The Quantum Baseline Network

Manish Kumar Shukla, Rahul Ratan, A. Yavuz Oruç
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742
e-mail: [manishk, rahulr, yavuz]@eng.umd.edu

Abstract — **We present the design of a self-routing quantum packet switch which improves on a previous design given by the authors [1]. Like the earlier design, this switch too routes packets represented by quantum bits (qubits) but it reduces the routing overhead per packet from $O(\log N)$ qubits to $O(1)$ qubits by eliminating the need for an extra "dummy" input-output pair and their associated "dummy" packets at each 2×2 internal switch. The quantum self-routing switch creates a superposition of all the maximum size non-blocking subsets of input packets at its outputs, which cannot be achieved by any classical self-routing switching network that is internally blocking. In addition to the network design, we give a method to characterize the output quantum state of the switch using the concepts of frames and balanced matrices.**

I. Introduction

In this paper, we introduce the idea of using quantum gates and quantum bits (qubits) to make efficient interconnection networks with low hardware and routing costs and as an example give a detailed design of the Quantum Baseline Network (QBN). In conventional interconnection network design there is a tradeoff between factors of practical interest like the ability of a network to connect any pair of idle terminals under arbitrary traffic conditions ("non-blockingness"), ease of routing, routing delay, scalability and crosspoint count (also called crosspoint complexity). Many strictly non-blocking $N \times N$ networks are known which can route all possible $N!$ one-to-one maps between their inputs and outputs but either their crosspoint complexity is high [2] [3] [4] [5] [6] or they have high routing cost [7]. Other interconnection fabrics like Delta [8] and Banyan networks [9] designed to have simple decentralized routing and low crosspoint complexity are internally blocking [10]. Intuitively, in an interconnection fabric with low number of crosspoints, blocking occurs because the internal path of one input-output connection may have to pass through the same internal link as that required by another input-output connection, i.e., blocking is a result of output contention amongst packets incident on the same internal switch. We propose using quantum circuits for building efficient interconnection networks which use quantum parallelism to reduce internal blocking and simplify routing while having low crosspoint complexity. Thus, such quantum networks can *potentially* combine the non-blocking functionality of Cantor/Clos-like networks with low routing and hardware costs of self-routing Banyan-like networks. The rapidly emerging research field of quantum computing seeks to take advantage of the inherent parallelism and certain unique properties (for example, entanglement) of quantum systems in providing more efficient solutions to existing problems or enabling solutions not feasible with conventional techniques. By using such quantum concepts, Shor's algorithm [11] finds prime factors of a number in polynomial time (versus exponential complexity for classical algorithms) and Grover's quantum search algorithm [12] searches an unstructured database in $O(\sqrt{N})$ time (versus $O(N)$ classically).

Our quantum networks are made up of quantum gates and operate on packets represented by quantum bits (qubits). Qubits exhibit the unique property of existing in a superposed state - a classical bit can exist in only two states "0" and "1" but a qubit can exist in a superposed state like $1/\sqrt{2}\left(|0\rangle + |1\rangle\right)$. On observation (with respect to the $|0\rangle, |1\rangle$ basis) this qubit's state collapses and we observe either state $|0\rangle$ or state $|1\rangle$ each with probability $\left(1/\sqrt{2}\right)^2 = 1/2$. Until the observation (or measurement) of a qubit, any quantum gate acts simultaneously or in parallel on both the $|0\rangle$ and $|1\rangle$ parts of the state, and this is the basis of quantum parallelism. Our basic idea is to create a superposition of permutations of contending packets at the outputs of internal switches and then route the resulting superposed packets in parallel. In particular, consider the qubit packets P_1 and P_2 input to a 2×2 switch S with outputs O_1 and O_2. Both packets are addressed to a single output, say O_1. Classically, we either buffer or drop one randomly chosen packet and route the other, but using quantum superposition a state of the form $1/\sqrt{2}(|P_1, P_2\rangle + |P_2, P_1\rangle)$ can be created at the outputs of the switch. The two entries in each *ket* ($|\rangle$) correspond to packets at O_1 and O_2 respectively. Both P_1 and P_2 are present at O_1 and a simple measurement in computational basis ($|0\rangle, |1\rangle$) on each qubit in the packets will give either P_1 or P_2 at O_1, each with probability $1/2$ and correspondingly P_2 and P_1 respectively at O_2. Thus, blocking doesn't occur till the measurement is made. One implication of this is that packets can be routed to their destinations without incurring any time overhead to resolve the contentions between them (low routing complexity). Also, the superposition created in this way can be used to route and process the packets in parallel (no blocking).

