Parallel QoS Scheduling for WDM Optical Interconnection System Using a New Ranked Hopfield Neural Network

Po-Lung Tien, *Member, IEEE*, and Bo-Yu Ke, *Student Member, IEEE*

***Abstract*—In this paper, we propose a parallel QoS scheduler for a WDM optical interconnection system (WOPIS), using a new ranked Hopfield neural-network (RHNN). The WOPIS contains a set of Clos-like optical switches and a handful of output FDL-based optical buffers. The RHNN scheduler determines an optimal set of neurons (I/O paths) to be enabled, achieving maximal system throughput and priority differentiation subject to the switch- and buffer-contention-free constraints. Structured with ranked neurons, the RHNN allows higher-rank neurons (higher-priority and/or lower-delay paths) to disable lower-rank neurons that have been enabled during previous iterations. Ranking the neurons unfortunately gives rise to a convergence problem. We present two theorems that give the sufficient conditions for the RHNN scheduler to converge to the optimal solution. We demonstrate via simulation results that, with the computation time within one system slot time, the RHNN scheduler achieves near 100% throughput and multi-level prioritized scheduling.**

***Index Terms*—Hopfield neural networks, optical interconnect, parallel scheduling, quality of service (QoS).**

I. Introduction

WDM optical interconnect [1], [2], which provides exceedingly high bandwidth and low power dissipation, has recently been considered as a prominent candidate for interconnecting mass parallel computing processors particularly in datacenters. In our earlier work [3], we have experimented a 10-Gb/s WDM optical interconnect switching system, which contains a set of optical space switches, downsized feed-forward FDL-based optical buffers [4], and tunable optical wavelength converters (TOWC's) [5]. In particular, each optical space switch is of broadcast-and-select structure implemented by interconnected 2×2 semiconductor-optical-amplifiers (SOAs) [6], [7] switching elements. The switches exhibit excellent properties, such as fast response time, large extinction ratio, and low crosstalks, however at the expense of higher power consumption and large fully integrated switching elements.

Numerous optical switching-element architectures have been proposed. Of these proposals, the approach of using a single modulation stage [8] to directly realize $1\times N$ or $N\times N$ switches receives much attention. The approach basically has potential advantages over conventional schemes with respect to low optical power consumption and insertion loss, when scaling to large switches. Among such single-modulation-stage switches, the InP-InGaAsP optical phased-array switch (OPAS) [8] has particularly demonstrated fast configuration speed, hence advantageous to serve as a basic switching element in constructing large optical interconnection systems. Pertaining to the number of switching stages, it has been shown that, by incorporating an efficient scheduling algorithm [4], a switch with three stages (e.g., Clos network [9]) can achieve a throughput that is as superior as its counterparts with more switching stages. In this work, we aim at designing a high-performance parallel QoS scheduling approach for the WDM OPAS-based optical interconnection system (WOPIS).

Packet scheduling for WOPIS is a complex problem and impractical to be solved by existing sequential-based algorithms. Thanks to the advances in VLSI and parallel computation technologies, Hopfield neural network (HNN) [10] and its variants [11]–[16] have been successfully employed in solving hard optimization problems. Among them, a few HNN approaches [17], [18] have been proposed to particularly deal with packet scheduling for electronic switches. The work [17] proposed an NN architecture for packet routing through a crossbar switch achieving near-optimum throughput. The work [18] experimentally implemented an optoelectronic neural-network for crossbar switch controller that achieves maximum scalability and throughput. However, all these approaches have been applied to buffer-less switches. Most importantly, due to the requirement of energy-function convergence, existing HNNs operate on the basis of having symmetric neuron weights. To our knowledge, our work is the first HNN that adopts asymmetric neuron weights to efficiently accomplish prioritized packet scheduling for optical buffered switches.

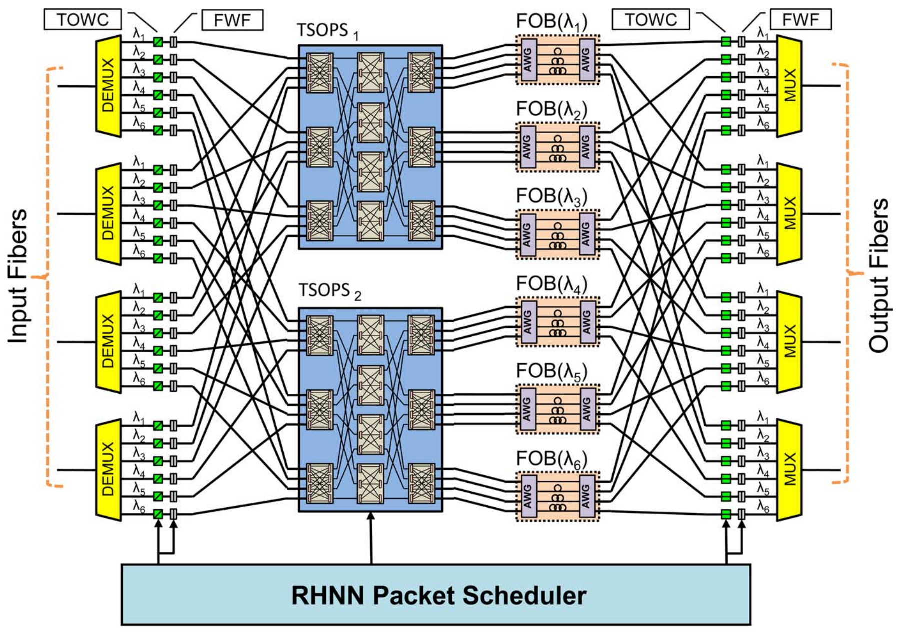
In this paper, we propose a new *ranked* Hopfield neural network (RHNN) parallel scheduler for WOPIS, achieving maximal system throughput and priority differentiation. The RHNN is specially structured with ranked neurons. With each neuron being associated with an I/O path within WOPIS, the RHNN scheduler allows higher-rank neurons to disable previously enabled lower-rank neurons. Ranking the neurons results in asymmetric neuron weights, which unfortunately gives rise to a convergence problem. To resolve the problem,

Manuscript received February 12, 2011; revised June 02, 2011; accepted June 02, 2011. Date of publication June 16, 2011; date of current version July 29, 2011. This work was supported by the National Science Council (NSC), Taiwan, under Grant 98-2221-E-009-067-MY3.

The authors are with the Department of Electrical Engineering, National Chiao Tung University, 30050 Hsinchu, Taiwan (e-mail: polungtien@gmail.com; pyk13.cm94@g2.nctu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JLT.2011.2159578



Legend:
TSOPS: Three-Stage OPAS-based Space Switch; FWF: Fixed Wavelength Filter;
FOB (λ_i): FDL Optical Buffers for λ_i; TOWC: Tunable Optical Wavelength Converter;
AWG: Arrayed Waveguide Grating;

Fig. 1. WOPIS: system architecture.

we rigorously present two theorems that supply the sufficient conditions for the RHNN scheduler to converge to the optimal solution. We demonstrate via simulation results that the RHNN scheduler achieves near 100% throughput and multi-level prioritized scheduling.

The remainder of this paper is organized as follows. In Section II, we present the architecture of WOPIS. In Section III, we formally define the QoS scheduling problem. We then detail the RHNN scheduler in Section IV and demonstrate simulation results in Section V. Finally, conclusion remarks are given in Section VI.

