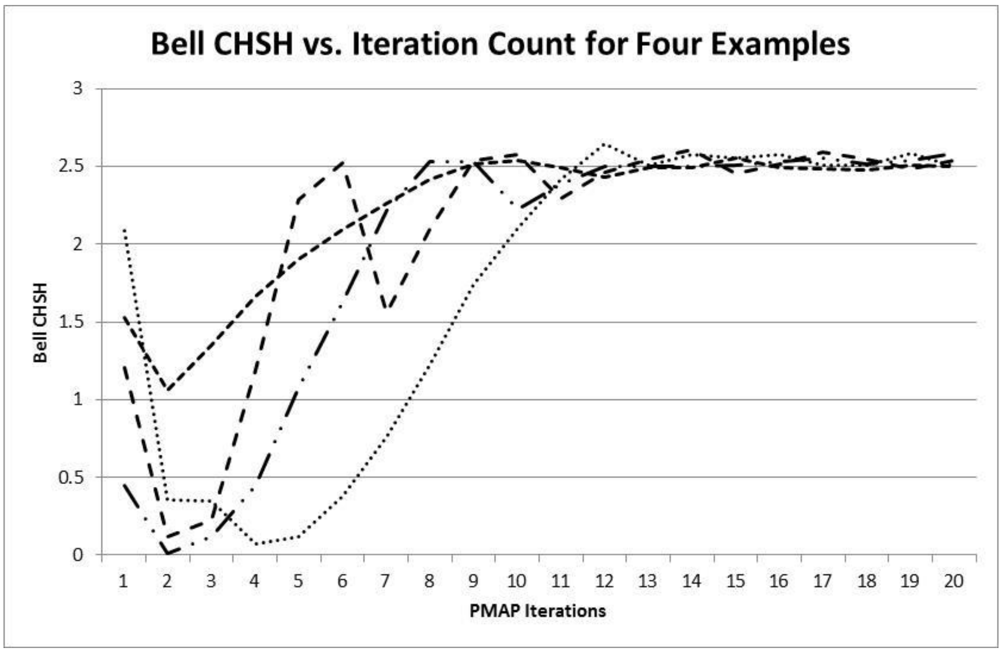

Figure 13: Four examples of Bell CHSH convergence using PMAP

It can be seen in Figure 13 that the examples are clearly converging to a value just over 2.5 (the maximum possible being approximately 2.54). This convergence is driven by the changing tempo on the output of the PMAP circuit (the output of the second MAND in Figure 12). The tempos for these four examples can be seen in Figure 14. The second MAND output for eight iterations of an actual convergence example can be heard here: https://soundcloud.com/alexiskirke/quantum-pmap-convergence-example

Looking at twenty eight of the thirty example runs, average errors during convergence are shown in Figure 15. The reason only 28 are included is that two of the runs became stuck in local maxima. Run 12 stuck at Bell CHSH value between 2.4 and 2.5, corresponding to phase alpha of around 0.6π rather than 0.5π. So this was still close to maximum entanglement. However run 13 became stuck in a local maximum a long way from the quantum region: a Bell CHSH of 0.7 which corresponded to a phase α of around 1.84π.