In our previous paper [1] we took a first step in this direction when we gave the design of a self-routing quantum Banyan interconnection network. In this paper, we build on that approach to design an improved quantum baseline network (QBN) that uses fewer quantum gates and has a lower routing overhead. It should be noted that in the example given above, packets P_1 and P_2 are also present at output O_2, which is not desirable because both packets were addressed to O_1. In [1] we used an extra input-output pair at each 2×2 switch to route "dummy" packets (distinguishable from normal packets) to "unwanted" outputs like O_2 to identify contentions. The distinguishability requirement for "dummy" packets resulted in the number of address qubits in packet headers being doubled to $2\log_2 N$. In this paper, we give a much improved scheme to distinguish "invalid" packets by using just one extra qubit per packet header and without adding any "dummy" input-output pairs. Additionally, we characterize the permutations which can be self-routed through the QBN without internal blocking by using the concept of balanced matrices and frames introduced in [13] and [14] respectively. We then describe a method which determines the distribution of subpermutations generated from a permutation assignment which suffers internal blocking while being self-routed through the QBN.

The rest of the paper is organized as follows. In Section II,

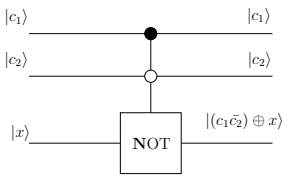


Figure 1: A controlled-controlled NOT (CC-NOT) quantum gate.

we introduce some basic concepts of quantum computing and baseline networks. In Section III, we present the design of a 2× 2 quantum switch that is capable of creating a superposition of its input packets in the case of an output contention. In Section IV, we present the design of a self-routing quantum baseline network (QBN). We discuss the output state of the QBN for permutation assignments in Section V. Section VI contains the concluding remarks.

II. Preliminaries

We give a short introduction to the concepts of quantum computing and self-routing interconnection networks and also give the notation used in the rest of the paper.

II.A Qubits and Superposition

The indivisible unit of classical information is the bit, which can take either one of two values: 0 or 1. The corresponding unit of quantum information is the quantum bit or *qubit*. A qubit's state is a vector in a two dimensional complex Hilbert space. The elements of an orthonormal basis for this space are represented as $|0\rangle$ and $|1\rangle$. A qubit can also exist in a superposition of the '0' and '1' states. In general, a qubit's state can be written as $|x\rangle = a\,|0\rangle + b\,|1\rangle$ where $a, b \in \mathbb{C}$ and $|a|^2 + |b|^2 = 1$. $|x\rangle$ is also represented as $|x\rangle = [a\ b]^T$. On measurement (w.r.t. the above basis) the qubit is observed to be found either in state $|0\rangle$ or in state $|1\rangle$ with probability $|a|^2$ and $|b|^2$ respectively. The state of a system of multiple qubits can be written by taking the tensor product of individual state vectors [15] [16].

II.B Quantum Gates

The state of qubits can be transformed via quantum gates and circuits made using such gates [17]. These gates are unitary transformations (hence are reversible) acting on a fixed number of qubits. Reversibility implies that given the output of a gate, the corresponding input is uniquely determined. A one qubit gate, which is extensively used is the Hadamard gate. The transformation matrix for this gate is

$$\mathbf{H} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{1}$$

It transforms states $|0\rangle$ and $|1\rangle$ as: $|0\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|1\rangle \rightarrow \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. Thus, a Hadamard gate puts a qubit in state $|0\rangle$ or $|1\rangle$ into an equal superposition of $|0\rangle$ and $|1\rangle$. Other quantum gates that are extensively used for manipulating qubits are controlled quantum gates, e.g. a controlled-Hadamard or a controlled-NOT gate [16]. A controlled gate becomes active depending on the state of some control qubits. One such gate, controlled-controlled-NOT (CC-NOT) gate which has two control qubits (c_1 and c_2) is shown in Figure 1. This gate does the following operation

$$|c_1\rangle\ |c_2\rangle\ |x\rangle \xrightarrow{\textbf{CC-NOT}} |c_1\rangle\ |c_2\rangle\ |(c_1.\bar{c_2}) \oplus x\rangle \tag{2}$$

i.e., it inverts x when $c_1 = 1$ (indicated by solid circle) and $c_2 = 0$ (indicated by open circle). This can be extended to quantum gates with multiple control qubits. We will be using such NOT and Hadamard gates with multiple control qubits in our quantum switch designs.

II.C The Switch Gate

The basic building block of the quantum switch is a quantum gate called *switch gate* [1]. This gate is represented in Figure 2. It is a $(2n+1)$–qubit controlled-quantum gate having one control qubit and and two n-size sets of target qubits, each set representing a packet of size n. If the control qubit is set in state $|1\rangle$ then the gate interchanges the two sets of target qubits and if the control qubit is in state $|0\rangle$, it leaves them unchanged. Representing the state of the control qubit by $|c\rangle$, and the states of the two sets of target qubits by strings $|\vec{x}\rangle = |x_1 x_2 \cdots x_n\rangle$ and $|\vec{y}\rangle = |y_1 y_2 \cdots y_n\rangle$ respectively, where $c, x_i, y_i \in \{0, 1\}$, $i = 1 \cdots n$, the function of this gate can be written as

$$|c\rangle|x_1 \cdots x_n\rangle|y_1 \cdots y_n\rangle \xrightarrow{QSG} |c\rangle|u_1 \cdots u_n\rangle|v_1 \cdots v_n\rangle \tag{3}$$

where $u_i = \bar{c}x_i + cy_i$ and $v_i = \bar{c}y_i + cx_i$. It can be easily verified that the gate performs following functions depending on the state of the control qubit