II. WOPIS System Architecture

WOPIS consists of two subsystems (see Fig. 1): the optical interconnect subsystem, and the RHNN packet scheduler. While each packet header that carries the label and QoS (priority) information is electronically processed by the RHNN scheduler, the payload is transported within the optical interconnect subsystem in the optical domain. The optical subsystem consists of four sections: input, three-stage OPAS-based optical space switch (TSOPS), output optical buffer, and output. In the input section, there are input fibers each carrying a number of input wavelengths. After DEMUX, a tunable optical wavelength converter (TOWC) converts each packet's input wavelength to an internal wavelength that corresponds to a free space in the output optical buffer.

In the TSOPS section, each TSOPS is responsible for switching packets carried by a specific cluster of wavelengths. For example, as shown in Fig. 1, space switches TSOPS_1 and TSOPS_2 are for λ_1–λ_3 and λ_4–λ_6, respectively. The structure of a TSOPS can easily be explained via the example in Fig. 1. In the example, a 12×12 TSOPS is comprised of a total of ten OPAS-based switching elements, in which there are three 4×4 switching elements in each of the first and last stages, and four 3×3 elements in the middle stage. Each $N \times N$ OPAS-based switching element consists of N OPASs of size $1 \times N$ that are fully connected to the other N OPASs of size $N \times 1$. Being wavelength selective, each OPAS (of size $1 \times N$ or $N \times 1$) operates like an arrayed waveguide gratings (AWG) [19]. For instance, a 1×4 OPAS with four wavelengths $(\lambda_1$–$\lambda_4)$ arriving at the input port, will have wavelength λ_k departed from the k_{th} output, where $k = 1$ to 4.

In the output buffer section, there is an FDL optical buffer (FOB) for each input wavelength. Each FOB is shared by all output ports. An FOB is composed of a pair of AWG's and F optical FDL's connecting the AWG's, resulting in a total of $F \times L$ buffer positions (including those with no delay), where L is the number of internal wavelengths. It is worth noting that, a packet entering the FOB at the i_{th} input port will exit the buffer from the i_{th} output port after receiving a certain delay time determined by the internal wavelength. Thus, for any FOB, an internal wavelength of a packet uniquely determines the delay received by the packet. In the output section, there are $N \times W$ FWM-based TOWC's, and N output fibers each carrying W wavelengths. At each output port of the second AWG of an FOB, buffer contention occurs when multiple packets attempt to depart from the output port simultaneously. Finally, the RHNN packet schedule is responsible for scheduling newly arriving packets of different priorities such that the switch- and buffer-contention-free constraints are satisfied, in an effort to achieve maximal system throughput and priority differentiation.

III. QoS Scheduling Problem Definition

Because packet scheduling for different TSOPS is completely identical and independent, we thereinafter describe the scheduling problem under an assumption that there is only one

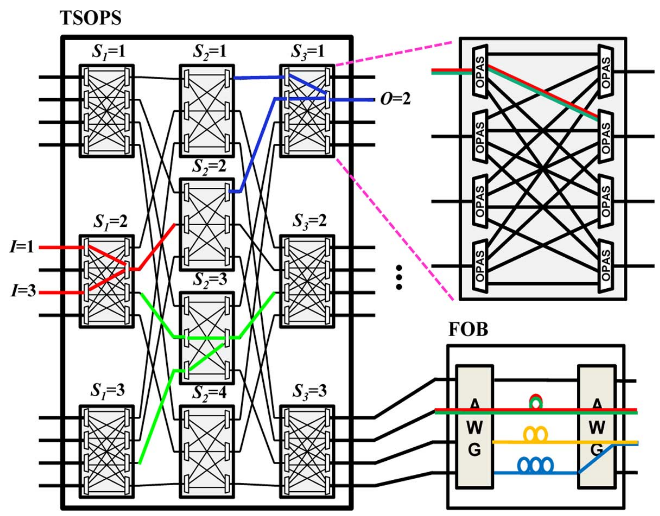


Fig. 2. Switch and buffer contention—an illustration.

TSOPS in the entire system, as shown in Fig. 2. Let N denote the total number of input/output fibers, L the number of internal wavelengths, M the number of switching elements in the first/last stage of TSOPS, and K the number of switching elements in the second stage. Notice that the number of switching elements in the first and last stage is the same as the number of input/output wavelengths. Let I (O) denote the index of the input (output) port of a switching element in the first (third) stage, where $I, O = 1$ to N. Let $S_1(S_3)$ denote the index of a first-stage (third-stage) switching element, where $S_1, S_3 = 1$ to M; and S_2 the index of a second-stage switching element, where $S_2 = 1$ to K. Let λ denote the internal wavelength, where $\lambda = 1$ to L; and F the number of FDLs in each FOB. For the example in Fig. 2, $N = 4$, $L = 4$, $M = 3$ and $K = 4$.

We define three functions in the following. First, $pr(I, S_1)$ denotes the priority of a packet taking the path from input port I of switching element S_1 in the first stage of TSOPS. Function $d\ (O, \lambda)$ denotes the delay of a buffer that is associated with output port O (of any switching element in the last stage) and internal wavelength λ. Function $bfs(S_3, O, \lambda)$ denotes the status of the buffer location in FOB that is uniquely identified by switching element S_3 in the third stage, output port O, and internal wavelength λ. $bfs(S_3, O, \lambda) = 1$ if the corresponding buffer is occupied; and 0 otherwise.

Definition 1: An *in-out path* for a packet carried by internal wavelength λ, denoted as $(I, S_1, S_2, S_3, O, \lambda)$, is a route that starts from input port I of the first-stage switch S_1, through the second-stage switch S_2, to output port O of the third-stage switch S_3. The *status of an in-out path* $(I, S_1, S_2, S_3, O, \lambda)$ is defined by an index function, $Y_{(I,S_1,S_2,S_3,O,\lambda)}$; namely $Y_{(I,S_1,S_2,S_3,O,\lambda)} = 1$, if the in-out path is taken by a packet; and 0, otherwise.

Definition 2: A *valid path set* (U) is a set of in-out paths that follow the FOB rule: any packet entering the FOB from the i_{th} input port (of the first AWG) always departs from the i_{th} output port (of the second AWG). Mathematically, the set can be given as

$$U \equiv \{(I,S_1,S_2,S_3,O,\lambda) | \{\lambda - [(\lambda - O) \bmod F] \bmod N\} = O\}.$$

Basically, switch contention occurs when two or more packets carried by the same internal wavelength attempt to pass through the same internal link within TSOPS. Accordingly, together with the wavelength selective property of OPAS, switch-contention-free scheduling has to meet two following guidelines. The first guideline stipulates that packets from different inlets of a switching element but carried by the same internal wavelength cannot be simultaneously switched to the same outlet of the element. This is illustrated in the example in Fig. 2. In the figure, switch contention occurs at three different stages of TSOPS due to using the same internal wavelength (same color in the figure).