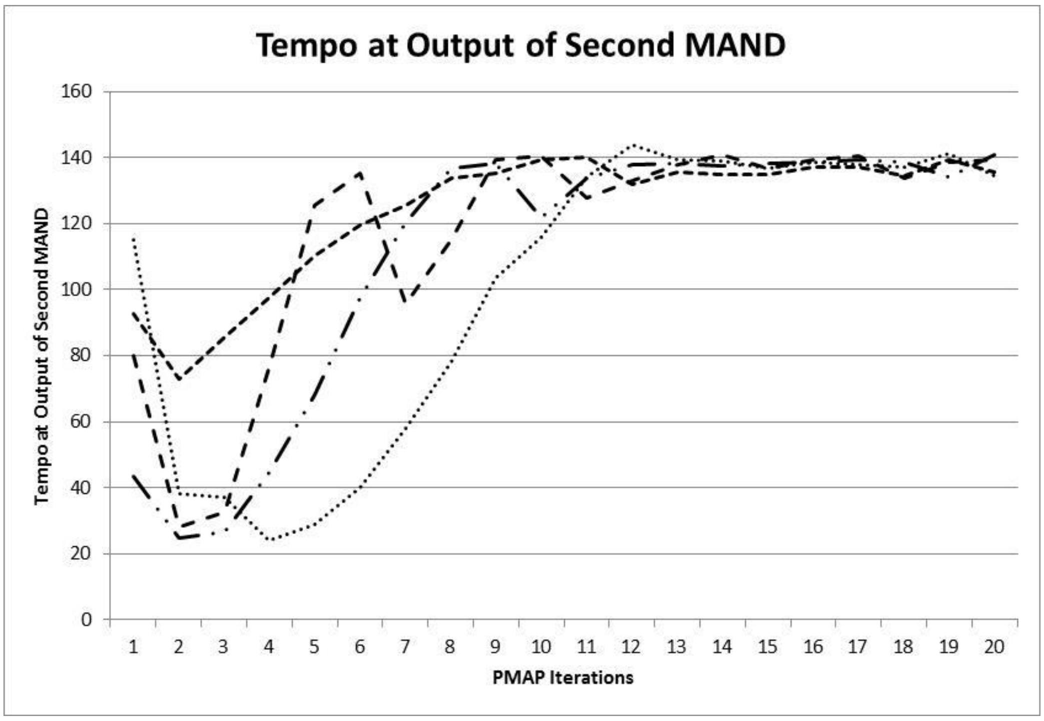

Figure 14: The four examples of Bell CHSH convergence using PMAP from Figure 13, shown with their tempo outputs on the second MAND in Figure 12.

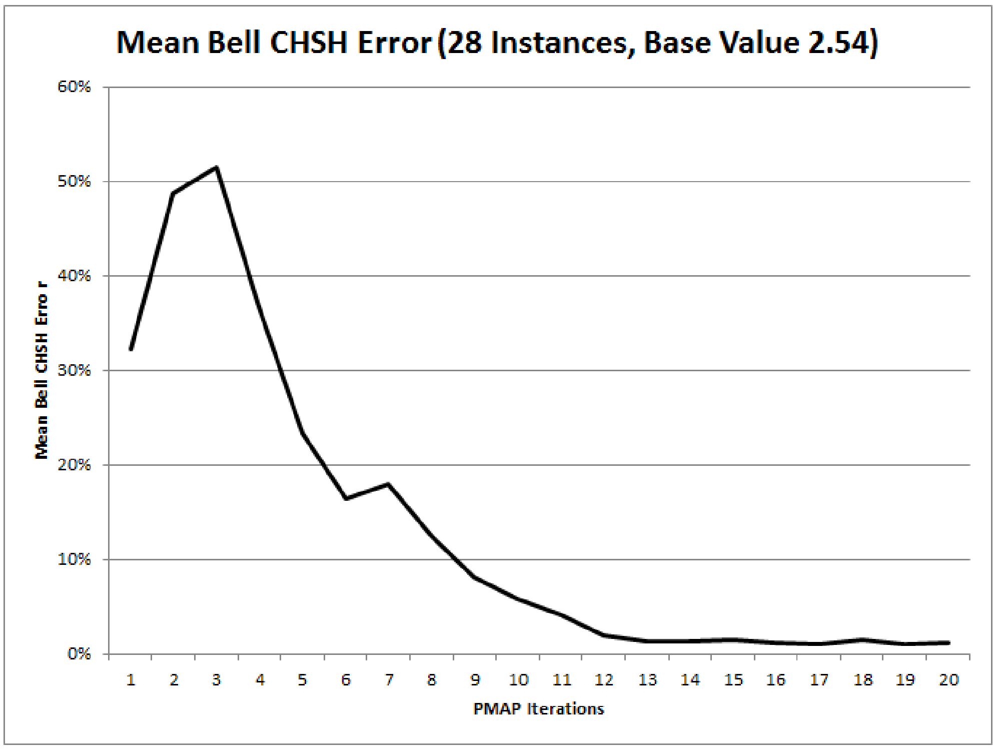

Figure 15: The mean error during convergence across 28 examples run, across twenty iterations.