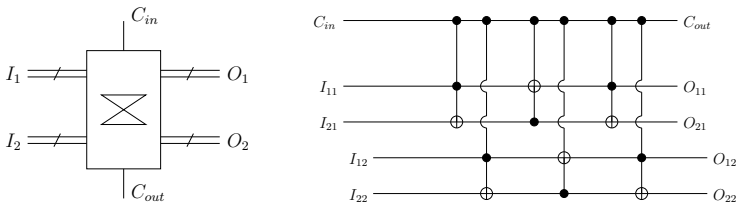
$$|0\rangle|\vec{x}\rangle|\vec{y}\rangle \longrightarrow |0\rangle|\vec{x}\rangle|\vec{y}\rangle,\ |1\rangle|\vec{x}\rangle|\vec{y}\rangle \longrightarrow |1\rangle|\vec{y}\rangle|\vec{x}\rangle \tag{4}$$

We use the switch gate to superpose the packets that contend for one output of a 2×2-switch and to route the superposition on that output. For example, if the control qubit of the gate is set in an equal superposition of states $|0\rangle$ and $|1\rangle$ then the action of the gate is $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|\vec{x}\rangle|\vec{y}\rangle \longrightarrow \frac{1}{\sqrt{2}}(|0\rangle|\vec{x}\rangle|\vec{y}\rangle + |1\rangle|\vec{y}\rangle|\vec{x}\rangle)$. Thus an equal superposition (probability of observation = 1/2) of packets $\vec{x}$ and $\vec{y}$ is created at both the outputs. Also, both terms within the parenthesis in the previous expression contain $\vec{x}$ as well as $\vec{y}$, thus, if we observe packet $\vec{x}$ at one of the outputs then packet $\vec{y}$ will be observed with certainty at the other output and vice-versa.

II.D Classical Baseline Network

An N input and N output baseline network has $n = \log_2 N$ stages, each having $N/2$ two input-two output switches that can be set either in *through* state or in *cross* state. In the through state the upper input is connected to to the upper output and the lower input to the lower output whereas in a cross state the upper input is connected to the lower output and the lower input to the upper output. The switches in each stage are numbered $0, \cdots, N/2-1$ from top to bottom using $n-1$ bit binary numbers. The N input and N output ports of each stage are labeled $0, \cdots, N-1$ from top to bottom using n bit binary numbers. Output port $o_1 \cdots o_n$ of the m^{th} stage is connected to the input port $i_1 \cdots i_n$ of the $m+1^{th}$ stage $(1 \le m \le n-1)$ where binary number $i_1 \cdots i_n$ is obtained by doing a right circular shift on lower $n-m+1$ bits of binary number $o_1 \cdots o_n$. The 16×16 baseline network is shown in Figure 3.

Packets can be *self routed* in the baseline network in following manner. Suppose the output addresses of the packets on the upper and lower inputs of a 2×2 switch in m^{th} stage $(1 \le m \le n)$ are binary numbers $a_1 a_2 \cdots a_n$ and $b_1 b_2 \cdots b_n$ respectively. This switch is set in through state if $a_m = 0$ and $b_m = 1$ and in cross state if $a_m = 1$ and $b_m = 0$. When both a_m and b_m are same there is a contention and one of the packets has to be either dropped or buffered. Next we discuss some connection properties of the baseline network which are used later in the paper.



(a) Block diagram. (b) Quantum circuit, packet size is 2 qubits.

Figure 2: The switch gate.

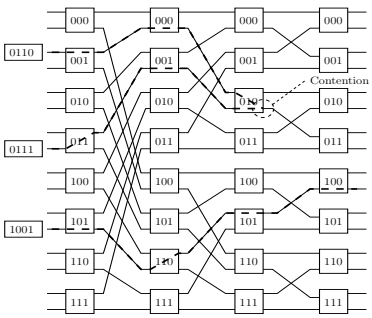


Figure 3: 16×16 Baseline network.

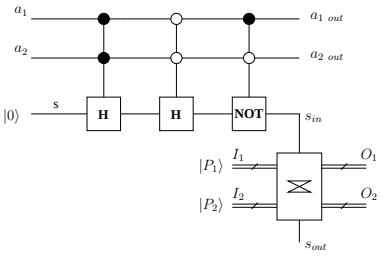


Figure 4: A 2×2 quantum switch.