The second guideline states that, within any switching element, multiple packets from the same inlets that are carried by different internal wavelengths always depart from different outlets (due to being wavelength selective of OPAS). A violation of this guideline is depicted in the first switching element at the third stage $(S_3 = 1)$ in Fig. 2. According to these two guidelines, the switch-contention-free constraint is comprised of six rules: $\Re_1$ to $\Re_3$ based on the first guideline, and $\Re_4$ to $\Re_6$ based on the second guideline.

$$\Re_1 \text{ to } \Re_6 : \sum_{G_i} Y_{(I,S_1,S_2,S_3,O,\lambda)} Y_{(I',S'_1,S'_2,S'_3,O',\lambda')} = 0, \quad \text{for } i = 1 \text{ to } 6$$

where

$$\begin{aligned}
G_1 \equiv &\{U \times U | (S_1, S_2, \lambda) = (S'_1, S'_2, \lambda'),\\
&(I, S_3, O) \neq (I', S'_3, O')\}\\
G_2 \equiv &\{U \times U | (S_2, S_3, \lambda) = (S'_2, S'_3, \lambda'),\\
&(I, S_1, O) \neq (I', S'_1, O')\}\\
G_3 \equiv &\{U \times U | (S_3, O, \lambda) = (S'_3, O', \lambda'),\\
&(I, S_1, S_2) \neq (I', S'_1, S'_2)\}\\
G_4 \equiv &\{U \times U | (I, S_1, S_2) = (I', S'_1, S'_2),\\
&(S_3, O, \lambda) \neq (S'_3, O', \lambda')\}\\
G_5 \equiv &\{U \times U | (S_1, S_2, S_3) = (S'_1, S'_2, S'_3),\\
&(I, O, \lambda) \neq (I', O', \lambda')\}\\
G_6 \equiv &\{U \times U | (S_2, S_3, O) = (S'_2, S'_3, O'),\\
&(I, S_1, \lambda) \neq (I', S'_1, \lambda')\}.
\end{aligned}$$

Generally, buffer contention occurs when two or more packets attempt to depart from the same output of the second AWG of the FOB. This contention takes place under two situations- between two newly arriving packets; and between a newly arriving packet and a packet currently in the buffer. In the following, we give rules $\Re_7$ and $\Re_8$ for avoiding buffer contention under these two situations, respectively:

$$\Re_7 : \sum_{G_7} Y_{(I,S_1,S_2,S_3,O,\lambda)} Y_{(I',S'_1,S'_2,S'_3,O',\lambda')} = 0$$

where

$$\begin{aligned}
G_7 \equiv &\{U \times U | (S_3, O, d(O, \lambda)) = (S'_3, O', d(O', \lambda')),\\
&(I, S_1, S_2, \lambda) \neq (I', S'_1, S'_2, \lambda')\}
\end{aligned}$$

$$\Re_8 : \sum_{U} bfs(S_3, O, \lambda) \times Y_{(I,S_1,S_2,S_3,O,\lambda)} = 0.$$

A violation of rule $\Re_7$ is exemplified in the second FDL of the FOB in Fig. 2. In this example, two packets carried by different wavelengths will depart from the FOB at the same time. Notice that this only occurs when the number of FDLs is less than the number of inlets/outlets of the FOB. A violation of rule $\Re_8$ is shown in the third and forth FDLs of the FOB in Fig. 2. With the above eight rules, $\Re_1$ to $\Re_8$, letting $G \equiv G_1 \cup G_2 \cup G_3 \cup G_4 \cup G_5 \cup G_6 \cup G_7$, we now define the contention-free constraint.

Contention-Free (CF) Constraint:

$$f(\mathrm{CF}) \equiv \frac{1}{2}\sum_{G} Y_{(I,S_1,S_2,S_3,O,\lambda)} Y_{(I',S'_1,S'_2,S'_3,O',\lambda')} + \sum_{U} bfs(S_3,O,\lambda) \times Y_{(I,S_1,S_2,S_3,O,\lambda)} = 0. \quad (1)$$

Notice that, to satisfy the CF constraint in (1), one can just select no or few in-out paths, resulting in poor throughput. To mitigate the problem, we introduce the second constraint, called *Single Assignment (SA)*, which sets a target of assigning a path for each newly-arriving packet, thereby achieving maximal throughput.

Single Assignment (SA) Constraint:

$$f(\mathrm{SA}) \equiv \sum_{I,S_1,O}\left[\left(\sum_{S_2,S_3,\lambda} Y_{(I,S_1,S_2,S_3,O,\lambda)}\right) - n(I,S_1,O)\right]^2 = 0 \quad (2)$$

where $(I,S_1,S_2,S_3,O,\lambda) \in U$, and $n(I,S_1,O)$ is a binary function that is equal to 1 if there is an incoming packet from input port I of the first-stage switch, S_1, which is destined to output port O of a third-stage switch; and 0, otherwise.

So far, we have described two general constraints, i.e., *CF* and *SA*. With QoS taken into account, we now introduce two priority-oriented constraints. First, all packets have different priorities. The first priority constraint, referred to as *Priority-First (PF)*, stipulates that newly-arriving higher priority packets take precedence over newly-arriving lower priority packets. Notice that, by satisfying constraint *SA*, system throughput can be maximized if we only consider a single batch of packet arrivals. If we consider the real situation with consecutive batches of packet arrivals, unsurprisingly, system throughput increases with lower buffering delay. Hence, the second priority constraint, called *Minimum Delay (MD)*, stipulates that smaller delay buffer space take precedence over larger delay buffer space. Having defined the four above constraints, we are now at the stage of defining the QoS scheduling problem.

QoS Scheduling Problem Definition: For WOPIS, consider consecutive batches of packet arrivals, the QoS scheduling problem is to find a set of in-out paths for each batch, achieving two general constraints, *CF* and *SA*, and two priority constraints, *PF* and *MD*. (In other words, QoS scheduling aims to satisfy contention-free prioritized scheduling, specified by constraints *CF* and *PF*, while achieving maximum throughput, specified by constraints *SA* and *MD*.

Definition 3: (i) Under the *CF* constraint, a *contention conflicting group* is defined to be a set of in-out paths that are mutually contending with each other. Hence, the RHNN scheduler is to select at most one in-out path in each contention conflicting group; and (ii) Under the *SA* constraint, a *packet conflicting group* for a packet is defined to be a set of in-out paths, each of which is a legitimate path candidate for the packet. Hence, the RHNN scheduler is to select exactly one in-out path in the group for the packet.

IV. The RHNN Scheduler

In this section, we first briefly describe the HNN approach and address its limitation of dealing with the priority-based scheduling problem. We then propose the RHNN, i.e., an HNN with ranked neurons, and formally describe its general structure and detailed design via two theorems for resolving a generic priority-based optimization problem. Through applying the theorems to our QoS scheduling problem, we present the RHNN scheduler design including the sufficient conditions for the RHNN scheduler to converge to the optimal solution.

A. RHNN- General Structure and Design

An HNN is a single-layer feedback network that has widely been used to solve optimization problems specified by constraints. It consists of a set of neurons that are interconnected with purpose-designed neuron weights. During the iterative operation, the update rule for each neuron x is defined as [10]: $V_x^{(k+1)} = u(\mathrm{net}_x^{(k)})$, where $V_x^{(k+1)}$ is the output of neuron x ($= 1$ if enabled) at update iteration $k+1$; $u(z)$ is a unit step function that is equal to 1 if $z \geq 0$, and 0, otherwise; $\mathrm{net}_x^{(k)}$ is the net weighted input of neuron x at iteration k, namely $\mathrm{net}_x^{(k)} = \sum_{y \neq x} W_{xy} V_y^{(k)} - \theta_x$, where W_{xy} is the neuron weight from neuron y to neuron x, and θ_x is a system parameter, called threshold, for neuron x. That is, the output of each neuron at iteration $k+1$ merely depends on the pre-assigned weights and other neurons' outputs at iteration k. For neuron x, a positive weight from neuron y is viewed as excitatory, and a negative weight is regarded as inhibitor. Significantly, the behavior of an HNN is characterized by the energy function,

$$E \equiv -\frac{1}{2}\sum_{x}\sum_{y \neq x} W_{xy} V_y V_x + \sum_{x} \theta_x V_x. \quad (3)$$

Ultimately, solving an optimization problem by an HNN is to determine the neuron weights by equating the problem constraints with the energy function in (3).