These local maxima in the PMAP system, which are not present in the quantum computer, are not surprising, given how different the modalities are in the calculation: quantum versus PMAP. What is surprising is that only two of them were found in 30 runs, and that only one of them actually "broke" the functionality of the system. Even with the outliers included, the average Bell CHSH value across all 30 runs after 20 iterations was 2.45 and standard deviation 0.34, with most of the standard deviation coming from a single outlier.

A form of attractor diagram is show in Figure 16 – plotting the average phase alpha (divided by 2π) during convergence against the mean tempo on the output of the PMAP circuit – for the 23 examples that converged to 0.5π. The average starting point is around 0.7 and all then converge to an α of around 0.5π at a tempo just under 140. For five of the examples, the starting point and delta meant that it converged to 2.5π rather 0.5π, which is also the same maxima of the Bell CHSH curve. These are in shown Figure 17.

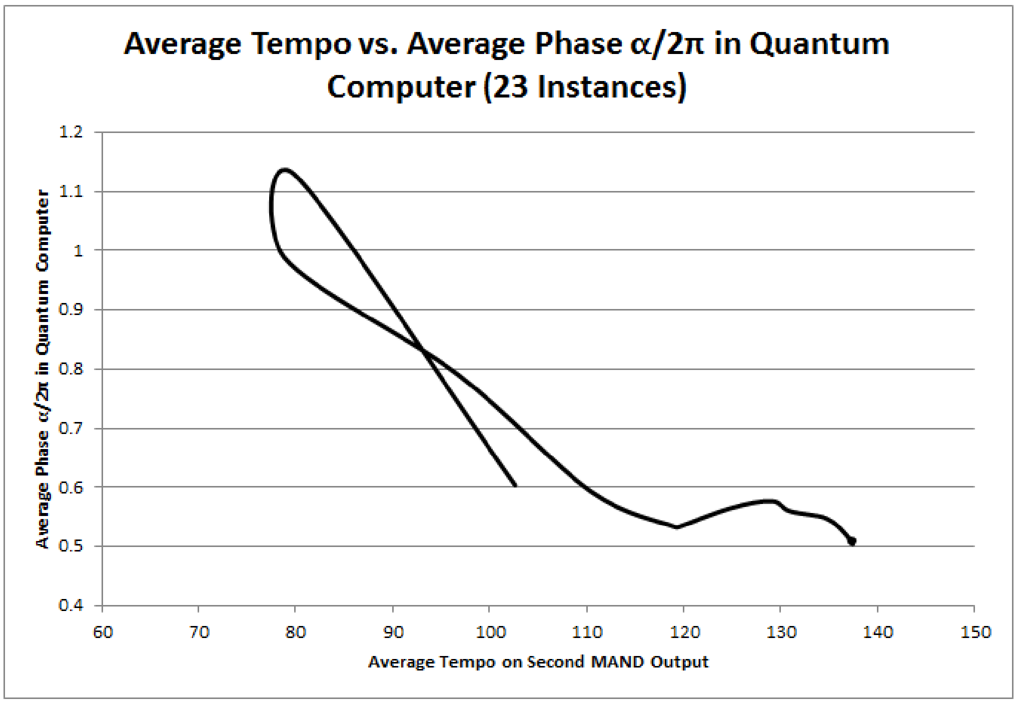

Figure 16: Mean Tempo vs. Mean Phase α/2π for 23 of the 30 examples

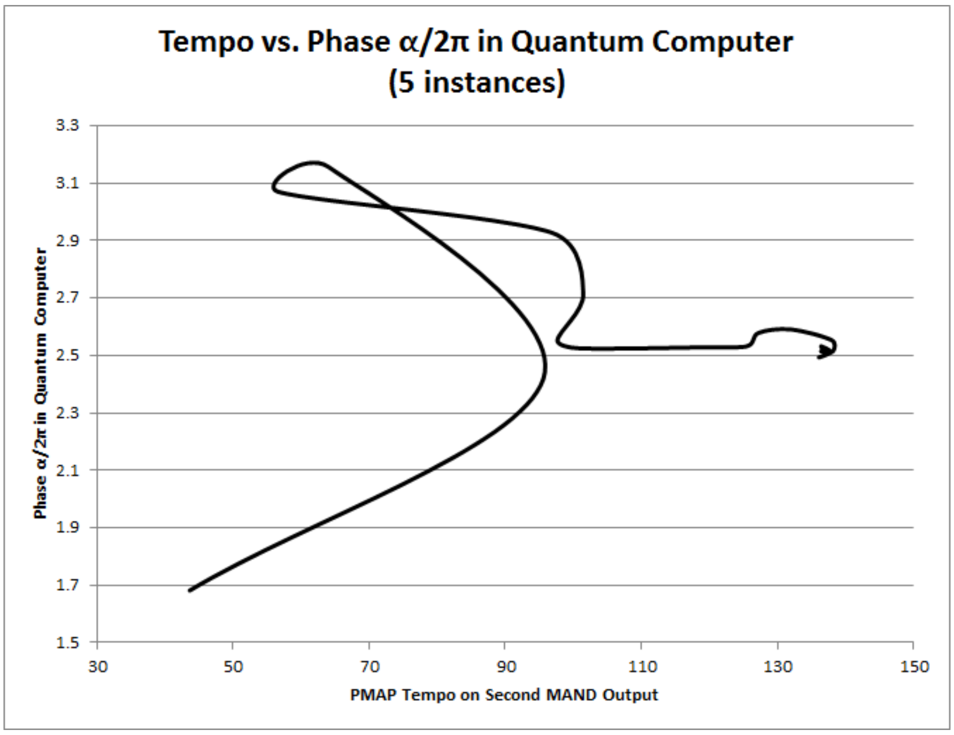

Figure 17: Tempo vs. Phase α/2π for 5 Examples which converge towards 2.5

Given the non-triviality of the Bell CHSH calculation, it is unlikely that the PMAP circuit is causing the convergence by chance. However to further examine this, various elements in the PMAP circuit were adjusted to confirm the system did not converge for any similar PMAP circuit. Firstly the MNOT after the MSHIFT register from Figure 12 was removed. This was then replaced, and the first MAND was changed to a MOR. Ten runs were done for each of these conditions. For the first the average final Bell value after 20 iterations was 0.85 with standard deviation 0.72. For the second it was average value 0.94 with standard deviation 0.89. This compares unfavourably with the 30 runs with the Figure 12 set-up – the average bell value (including the outlier) was 2.45 and standard deviation 0.34.

In terms of PMAP's originally envisioned functionality – giving insight into the computation process – do the PMAP melodies give insight into what is occurring? Suppose two virtual "probes" are placed into the circuit in Figure 12 at the output of the first MAND

and the output of the second MAND gate. At maximum bell value achievable – 2.538 – the tempo on the output of the first MAND is 142, and the tempo at the output of the second MAND is also 142. Away from the maximum bell values these tempos are different. Also the further away the PMAP system is from entangling the photons the lower those tempos are. This means that listening to the PMAP data at these two points can have three effects:

1. It gives an insight into the process development – the closer the system comes to entanglement, the more in synch the two PMAP streams will sound.
2. It gives an emotional insight into what is occurring: higher tempo in a major key communicates higher "happiness" [18]. Thus the closer the system gets to fulfilling its aim (maximum entanglement of qubits) the happier it sounds. This fulfills one of PMAPs aims to give an emotional insight into calculations.
3. It gives an insight into the process itself. A non-expert might observe that for the system to achieve entanglement requires the outputs of the two MANDs to be higher and more similar tempos. So it might start the non-expert thinking along the lines of what photon output counts would lead to this tempo. This provides another model of considering the nature of the entanglement equations which may be more understandable to an individual unfamiliar with quantum computing.