Consider the m^{th} stage of the N input baseline network, where $2 \leq m \leq n$. Divide the inputs of the 1^{st} stage into 2^{n-m} consecutive disjoint sets of size 2^m each and number these sets $0, \cdots, 2^{n-m}-1$ from top to bottom. A 2×2 switch $b_1 b_2 \cdots b_{n-1}$ in the m^{th} stage of the baseline network can be reached only by the inputs in the input set number p, where p is binary number $b_m \cdots b_{n-1}$. Switches in the last stage of the network can be reached by every input. Also, the top $m-1$ address bits of any packet self-routed to switch $b_1 b_2 \cdots b_{n-1}$ in the m^{th} stage are $b_1 \cdots b_{m-1}$.

III. Quantum Switch

A simple design for a 2×2 quantum switch was given in [1]. The quantum circuit of this switch is shown in Figure 4. Two sets of qubits of size w each form the input packets of the switch. The address bits that determine the outputs are fed separately to the switch and are labeled a_1 and a_2 respectively. These bits function as the control inputs to the switch and are discarded after use. P_1 and P_2 (size $w-1$ each) are the input packets without a_1 and a_2 respectively. Another qubit s, which is initialized to state $|0\rangle$, is used as a scratch qubit to generate the control input s_{in} to the switch gate.

If $a_1 = a_2$, one of the Hadamard gates changes the state of s to $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and the switch gate creates an equal superposition of P_1 and P_2. If a_1 is $|1\rangle$ and a_2 is $|0\rangle$ then the NOT gate sets s to state $|1\rangle$, i.e., s_{in} is $|1\rangle$ and the switch gate is set in *cross* state. When a_1 is $|0\rangle$ and a_2 is $|1\rangle$, s remains in state $|0\rangle$ and the switch gate is set in *through* state. Thus, representing the state of the system by $|a_1, a_2, P_1, P_2\rangle$, the function of the switch may be described as

$$\begin{aligned}
|0,0,P_1,P_2\rangle &\longrightarrow \frac{1}{\sqrt{2}}(|0,0,P_1,P_2\rangle + |0,0,P_2,P_1\rangle) \\
|0,1,P_1,P_2\rangle &\longrightarrow |0,1,P_1,P_2\rangle \\
|1,0,P_1,P_2\rangle &\longrightarrow |1,0,P_2,P_1\rangle \qquad (5) \\
|1,1,P_1,P_2\rangle &\longrightarrow \frac{1}{\sqrt{2}}(|1,1,P_1,P_2\rangle + |1,1,P_2,P_1\rangle)
\end{aligned}$$

Even though this design creates a superposition of the contending packets at the desired output, a complementary superposition is created on the other output also which is undesirable. Even if a baseline network is made using this switch, the outputs of the network might receive packets that are not addressed to them. Also, it will not be possible to verify whether the received packet was intended for that output or not because some packets will be delivered to wrong address and their address bits are removed by the network. This problem was solved in our earlier design [1] by swapping the unwanted packet superposition on the unused output with a dummy packet which was distinguishable from other data packets. This design involved $O(\log N)$ extra qubits for each 2×2 switch. In the next section, we present a 2×2 quantum switch having an overhead of a constant number qubits and uses an extra qubit called *valid qubit* or *v-qubit* in each packet.

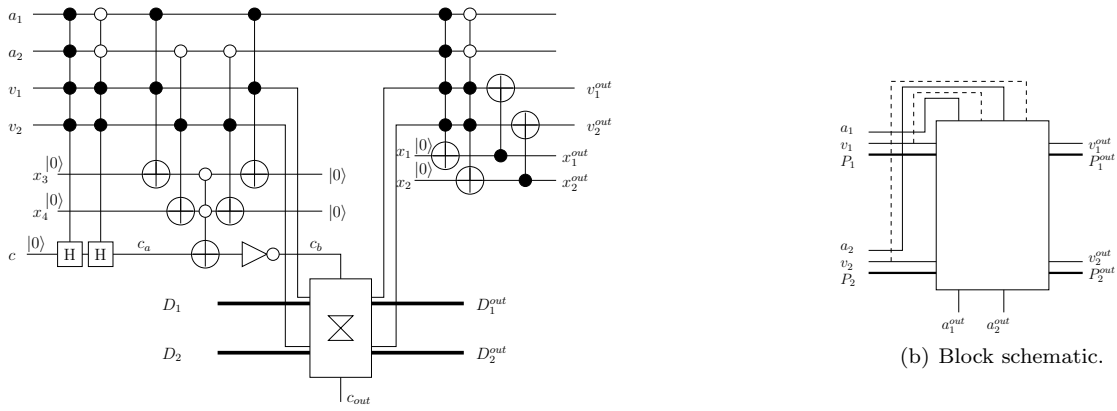
III.A 2×2 Quantum switch using v-qubits

In this section we give the design of a 2×2 quantum switch which uses a constant number of extra qubits. This switch is shown in Figure 5(a). Instead of using a dummy packet as in [1], we use a qubit called valid qubit (or v-qubit) in each data packet. Whenever a packet is routed incorrectly, its v-qubit is set to 0. A packet with its v-qubit set to 0 is called *invalid*. Output address of an invalid packet is not considered while routing, it is routed randomly through the baseline network made using this 2×2 quantum switch. We use the v-qubits to mark the unwanted superposition of packets on the unused output as *invalid* whenever an output contention occurs at a 2×2 switch.