It has been proved [10] that, via iteration the energy function (E) will monotonically decrease and eventually converge to an equilibrium state (absolute minimum value) if three requirements are met. They are: 1) no self-feedback connection (i.e., $W_{xx} = 0$); 2) symmetric weights (i.e., $W_{xy} = W_{yx}$); and 3) asynchronous neuron update (i.e., one neuron update at a time). These requirements, however, pose a severe challenge to solving priority-based problems. Due to the symmetric-weight requirement, lower-priority neurons have the same probability to be enabled as higher-priority neurons. Assume a low-priority neuron is first enabled; then a higher-priority neuron can no longer be enabled, if the problem constraint unfortunately demands no more than one neuron in the same group (e.g., contention or packet conflicting group).

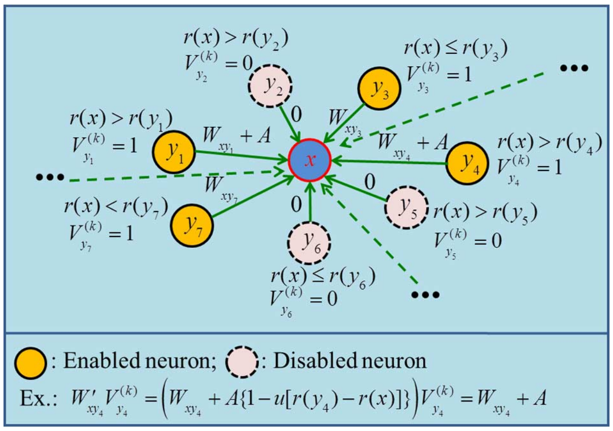


Fig. 3. RHNN operation (iteration k).

To meet the challenge, we propose the RHNN with *ranked* neurons that facilitates prioritized enabling of neurons. Let $r(x)$ be the rank function of neuron x. The neuron weights between higher-rank and lower-rank neurons are asymmetrically structured such that higher-rank neurons are assured of receiving extra positive stimulations, called **rank stimulation** that is of specific form $A\times\{1-u[r(y)-r(x)]\}$, from the lower-rank neurons. Fig. 3 shows an example of RHNN operation at iteration k. Focusing on neuron x, any enabled neuron with rank lower than $r(x)$ will send extra $A(\because A\times\{1-u[r(y)-r(x)]\} = A\times(1-0) = A)$ to neuron x. In Fig. 3, for example, enabled neuron y_4 has the lower rank than that of x; it will then send $W_{xy_4}+A$ to neuron x while x sends $W_{y_4x}(=W_{xy_4})$ to y_4 (not shown in the figure), manifesting the asymmetric-weight characteristic of RHNN. Provided that A is a large enough positive constant, such a rank-stimulation design allows a higher-rank neuron in a conflicting group to be enabled even though a lower-rank neuron in the same group has already been enabled.

Nevertheless, due to asymmetric weights, RHNN violates the second requirement of HNN for guaranteeing that the energy function asymptotically converges to the equilibrium state. To resolve this convergence problem, in the following we rigorously present two theorems that supply the sufficient conditions for the RHNN scheduler to converge to the optimal solution. Specifically, the two sufficient conditions of the theorems set forth the designation of the neuron weights. To our QoS scheduling problem, Theorem 1 is to be applied for the CF-constraint given in (1); and Theorem 2 is for the SA-constraint given in (2). For the ease of explanation, we denote $x_n^{(k)}$ as the n_{th} highest-rank neuron in the set of enabled neurons at iteration k; and $x_{\langle n\rangle}$ as the n_{th} highest-rank neuron in the set of all neurons. In addition, $\{x_n^{(k)}\} = \emptyset$ if the number of enabled neurons at iteration k is less than n; and $r(x) = 0$, if $x \in \emptyset$.

Theorem 1 (in words): Under HNN, a given constraint which assures at most one neuron will be enabled when the HNN converges to the equilibrium state, then the specially designed RHNN (see below) subject to the same constraint assures at most one neuron with the highest rank will be enabled when it converges to the equilibrium state.

Theorem 1: Given a constraint, $C : f_C \equiv \rho\sum_x\sum_{y\neq x} V_y V_x + \sum_x \tau(x) V_x = 0$, where V_x is the output of neuron x, $\tau(x) > -2\rho$, and $\rho > 0$, the neuron weight of HNN, W_{xy}, is attained from equating energy function E with f_C. For RHNN, if the neuron weight is given as $W'_{xy} = W_{xy} + A \times \{1 - u[r(y) - r(x)]\}$, where A is a pre-assigned coefficient of rank stimulation, and $u(\cdot)$ is a unit step function, then the RHNN will converge to the equilibrium state under which *at most* one neuron with the highest rank would be enabled.

Proof: By equating f_C to energy function E in (3), we obtain $W_{xy} = 2\rho$ and $\theta_x = \tau(x)$. By assigning $A = 2\rho$, we get $W'_{xy} = -2\rho + 2\rho\{1 - u[r(y) - r(x)]\}$ for RHNN. Thus, $\text{net}'^{(k)}_x = \sum_{y\neq x} W'_{xy} V_y^{(k)} - \theta_x = -2\rho\sum_{y\neq x, r(y)\geq r(x)} V_y^{(k)} - \tau(x)$ at iteration k. We now consider the state change of any neuron, say x, from iteration k to $k+1$, under two cases: $\tau(x) > 0$, and $\tau(x) \le 0$.

First, under $\tau(x) > 0$, since $\rho > 0$ and $\sum_{y\neq x, r(y)\geq r(x)} V_y^{(k)} \geq 0, \text{net}'^{(k)}_x$ has a negative value. That is to say, neuron x will be disabled at iteration $k+1$ (regardless of its rank). This is true for all x if $\tau(x) > 0$. Thus, all neurons would stay disabled, with the result that the RHNN will converge to the equilibrium state under which R is satisfied.

Second, we consider all neurons x, where $\tau(x) \le 0$. We are to prove that $r(x_1^{(k+1)}) = r(x_1^{(k)})$ is monotonically increasing $(r(x_1^{(k+1)}) \geq r(x_1^{(k)}))$ and converges to $r(x_{\langle 1\rangle})$ (i.e., the neuron with the highest rank would be enabled), and other neurons will be disabled. To prove this, the four following cases are considered.

Case I: We assume there is no enabled neuron at iteration k (i.e., $\{x_1^{(k)}\} = \emptyset$). Since $V_y^{(k)} = 0$ for all y, we have $\text{net}'^{(k)}_x = -\tau(x) \geq 0$, which indicates that x will be enabled at iteration $k+1$ (regardless of its rank). This implies $r(x_1^{(k+1)}) > r(x_1^{(k)})$.

Case II: We assume there is at least one enabled neuron at iteration k (i.e., $\{x_1^{(k)}\} \neq \emptyset$) and $r(x) > r(x_1^{(k)})$. Since the rank of any enabled neuron is smaller than $r(x)$, we immediately get $\sum_{y\neq x, r(y)\geq r(x)} V_y^{(k)} = 0$, implying $\text{net}'^{(k)}_x = -\tau(x) \geq 0$. Therefore, x will be enabled at iteration $k+1$, resulting in $r(x_1^{(k+1)}) = r(x) > r(x_1^{(k)})$.