Although this particular PMAP process is not a practical one – we know how to entangle qubits – it gives a demonstration of how more useful processes could also be given greater transparency thanks to unconventional virtual computing, as well as how UVC can be combined with other forms of computation to give consistent functionality.

6 CONCLUSIONS

The purpose of this paper has been to introduce the concept of Unconventional Virtual Computing (UVC). In particular a form of UVC has been introduced called PMAP. It has been applied in a simple hybrid system involving a UVC and a simulated photonic quantum computer. The UVC successfully kept the quantum computer qubits in a state of entanglement using gradient descent. It also showed how UVC can give insight into the computation going on, in novel ways.

It is interesting to note that this not the first time sound has been used in quantum computing: researchers at the University of Bristol Centre for Quantum Photonics have used a system involving photo-diodes, a tone generator and a loudspeaker to detect the location of Hong-Ou-Mandel dips [29] in hardware photon beam splitters. However – and putting aside any contributions to UVC - to build a music-based system which can interface with real single photons, real quantum systems which we can't see and which are actually entangled, is to our knowledge a new contribution.

This paper has been written from the point of view that with the increasing virtualization of computers, and the recognition that this year's virtual computers are as fast as the hardware computers of 10 years ago, it is becoming clear that we are only limited in our modes of computation by our imagination. Given that improvements in computer efficiency are not always due to increasing computation speed, UVC has the potential for speeding up working with computers by making their processes more human-understandable.

References

1. Goldberg, R. (1974) Survey of Virtual Machine Research, Computing, June, 34-45
2. Freund,S, Mitchell, J. (1999). A formal framework for the Java bytecode language and verifier. In Proceedings of the 14th ACM SIGPLAN conference on Object-

oriented programming, systems, languages, and applications (OOPSLA '99), Berman, A. (Ed.). ACM, New York, NY, USA, 147–166.

3. VMWare Inc. (2007) Understanding Full Virtualization, Paravirtualization, and Hardware Assist, Retrieved April 2015, from http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf
4. Seo, K., Hwang, H., Moon, I., Kwon, O., Kim, B. (2014) Performance Comparison Analysis of Linux Container and Virtual Machine for Building Cloud, Advanced Science and Technology Letters 66:105-111.
5. Guohui W. Ng, T. (2010) The Impact of Virtualization on Network Performance of Amazon EC2 Data Center, 2010 Proceedings IEEE INFOCOM, , 14-19 March 2010, pp.1-9.
6. Huber, N., von Quast, M. Et al. (2011) Evaluating and Modeling Virtualization Performance Overhead for Cloud Environments. In Proceedings of the 1st International Conference on Cloud Computing and Services Science (CLOSER 2011), Noordwijkerhout, The Netherlands, May 7-9, SciTePress. May 2011, pp. 563 - 573.
7. Coates, A., Huval, B. et al. (2013) Deep learning with COTS HPC Systems . Stanford, CA: Stanford University Computer Science Dept.
8. Garmon, J. (2010) What were the original system requirements for Windows 1.0? Accessed April 2015, From http://www.jaygarmon.net/2010/11/what-were-original-system-requirements.html
9. Jones, J. (2010) The Emergence and Dynamical Evolution of Complex Transport Networks from Simple Low-Level Behaviours, International Journal of Unconventional Computing 6(2):125-144.