Qubit c is used to set the switch gate in a *through* or *cross* or an equal superposition of *through* and *cross* states depending on the values of the address qubits and v-qubits of packets on the upper and lower inputs. These qubits are labeled a_1, v_1 and a_2, v_2 respectively for the two packets. The v-qubits are switched along with the packets whereas the address qubits are discarded. Whenever an output contention occurs, one of the controlled Hadamard gates changes the state of qubit c to $1/\sqrt{2}(|0\rangle + |1\rangle)$, otherwise they do not affect c and it remains in state $|0\rangle$. When there is no contention, qubit c is set in state $|0\rangle$ or $|1\rangle$ depending on the values of a_1, v_1, a_2 and v_2 according to the following table.

v_1v_2 \ a_1a_2	00	01	11	10
00	x	x	x	x
01	1	0	0	1
11	x	0	x	1
10	0	0	1	1

0: Through state
1: Cross state



(a) Quantum circuit.

(b) Block schematic.

Figure 5: 2×2 Quantum switch with v-qubits.

Thus the control input to the 2×2 switch gate should be $a_1v_1 + \bar{a_2}v_2$, which is done in the quantum circuit using reversible logic. It can be easily verified that $c_b = \{c_a \oplus (a_1v_1)'(a_2{}'v_2)'\}'$. This logic does not affect the state of c_a when it is in state $1/\sqrt{2}(|0\rangle+|1\rangle)$ and changes it to $a_1v_1+\bar{a_2}v_2$ when it is in state $|0\rangle$. Finally, in case of contention the v-qubits of the packets on unused output are marked invalid by swapping the states of v-qubits with $|0\rangle$. Note that the two auxiliary qubits x_3 and x_4 used in the circuit remain in their original state $|0\rangle$ and can be ignored because a measurement on them does not have any effect on the functioning of the circuit. Only the auxiliary qubits x_1, x_2 and the control qubit c may end up in an altered state. Denoting the packets as (a_i, v_i, P_i), $i = 1, 2$, the functioning of the switch when both the input packets are valid is:

$$
\begin{aligned}
&|(0,1,P_1),(0,1,P_2)\rangle \longrightarrow \\
&\quad \frac{1}{\sqrt{2}}\left(|(0,1,P_1),(0,0,P_2)\rangle + |(0,1,P_2),(0,0,P_1)\rangle\right) \\
&|(0,1,P_1),(1,1,P_2)\rangle \longrightarrow |(0,1,P_1),(1,1,P_2)\rangle \\
&|(1,1,P_1),(0,1,P_2)\rangle \longrightarrow |(0,1,P_2),(1,1,P_1)\rangle \qquad (6) \\
&|(1,1,P_1),(1,1,P_2)\rangle \longrightarrow \\
&\quad \frac{1}{\sqrt{2}}\left(|(1,0,P_1),(1,1,P_2)\rangle + |(1,0,P_2),(1,1,P_1)\rangle\right)
\end{aligned}
$$

If one of the input packets is invalid, the switch is simply set to route the valid packet. We do not care about the setting of the switch when both the input packets are invalid. A block schematic for this switch is shown in Figure 5(b).

IV. Quantum Baseline Network

The 2×2 quantum switch described in Section III.A is used to form the quantum baseline network (QBN). An N input ($N = 2^n$) QBN has $N/2$ stages of 2×2 quantum switches connected in baseline topology described in Section II.D. In this section we give an example of 4×4 QBN shown in Figure 6 to explain its functionality.

Suppose inputs 0, 1, 2 and 3 have packets P_3, P_2, P_0 and P_1 destined for outputs 3, 2, 0 and 1 respectively. We represent the data part of these packets as D_3, D_2, D_0 and D_1 respectively. The packets are represented as tuple (a_1a_2, v, D), where a_1a_2 is the binary number equal to the output address of the packet, v corresponds to the v-qubit and D is the packet data. We write the state of the quantum wires at locations marked in Figure 6 by vertical dotted lines. The order of the components in ket-notation $|W_1, W_2, \cdots, W_n\rangle$, corresponds to the order in which we encounter the wires as we traverse the dotted line from top to bottom. As mentioned earlier, the quantum state at any dotted line is given by the tensor product of the quantum states of the switch outputs.