Case III: We assume there is at least one enabled neuron at iteration k (i.e., $\{x_1^{(k)}\} \neq \emptyset$) and $r(x) = r(x_1^{(k)})$. Note that x can be either already enabled or disabled at iteration k. If x is enabled and is the only one neuron with the highest rank, then $\sum_{y\neq x, r(y)\geq r(x)} V_y^{(k)} = 0$ and $\text{net}'^{(k)}_x = -\tau(x) \geq 0$. Thus, x will stay enabled at iteration k and hence $r(x_1^{(k+1)}) = r(x) = r(x_1^{(k)})$. However, if there is more than one enabled neurons with the highest rank (including x), then we have $\sum_{y\neq x, r(y)\geq r(x)} V_y^{(k)} \geq 1$. Since $\tau(x) > -2\rho$, we get $\text{net}'^{(k)}_x \leq -2\rho - \tau(x) < 0$, which implies x will be disabled at iteration $k+1$. Since those other neurons with the highest rank remained enabled, $r(x_1^{(k+1)})$ is unchanged, i.e., $r(x_1^{(k+1)}) = r(x_1^{(k)})$. On the other hand, if x is disabled at iteration k, we have $\sum_{y\neq x, r(y)\geq r(x)} V_y^{(k)} \geq 1$. By the same token, $\text{net}'^{(k)}_x \leq -2\rho - \tau(x) < 0$, with the result that x will stay disabled and $r(x_1^{(k+1)}) = r(x_1^{(k)})$.

Case IV: We assume there is at least one enabled neuron at iteration k (i.e., $\{x_1^{(k)}\} \neq \emptyset$) and $r(x) < r(x_1^{(k)})$. We have

$\sum_{y \neq x, r(y) \geq r(x)} V_y^{(k)} \geq 1$, regardless of the state of x at iteration k. Namely, $\mathrm{net}'^{(k)}_x \leq -2\rho - \tau(x) < 0$, yielding that x will be disabled at iteration $k+1$. However, since $r(x) < r(x_1^{(k)})$, the state change of x is irrelevant to $r(x_1^{(k+1)})$, which implies $r(x_1^{(k+1)}) = r(x_1^{(k)})$.

From the four above cases, we have proved that $r(x_1^{(k)})$ is monotonically increasing $(r(x_1^{(k+1)}) \geq r(x_1^{(k)}))$. Since $r(x_1^{(k)})$ is bounded by $r(x_{\langle 1 \rangle})$, $r(x_1^{(k)})$ will thus converge to $r(x_{\langle 1 \rangle})$ after finite iterations. In other words, the neuron with the highest rank would be enabled. Because all other neurons have the ranks that are less than or equal to $r(x_{\langle 1 \rangle})$, according to cases III and IV, they will be disabled. The theorem is thus proved. ■

Theorem 2 (in words): Under HNN, a given constraint which assures exactly n neurons will be enabled when the HNN converges to the equilibrium state, then the specially designed RHNN (see below) subject to the same constraint assures n neurons associated with the n-highest ranks will be enabled when it converges to the equilibrium state.

Theorem 2: Assume constraint C requires *exactly* n neurons to be enabled, i.e., $C : \sum_x V_x = n$, where V_x is the output of neuron x. For the RHNN, if the neuron weight is given as $W'_{xy} = W_{xy} + A \times \{1 - u[r(y) - r(x)]\}$, where A is a pre-assigned coefficient of rank stimulation and $u(\cdot)$ is a unit step function, then exactly n neurons associated with the n-highest ranks will be enabled when the RHNN converges to the equilibrium state.

Proof: We first rewrite constraint C to $f_C \equiv [(\sum_x V_x) - n]^2 = 0$. By equating f_C to the energy function E in (3), we obtain $W_{xy} = -2$ and $\theta_x = -(2n-1)$. For RHNN, assigning $A = 2$, rendering $W'_{xy} = -2 + 2\{1 - u[r(y) - r(x)]\}$, we arrive at

$$\begin{aligned} \mathrm{net}'^{(k)}_x &= \sum_{y \neq x} W'_{xy} V_y^{(k)} - \theta_x \\ &= -2 \sum_{y \neq x} V_y^{(k)} + 2 \sum_{r(x) > r(y)} V_y^{(k)} + (2n-1) \\ &= -2 \sum_{y \neq x, r(x) \leq r(y)} V_y^{(k)} + (2n-1) \end{aligned}$$

at iteration k. To prove that n neurons with the n-highest ranks will be enabled and other neurons will be disabled, we need to show that $r(x_n^{(k)})$ is monotonically increasing $(r(x_n^{(k+1)}) \geq r(x_n^{(k)}))$ and converges to $r(x_{\langle n \rangle})$. To this end, we consider four following cases.

Case I: We assume there are less than n enabled neurons at iteration k, i.e., $\{x_n^{(k)}\} = \emptyset$. Since $\sum_{y \neq x, r(x) \leq r(y)} V_y^{(k)} \leq n-1$, we have $\mathrm{net}'^{(k)}_x \geq -2(n-1) + (2n-1) = 1$, namely neuron x will be enabled at iteration $k+1$ (regardless of its rank). Focusing on the rank, if there exists $n-1$ enabled neurons and x is disabled at iteration k, then the enabling of x yields $r(x_n^{(k+1)}) > r(x_n^{(k)})$. Otherwise, we have $r(x_n^{(k+1)}) = r(x_n^{(k)})$.

Case II: We assume there are at least n enabled neurons at iteration k (i.e., $\{x_n^{(k)}\} \neq \emptyset$) and $r(x) > r(x_n^{(k)})$. Since the maximum number of enabled neurons with ranks being equal to or higher than $r(x)$ is $n-1$, we thus have $\sum_{y \neq x, r(x) \leq r(y)} V_y^{(k)} \leq n-1$, and $\mathrm{net}'^{(k)}_x \geq -2(n-1) + (2n-1) = 1$, indicating that x will be enabled at iteration $k+1$. Now, if x has already been enabled at iteration k, the neuron with the n_{th} highest rank remains the same, i.e., $r(x_n^{(k+1)}) = r(x_n^{(k)})$. Otherwise, the enabling of neuron x results in the increase of the n_{th} highest rank in the enabled-neuron set, namely $r(x_n^{(k+1)}) > r(x_n^{(k)})$.

Case III: We assume there are at least n enabled neuron at iteration k (i.e., $\{x_n^{(k)}\} \neq \emptyset$) and $r(x) = r(x_n^{(k)})$. Note that x can be either enabled or disabled at iteration k. If x is enabled and there are less than n enabled neurons (excluding x) with ranks being equal to or higher than $r(x)$, then $\sum_{y \neq x, r(x) \leq r(y)} V_y^{(k)} \leq n-1$, implying $\mathrm{net}'^{(k)}_x \geq -2(n-1) + (2n-1) = 1$. Thus, x will stay enabled at iteration $k+1$; and $r(x_n^{(k+1)}) = r(x) = r(x_n^{(k)})$. However, if there are at least n enabled neurons (excluding x) with rank being equal to or higher than $r(x)$, then we have $\sum_{y \neq x, r(x) \leq r(y)} V_y^{(k)} \geq n$, and $\mathrm{net}'^{(k)}_x \leq -2n + (2n-1)$, which implies x will be disabled at iteration $k+1$. Since those other neurons with the rank equal to $r(x_n^{(k)})$ remained enabled, $r(x_n^{(k+1)})$ is unchanged, i.e., $r(x_n^{(k+1)}) = r(x_n^{(k)})$. On the other hand, if x is disabled at iteration k, we get $\sum_{y \neq x, r(x) \leq r(y)} V_y^{(k)} \geq n$. Thus, $\mathrm{net}'^{(k)}_x \leq -2n + (2n-1) \leq -1$, with the result that x will stay disabled and $r(x_n^{(k+1)}) = r(x_n^{(k)})$.