10. Bull, L., Budd, A., Stone, C., Uroukov, I., Costello, B. d. L. and Adamatzky, A. (2008) Towards unconventional computing through simulated evolution: Control of nonlinear media by a learning classifier system. Artificial Life, 14 (2). pp. 203-222.
11. Spector, Lee, et al. "Finding a better-than-classical quantum AND/OR algorithm using genetic programming." Proceedings of the Congress on Evolutionary Computation. Vol. 3. 1999.
12. Kirke, A., Miranda, E. (2014) Towards Harmonic Extensions of Pulsed Melodic Affective Processing - Further Musical Structures for Increasing Transparency in Emotional Computation. International Journal of Unconventional Computation, 10(3): pp. 199-217.
13. Kirke, A., Miranda, E.R. (2014) Pulsed Melodic Affective Processing: Musical structures for increasing transparency in emotional computation. Simulation, 90(5): pp. 606-622.
14. Malatesa, L., Karpouzis, K. et al. (2009). Affective intelligence: the human face of AI, In Artificial intelligence, Springer-Verlag.
15. Banik, S., Watanabe, K. et al. (2008) Affection Based Multi-robot Team Work, In Lecture Notes in Electrical Engineering, pp. 355--375.
16. Picard, R. (2003). Affective Computing: Challenges, International Journal of Human-Computer Studies, Vol. 59, No. 1-2, pp. 55-64.
17. Kirke, A., Miranda, E. (2015). A Multi-Agent Emotional Society Whose Melodies Represent its Emergent Social Hierarchy and Are Generated by Agent Communications, Journal of Artificial Societies and Social Simulation, 18(2).
18. Juslin, P., Laukka, P. (2004). Expression, perception, and induction of musical emotion: a review and a questionnaire study of everyday listening. Journal of New Music Research, vol. 33, pp. 216-237.

19. Juslin, P. (2005). From Mimesis to Catharsis: expression, perception and induction of emotion in music, In Music Communication, Oxford University Press, pp. 85-116.
20. Chang, M., Wang, G. at al. (2010) Sonification and vizualisation of neural data. Proceedings of the International Conference on Auditory Display, June 9-15, Washington D.C.
21. Kirke, A., Miranda, E. (2012). Pulsed Melodic Processing – the Use of Melodies in Affective Computations for Increased Processing Transparency. In Music and Human-Computer Interaction, S. Holland, K. Wilkie, P. Mulholland and A. Seago (Eds.), London: Springer.
22. Kielpinski, D., Monroe, C., Wineland, D. (2002) Architecture for a large-scale ion-trap quantum computer, Nature 412:709-711.
23. Kelly, J., et al. (2015) State preservation by repetitive error detection in a superconducting quantum circuit, Nature, 519:66-69.
24. Knill, E., Laflamme, R., Milburn G. (2001) A scheme for efficient quantum computation with linear optics, Nature 409:46–52.
25. Shadbolt, P., Verde, M., Peruzzo, A., Politi, A., Laing, A. Lobino, M., Matthews, J., Thompson, M., O'Brien, J. (2012). Generating, manipulating and measuring entanglement and mixture with a reconfigurable photonic circuit, Nature Photonics 6, pp45-49.
26. Shadbolt, P., Vértesi, T., Liang, Y. Branciard, C., Brunner, N., O'Brien, J. (2012) Guaranteed violation of a Bell inequality without aligned reference frames or calibrated devices. Scientific Reports, volume 2(article number 470).
27. Clauser, J., Horne, M., Shimony, A., Holt, R. (1969) Proposed experiment to test local hidden-variable theories. Phys. Rev. Lett. 23 (15), pp880–884.

28. Quantiki Wiki (2015) List of QC Simulators, From http://www.quantiki.org/wiki/List_of_QC_simulators, Accessed April 2015.

29. Hong, C., Ou, Z., Mandel, L. (1987)."Measurement of subpicosecond time intervals between two photons by interference. Phys. Rev. Lett. 59 (18): 2044–2046.