The input quantum state is $|(11, 1, D_3)\,(10, 1, D_2)\,(00, 1, D_0)\,(01, 1, D_1)\rangle$. The contentions for outputs at both the switches in the first stage cause them to create the following state at location B in the figure (representing packets at this location as tuple (a_2, v, D)):

$$
\begin{aligned}
\frac{1}{2}\Big[&\big|(1,0,D_3)(0,1,D_2)(0,1,D_0)(1,0,D_1)\big\rangle + \\
&\big|(0,0,D_2)(1,1,D_3)(0,1,D_0)(1,0,D_1)\big\rangle + \\
&\big|(1,0,D_3)(0,1,D_2)(1,1,D_1)(0,0,D_0)\big\rangle + \\
&\big|(0,0,D_2)(1,1,D_3)(1,1,D_1)(0,0,D_0)\big\rangle\Big]
\end{aligned}
$$

After the shuffle, the state at C is obtained by interchanging the middle two packets in each of the four tuples (*kets*) above. All the address bits of the packets are used by the networks at the end of the second stage and a packet at the output is represented as (v, D). There is no contention in any ket at any of the switches in the second stage because one packet at the inputs of both the 2×2 switches (for every ket) is an *invalid* packet. Thus the output state of the switch is

$$
\begin{aligned}
\frac{1}{2}\big[&|(1,D_0)(0,D_3)(1,D_2)(0,D_1)\rangle + \\
&|(1,D_0)(0,D_2)(0,D_1)(1,D_3)\rangle + \\
&|(0,D_3)(1,D_1)(1,D_2)(0,D_0)\rangle + \\
&|(0,D_2)(1,D_1)(0,D_0)(1,D_3)\rangle\big]
\end{aligned}
$$

On a measurement at the outputs of the switch, one of the kets in the above expression is observed with probability 1/4. The i^{th} component of a tuple is the packet at the i^{th} output. Thus, the probability of observing a valid packet P_i, $i = 0, 1, 2, 3$, at output i is 1/2. Also, valid and correctly routed packets are observed only at any two of the outputs. The other two outputs have invalid packets.

An $N \times N$ quantum baseline network is formed in a similar way. A 2×2 quantum switch at stage m of this network uses m^{th} most significant address qubits of its input packets for routing. In the next section we develop the tools to write the output quantum state of an $N \times N$ QBN for any given permutation output assignment.

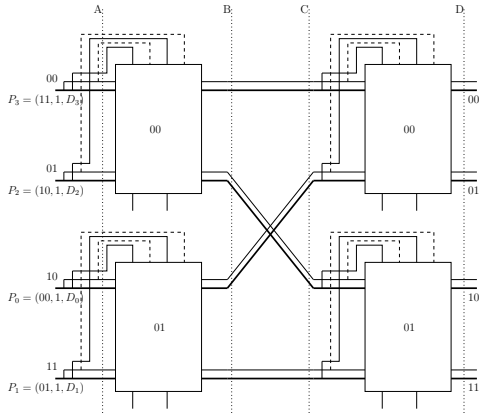
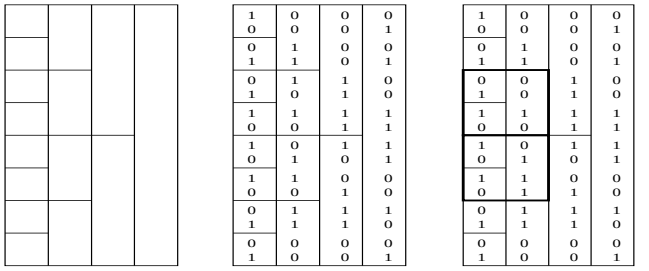


Figure 6: 4×4 Quantum baseline network.



(a) A 16×4 frame.

(b) A 16×4 permutation matrix which fits the frame.

(c) A 16×4 permutation matrix which does not fit the frame.

Figure 7: 16×4 Frame and permutation matrices.

V. The output quantum state of QBN

In this section we discuss the structure of the output quantum state. Given any permutation assignment, we give formulae for all the kets or packet sequences in the output quantum state and their coefficients. We use the ideas of balanced matrices and frames used by various authors and most recently by Çam [14]. We start with some definitions which can be used both for classical and quantum baseline networks.

Definition 1 Permutation Matrix: *A permutation assignment π in an $N \times N$ ($N = 2^n$) switch is represented by an $N \times n$ permutation matrix Π, where $\Pi(i, j)$ is the j^{th} most significant bit in the output address of the packet on input i.*

Definition 2 Balanced Matrix: *An $N \times n$ binary matrix B is* balanced *if and only if no n bit binary number appears more than once in its rows.*

Definition 3 *Divide columns 1 to m of an $N \times n$ permutation matrix ($N = 2^n$ and $1 \leq m \leq n$) into 2^{n-m} consecutive blocks ($2^m \times m$ sub-matrices) from top to bottom, each block having 2^m rows and m columns. The set consisting of all these sub-matrices is called a* frame.

Definition 4 *A $N \times n$ permutation matrix is said to* fit *an $N \times n$ frame if each sub-matrix in the frame is balanced.*

A frame for $N = 16$ is shown in Figure 7(a). A permutation matrix which fits the 16×4 frame is shown in Figure 7(b). A permutation matrix that does not fit this frame is shown in Figure 7(c) where the highlighted sub-matrices of this matrix are not balanced.