Case IV: We assume there are at least n enabled neuron at iteration k (i.e., $\{x_n^{(k)}\} \neq \emptyset$) and $r(x) < r(x_n^{(k)})$. Clearly, $\sum_{y \neq x, r(y) \geq r(x)} V_y^{(k)} \geq n$, regardless of the state of x at iteration k. Thus, $\mathrm{net}'^{(k)}_x \leq -2n + (2n-1) = -1$, yielding that x will be disabled at iteration $k+1$. However, since $r(x) < r(x_n^{(k)})$, the state change of x is irrelevant to $r(x_n^{(k+1)})$, which implies $r(x_n^{(k+1)}) = r(x_n^{(k)})$.

According to the four cases, we have proved that $r(x_n^{(k)})$ is monotonically increasing. Since $r(x_n^{(k)})$ is bounded by $r(x_{\langle n \rangle})$, $r(x_n^{(k)})$ will eventually converge to $r(x_{\langle n \rangle})$. It's worth noting that as $r(x_n^{(k)})$ converge to $r(x_{\langle n \rangle})$, all neurons with rank higher than or equal to $r(x_{\langle n \rangle})$ will also be enabled. Once $r(x_n^{(k)})$ converges, the n highest-rank neurons will stay enabled at all iterations. For any remaining neuron x, since $r(x) \leq r(x_n^{(k)})$, according to cases III and IV, they will be disabled. The theorem is thus proved. ■

It's worth noting that enabling high-priority rank neurons may cause more than one neuron enabled in a contention/packet conflicting group, resulting in temporary increases in the energy function value. However, after finite neuron updates, the low-priority rank neuron will be disabled by the constraint, causing that the energy again decreases and eventually stabilize to its equilibrium state. Based on Theorems 1 and 2, in the next subsection, we propose the RHNN scheduler for the QoS scheduling problem.

We next show how RHNN can be implemented by analog circuit, namely in parallel. To this end, we show that the discrete-time form of RHNN can easily be transformed to a continuous-time form. Let the evolving rule of a neuron in the continuous RHNN be $C_x(dS_x/dt) = \sum_{y \neq x} W'_{xy} V_y - G'_x S_x - \theta_x$, where C_x is an input capacitance, S_x is an input voltage, $V_x = a(\lambda S_x)$ is the output of neuron $V_x, a(\lambda S_x) = 1/(1 + \exp(-\lambda S_x))$ is the activation function, $\lambda > 0$ is a gain parameter, $G'_x \equiv \sum_{y\neq x} W'_{xy} + g_x$, where g_x is an input conductance, and $W'_{xy} = W_{xy} + A_{xy}$ is a ranked neuron weight. Rank stimulation $A_{xy} \geq 0$ when $r(y) < r(x)$; and $A_{xy} = 0$ otherwise. By making $C_x \to 0$ and $\lambda \to \infty$, we have $G'_x S_x \to \sum_{y\neq x} W'_{xy} V_y - \theta_x$ and

$$V_x = a(\lambda S_x) \to a\left(\frac{\lambda}{G'_x} \times \left[\sum_{y\neq x} W'_{xy} V_y - \theta_x\right]\right) \to u\left(\sum_{y\neq x} W'_{xy} V_y - \theta_x\right)$$

i.e., the discrete behavior of RHNN with asynchronous update is logically identical to a continuous behavior of an RHNN.

B. RHNN for QoS Scheduling Problem

For an RHNN, let neuron output, neuron weight, and neuron threshold, be represented by $Y_{(I,S_1,S_2,S_3,O,\lambda)}$, $W_{(I,S_1,S_2,S_3,O,\lambda),(I',S'_1,S'_2,S'_3,O',\lambda')}$, $\theta_{(I,S_1,S_2,S_3,O,\lambda)}$, respectively. Then, the energy function can be expressed as

$$E = -\frac{1}{2} \sum_{\substack{(I',S'_1,S'_2,S'_3,O',\lambda') \\ \neq (I,S_1,S_2,S_3,O,\lambda)}} W_{\substack{(I,S_1,S_2,S_3,O,\lambda), \\ (I',S'_1,S'_2,S'_3,O',\lambda')}} \times Y_{(I,S_1,S_2,S_3,O,\lambda)} Y_{(I',S'_1,S'_2,S'_3,O',\lambda')} + \sum_{(I,S_1,S_2,S_3,O,\lambda)} \theta_{(I,S_1,S_2,S_3,O,\lambda)} Y_{(I,S_1,S_2,S_3,O,\lambda)}. \quad (4)$$

For constraint CF, by equating E in (4) with $f(\mathrm{CF})$ in (1), we can obtain the basic neuron weight (W^1) and threshold (θ^1) as shown in (5), at the bottom of this page, where $T((I,S_1,S_2,S_3,O,\lambda),(I',S'_1,S'_2,S'_3,O',\lambda'))$ is defined at the bottom of the page.Equation (5) implies that if two neurons/paths $(I,S_1,S_2,S_3,O,\lambda)$ and $(I',S'_1,S'_2,S'_3,O',\lambda')$ belong to the same contention conflicting group, then the basic weights connecting the two neurons are inhibitor-oriented, i.e.,

$$W^1_{\substack{(I,S_1,S_2,S_3,O,\lambda), \\ (I',S'_1,S'_2,S'_3,O',\lambda')}} = W^1_{\substack{(I',S'_1,S'_2,S'_3,O',\lambda'), \\ (I,S_1,S_2,S_3,O,\lambda)}} = -1.$$

By the same token, for constraint SA, we can obtain the basic neuron weight (W^2). However, due to the existing of the square term $\sum_{I,S_1,O} n^2(I,S_1,O)$ in (2), which violates the standard form of energy function, $f(\mathrm{SA})$ in (2) is first transformed into

$$f(SA) = \sum_{\substack{(I,S_1,O)=(I',S'_1,O') \\ (S_2,S_3,\lambda)\neq(S'_2,S'_3,\lambda')}} Y_{(I,S_1,S_2,S_3,O,\lambda)} \times Y_{(I',S'_1,S'_2,S'_3,O',\lambda')} + \sum_{(I,S_1,S_2,S_3,O,\lambda)} [1-2n(I,S_1,O)] \times Y_{(I,S_1,S_2,S_3,O,\lambda)}. \quad (6)$$

By letting $E = f(\mathrm{SA})$, the neuron weights (W^2) and thresholds (θ^2) can be derived as

$$\begin{cases} W^2_{\substack{(I,S_1,S_2,S_3,O,\lambda), \\ (I',S'_1,S'_2,S'_3,O',\lambda')}} = -2 \times \delta(I,I')\delta(S_1,S'_1)\delta(O,O') \\ \theta^2_{(I,S_1,S_2,S_3,O,\lambda)} = 1 - 2 \times n(I,S_1,O) \end{cases} \quad (7)$$

where

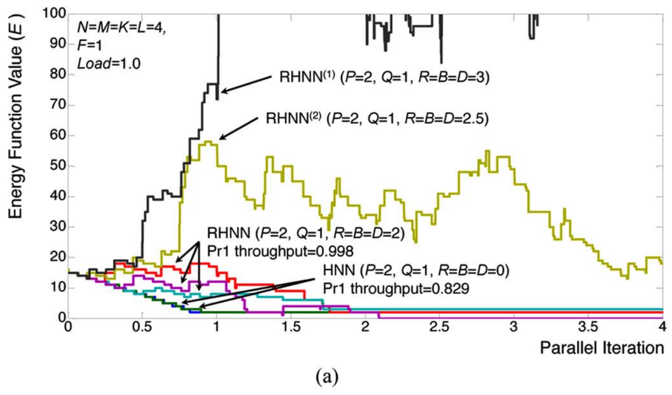
$$\delta(t,t') = \begin{cases} 1, & \text{if } t = t' \\ 0, & \text{if } t \neq t' \end{cases}.$$

Equation (7) indicates that if two neurons/paths belong to the same packet conflicting group, then the basic weights connecting the two neurons are inhibitor-oriented, i.e., $W^2 = -2\delta(I,I')\delta(S_1,S'_1)\delta(O,O') = -2$. Notice that both $f(\mathrm{CF})$ and $f(\mathrm{SA})$ are non-negative functions and their potentially minimum values are zero. Due to the decreasing and convergence properties of the energy function, $f(\mathrm{CF})$ and $f(\mathrm{SA})$ will then converge to zero at the end of the iterative process.