Theorem 1 *A permutation assignment can be self-routed in an $N \times N$ baseline network without blocking if and only if its permutation matrix fits an $N \times n$ frame.*

Proof: First, suppose that the $N \times n$ permutation matrix of the assignment does not fit the $N \times n$ frame. Then for some $1 \leq m < n$, an m column sub-matrix (of size $2^m \times m$) in the frame is not balanced, note that m cannot be equal to n since a permutation matrix is always balanced. There are two packets on the 2^m inputs corresponding to this sub-matrix that have the same m most significant output address bits, let these bits be $a_1 \cdots a_m$. Let the input ports of these two packets be $i_1 \cdots i_{n-m} i_{n-m+1} \cdots i_n$ and $i_1 \cdots i_{n-m} j_{n-m+1} \cdots j_n$. The $n - m$ most significant bits in the input addresses are the same because the two inputs belong to the same block of size 2^m. Using the self routing scheme described in Section II.D, both these packets get routed to the same switch $a_1 a_2 \cdots a_{m-1} i_1 \cdots i_{n-m}$ in the m^{th} stage. In the m^{th} stage the m^{th} most significant output address bit is used to choose the route, and since both the packets have the same m^{th} most significant output address bit a_m, there is a contention and the input permutation cannot be routed without blocking.

For the converse, if there is a contention at one of the 2×2 switches in m^{th} stage $1 \leq m < n$, then the top m address bits of the two input packets at this switch should be the same since any packet reaching switch $b_1 \cdots b_{n-1}$ in the m^{th} stage should have its top $m - 1$ address bits as $b_1 \cdots b_{m-1}$. Also, these two inputs belong to the same input block of size 2^m. Thus, there is an unbalanced $2^m \times m$ sub-matrix in the frame. Hence the permutation matrix does not fit the $N \times n$ frame.‖

A similar theorem for reverse baseline network is given in [14]. Here we have proved that the same condition ensures non-blocking routing in baseline network as well. Thus, the permutation in Figure 7(b) passes the baseline network without any contention while the one in Figure 7(c) has contention.

Definition 5 *The* output quantum state *of an $N \times N$ QBN (where $N = 2^n$) for any given output assignment is*

$$\sum_{i=1}^{K} a_i \left| (v_0^i, D_0^i)(v_1^i, D_1^i) \cdots (v_{N-1}^i, D_{N-1}^i) \right\rangle \tag{7}$$

where $\sum_{i=1}^{K} |a_i|^2 = 1$; and v_j^i, D_j^i ($0 \leq j \leq N-1$) respectively are the v-qubit and data part of the packet at output j of the QBN in the i^{th} ket.

On measurement the i^{th} ket will be observed with probability $|a_i|^2$. The smallest value of K is 1 which occurs when there is no contention. The following theorem follows directly from Theorem 1.

Theorem 2 *Let $\pi = (o_0, \cdots, o_{N-1})$ be a permutation of numbers $0, \cdots, N-1$ where $N = 2^n$, and input i of a QBN have a packet P_{o_i} destined to output o_i, where $0 \leq i \leq N-1$. If the permutation matrix of π fits the $N \times n$ frame then the output quantum state of the QBN is $|(1, D_0), \cdots, (1, D_{N-1})\rangle$, where D_i is the data part of packet P_i.*

If a permutation assignment does not fit the $N \times n$ frame it is broken into kets (packet tuples) each having some invalid

packets as shown in the example given in Section IV. Thus the value of K will be more than one for such an assignment. The exact number of invalid packets in each ket and the number of kets present in output quantum state can be easily derived using the ideas of frame and fitting discussed above. For this we first make a $N \times n$ permutation matrix fit an $N \times n$ frame by marking some of the rows as *invalid* using the following procedure called *balancing*.

Balancing a permutation matrix: First for $m = 1$, the row corresponding to one of each repeated 1 bit number (if any) in each 2×1 sub-matrix of the frame is marked *invalid*. Next, for $2 \leq m \leq n$, check only the valid rows (which are not marked invalid) and make the row corresponding to one of each repeated m bit binary number (if any) in every $2^m \times m$ sub-matrix of the frame as *invalid*. A permutation matrix which is obtained by this process is called a *balanced permutation matrix*.

Since there are many choices of rows that can be marked invalid in each step of the balancing procedure, there are many possible balanced matrices that can be generated by the balancing procedure. Following theorem gives the output quantum state of the QBN in this case.