Next, we are to determine the rank stimulations. With two priority-oriented constraints, we designate packet priority $pr(I,S_1)$ and buffering delay $d(O,\lambda)$ as two rank functions. Recall that there is only one rank function in Theorems 1 and 2. To apply Theorems 1 and 2, we give preference to packet priority over buffering delay, and consider one rank function at a time. Ultimately, the final neuron weight (W) from $(I',S'_1,S'_2,S'_3,O',\lambda')$ to $(I,S_1,S_2,S_3,O,\lambda)$, and neuron threshold (θ) can be given as shown in (8), at the bottom of the next page, where P,Q,R,B,D are the coefficients (corresponding to the constant, A, in Theorems 1 and 2). Notice that the third term at the right-hand side of (8) is the rank stimulation associated with packet priority. Specifically, if the two packets are in the same contention conflicting group, i.e.,

$$\begin{cases} W^1_{\substack{(I,S_1,S_2,S_3,O,\lambda), \\ (I',S'_1,S'_2,S'_3,O',\lambda')}} = -T((I,S_1,S_2,S_3,O,\lambda),(I',S'_1,S'_2,S'_3,O',\lambda')) \\ \theta^1_{(I,S_1,S_2,S_3,O,\lambda)} = bfs(S_3,O,\lambda) \end{cases} \quad (5)$$

$$\begin{matrix} T((I,S_1,S_2,S_3,O,\lambda), \\ (I',S'_1,S'_2,S'_3,O',\lambda')) \end{matrix} = \begin{cases} 1, & \text{if } (I,S_1,S_2,S_3,O,\lambda),(I',S'_1,S'_2,S'_3,O',\lambda') \in G \\ 0, & \text{if } (I,S_1,S_2,S_3,O,\lambda),(I',S'_1,S'_2,S'_3,O',\lambda') \notin G \end{cases}$$



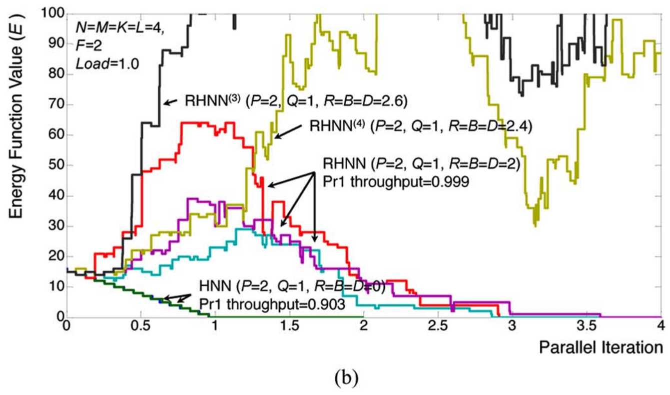


Fig. 4. Convergence behavior of the energy function. (a) Iteration traces under a smaller buffer. (b) Iteration traces under a larger buffer.

$T((I,S_1,S_2,S_3,O,\lambda),(I',S'_1,S'_2,S'_3,O',\lambda')) = 1$, from Theorem 1, the rank stimulation is $R \times \{1 - u[pr(I',S'_1) - pr(I,S_1)]\}$; otherwise the priority effect is nullified. The fourth term is the rank stimulation associated with buffering delay if the two packets have the same priority, i.e., $\delta(pr(I',S'_1),pr(I,S_1)) = 1$, and are in the same contention conflicting group. Furthermore, under the SA constraint, since all paths in the same packet conflicting group are for one packet, we only need to regard the buffering delay $d(O,\lambda)$ as the rank function when applying Theorem 2. This directly yields the last term of (8). Finally, the convergence of the RHNN scheduler is largely dependent on the determination of five coefficients, P,Q,R,B,D. Through extensive derivation (see Appendix), we arrive at one coefficient correlation, $P = 2Q = R = B = D > 0$, with which the RHNN scheduler converges to the optimal solution. This correlation of coefficients is also used for our simulation.

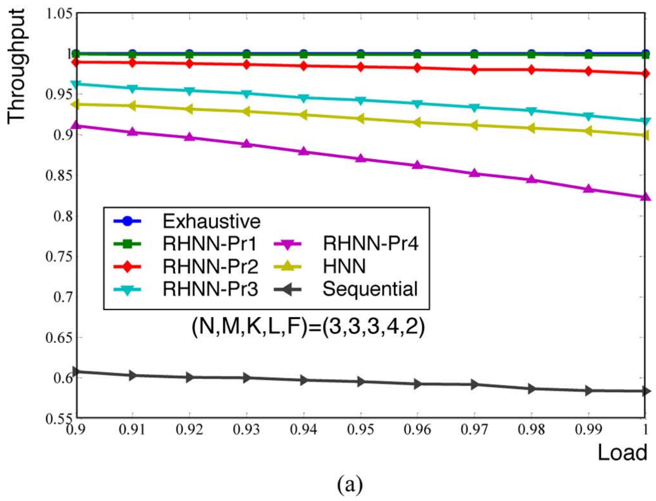
V. Simulation Results

We first demonstrate the convergence of the RHNN scheduler under different coefficients settings via simulation results. In the simulation, we assume that there are a total of $N \times M$ input packet flows entering into WOPIS at each system time slot, where N is the total number of input ports in each first-stage switch; and M is the total number of switching elements in the first stage. Define *Load* as the ratio of the mean number of newly-arriving packets to the total number of input ports in TSOPS $(N \times M)$; and traffic burstiness as the ratio of peak arrival rate to the mean arrival rate. Each packet flow arrival is generated by a two-state (ON and OFF) Interrupted Bernoulli Process (IBP) distribution. Specifically, the state transition probability from ON to OFF is 0.1, and from OFF to ON is $\mathrm{load}/10(1-\mathrm{load})$, resulting in a burstiness of $1/\mathrm{load}$. Finally, *system throughput* is defined as the ratio of the mean number of successfully scheduled packets to $N \times M$.

In Fig. 4, we show the simulation result of the convergence behaviors of the energy function. In the figure, we display the random iteration traces from HNN and our RHNN schedulers under different settings of coefficients, over a number of parallel iterations. It is worth noting that, since we carry out the simulation using a sequential computer, only one neuron is updated per each iteration. We recorded the total number of sequential iterations. The number of parallel iterations is obtained by dividing the total number of sequential iterations by the total number of neurons. The iteration-trace set for a buffer size of 4 $(F = 1)$ and 8 $(F = 2)$ are plotted in Fig. 4(a) and (b), respectively.