Theorem 3 *Let $\pi = (o_0, \cdots, o_{N-1})$ be a permutation of numbers $0, \cdots, N-1$ ($N = 2^n$) which does not fit an $N \times n$ frame. Suppose that input i of a QBN has a packet P_{o_i} destined to output o_i, where $0 \leq i \leq N-1$ and M possible balanced matrices can be generated by balancing the permutation matrix of π. Let $I_j \subset \{0, \cdots, N-1\}$ be the set of inputs corresponding to the invalid rows of the j^{th} balanced matrix, where $1 \leq j \leq M$. Then the output quantum state of the QBN is:*

$$\sum_{j=1}^{M} a_j \left| (v_0^j, D_0^j)(v_1^j, D_1^j) \cdots (v_{N-1}^j, D_{N-1}^j) \right\rangle \tag{8}$$

where v_i^j is 1 if $\pi^{-1}(i) \notin I_j$ and 0 if $\pi^{-1}(i) \in I_j$, $0 \leq i \leq N-1$. Also, D_i^j is the data part of packet P_i if and only if $v_i^j = 1$. Furthermore,

$$a_j = \left(\frac{1}{\sqrt{2}}\right)^{|I_j|} \tag{9}$$

where $|I_j|$ is the cardinality of set I_j.

Proof: The proof of this theorem is deferred to the full version of this paper.

On measurement, one of the kets in the output quantum state will be observed and the probability of finding the ket having $|I_j|$ invalid packets and $N - |I_j|$ valid packets is $(1/2)^{|I_j|}$. This is desirable as the kets having larger number of valid packets have higher probability of being measured. Probability of finding any particular packet at its desired output can be obtained by summing the probabilities of the kets in which that packet is valid.

VI. Conclusion

In this paper we have described the design of the quantum baseline network that creates a superposition of permutations of non-conflicting or "balanced" subsets of an input permutation assignment. This was done by setting the v-qubit in packets contending for the same output. As a result, all packets are routed through the network without blocking. We have also given a characterization of the output permutations generated from a given input permutation and the associated probability distribution. A simple measurement destroys the output superposition state and gives only one output permutation, which is equivalent to classical routing through a baseline network with random packet drops in case of contentions. More sophisticated measurements can be done to get more information about the packets from the output state. Another advantage of the quantum network over a conventional baseline network is that there is no need to take local decisions to resolve contentions.

The results of this paper show that quantum packet switching can be an effective approach to mitigate congestion in packet switches. An interesting direction of future work would be to incorporate routing with priorities in a quantum switching framework which we will incorporate in the full version of the paper. Also, extending our results to more powerful switching structures like Beneš networks with randomization and quantum search could lead to novel strictly non-blocking switching fabrics.

References

[1] M. K. Shukla, R. Ratan, and A. Y. Oruç, "A quantum self-routing packet switch," in *Proceedings of the 38th Annual Conference on Information Sciences and Systems CISS'04*, Princeton, NJ, USA, Mar. 2004, pp. 484–489.

[2] F. K. Hwang, *The Mathematical Theory of Nonblocking Switching Networks*, ser. Series on applied mathematics, F. K. Hwang, Ed. Singapore: World Scientific Publishing Co. Pte. Ltd, 1998, vol. 11.

[3] C. Clos, "A study of non-blocking switching networks," *Bell System Technical Journal*, vol. 32, no. 2, pp. 406–424, Mar. 1953.

[4] D. G. Cantor, "On nonblocking switching networks," *Networks*, vol. 1, pp. 367–377, 1971.

[5] M. Koppelman and A. Y. Oruç, "A self-routing permutation network," *Journal of Parallel and Distributed Computing*, pp. 140–151, Oct. 1990.

[6] C. Y. Lee and A. Y. Oruç, "Design of efficient and easily routable generalized connectors," *IEEE Trans. Commun.*, pp. 646–650, Feb. 1995.

[7] V. E. Beneš, *Mathematical Theory of Connecting Networks and Telephone Traffic*. Academic Press, 1965.

[8] J. H. Patel, "Performance of processor memory interconnections for multiprocessors," *IEEE Trans. Comput.*, vol. 30, no. 10, pp. 771–780, Oct. 1981.

[9] D. H. Lawrie, "Access and alignment of data in an array processor," *IEEE Trans. Comput.*, vol. 25, pp. 1145–1155, 1976.

[10] J. Y. Hui, *Switching and Traffic Theory for Integrated Broadband Networks*, R. Gallager, Ed. Kluwer Academic Publishers, 1990.

[11] P. W. Shor, "Polynomial-time algorithm for prime factorization and discrete logarithms on a quantum computer," *SIAM Review*, vol. 41, no. 2, pp. 303–332, 1999.

[12] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, May 1996, pp. 212–219.

[13] N. Linial and M. Tarsi, "Interpolation between bases and the shuffle-exchange network," *European Journal of Combinatorics*, vol. 10, pp. 29–39, 1989.

[14] H. Çam, "Rearrangeability of (2n -1)-stage shuffle-exchange networks," *SIAM Journal on Computing*, vol. 32, no. 3, pp. 557–585, Mar. 2003.

[15] J. Preskill. (1998) Physics 229: Advanced mathematical methods of physics – quantum computation and information. [Online]. Available: http://www.theory.caltech.edu/people/preskill/ph229

[16] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, Sept. 2000.

[17] D. Deutsch, "Quantum computational networks," *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, vol. 425, no. 1868, pp. 73–90, Sept. 1989.