In Fig. 4(a), we display three sets of curves: 1) non-QoS HNN scheduler with two traces; 2) QoS RHNN scheduler under the specially designed coefficients with three traces; and 3) QoS $\mathrm{RHNN}^{(1)}$ and $\mathrm{RHNN}^{(2)}$ schedulers under randomly selected coefficients. Note that, with the coefficients assigned as $P = 2Q = 2$ and $R = B = D = 0$, the HNN scheduler is a special form of RHNN scheduler. Its energy function is monotonically decreasing and quickly converges, achieving a system throughput of 0.829. Compare to HNN scheduler, due to the use of rank stimulation, our RHNN scheduler achieves higher system throughput (0.998) for higher-priority flows at an expense of slower convergence time. Specifically, the RHNN scheduler exhibits the energy function that temporally increases but eventually converges, which is conformed to Theorems 1 and 2. Moreover, using randomly selected coefficients, both $\mathrm{RHNN}^{(1)}$ and $\mathrm{RHNN}^{(2)}$ schedulers fail to converge, justifying

$$\begin{cases} W_{\substack{(I,S_1,S_2,S_3,O,\lambda),\\(I',S'_1,S'_2,S'_3,O',\lambda')}} = P \times W^1_{\substack{(I,S_1,S_2,S_3,O,\lambda),\\(I',S'_1,S'_2,S'_3,O',\lambda')}} + Q \times W^2_{\substack{(I,S_1,S_2,S_3,O,\lambda),\\(I',S'_1,S'_2,S'_3,O',\lambda')}} \\ \quad +R \times T((I,S_1,S_2,S_3,O,\lambda),(I',S'_1,S'_2,S'_3,O',\lambda')) \\ \qquad \times\{1 - u[pr(I',S'_1) - pr(I,S_1)]\} \\ \quad +B \times T((I,S_1,S_2,S_3,O,\lambda),(I',S'_1,S'_2,S'_3,O',\lambda')) \\ \qquad \times\delta(pr(I',S'_1),pr(I,S_1)) \times \{1 - u[d(O,\lambda) - d(O',\lambda')]\} \\ \quad +D \times \delta(I,I')\delta(S_1,S'_1)\delta(O,O') \times \{1 - u[d(O,\lambda) - d(O',\lambda')]\} \\ \theta_{(I,S_1,S_2,S_3,O,\lambda)} = P \times \theta^1_{(I,S_1,S_2,S_3,O,\lambda)} + Q \times \theta^2_{(I,S_1,S_2,S_3,O,\lambda)} \end{cases} \tag{8}$$



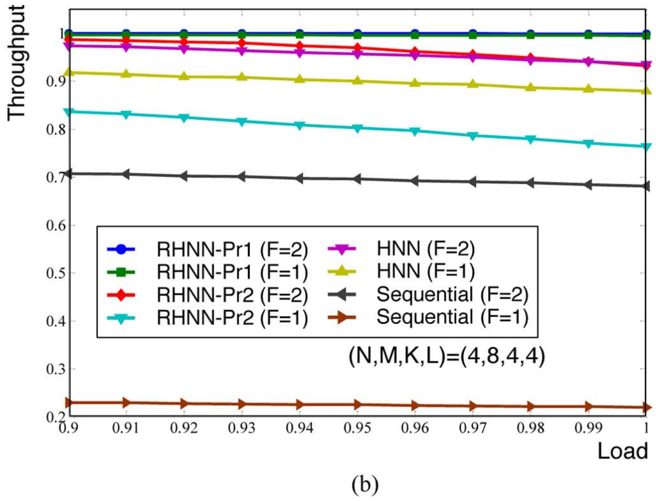
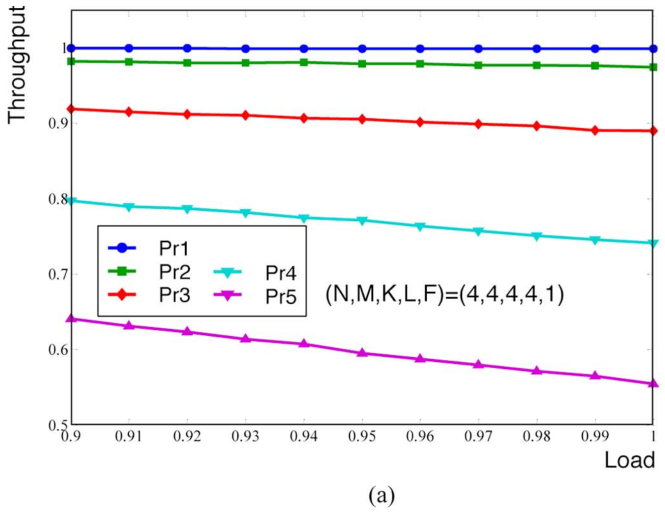


Fig. 5. System throughput comparision. (a) Throughput under a smaller-size TSOPS. (b) Throughput under a larger-size TSOPS.



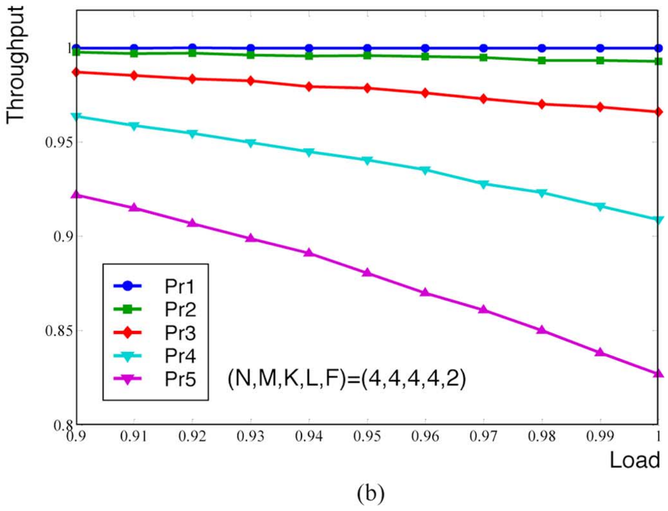


Fig. 6. QoS (five priority) differentiation of the RHNN parallel scheduler. (a) QoS differentiation under a smaller-size buffer. (b) QoS differentiation under a larger-size buffer.

our design in the previous section. Under a larger buffer size as shown in Fig. 4(b), we observe that the energy function also converges but taking slightly more parallel iterations.

We next draw comparisons of complexity and system throughput between the RHNN scheduler and two other packet scheduling algorithms—exhaustive (optimal), and sequential methods. The exhaustive method returns an optimal solution by testing all possible path combinations for newly-arriving packets, while the sequential method performs scheduling one packet at a time by searching the contention-free and minimum-delay paths sequentially. With L internal wavelengths and K second-stage switches, the exhaustive method requires a computational complexity of $O((KML)^{NM})$. The sequential method performs sequential check, requiring a complexity of $O(N^2M^3KL)$. Through experiments, our RHNN scheduler requires no more than ten parallel iterations (before it converges) that is nearly irrelevant to switch size, the numbers of internal wavelengths and FDLs. Consider a WOPIS with 10-Gb/s per wavelength and a packet size of 1000 bytes, each system slot time is 0.8 μsec long. To accommodate 20 parallel iterations within 0.8 μsec, the clock speed of a VLSI implementation of RHNN is 25 MHz, which is attainable by the current VLSI technology.

We show in Figs. 5 and 6 the system throughput based on four methods- RHNN, exhaustive, HNN, and sequential, under different switch sizes, FOB sizes, and priority levels. In Fig. 5(a), we adopt the TSOPS of a small size, i.e., $N = M = K = 3, L = 4$, and $F = 2$. The use of such a small-size TSOPS is because the exhaustive method fails to attain system throughput due to unmanageable complexity for any size larger than this. For the RHNN method, there are four priorities of packets. The results in Fig. 5(a) show that, while the sequential method yields the worst throughput among the four methods, our RHNN scheduler achieves a nearly 100% throughput for the highest-priority, which is as superior as that of the exhaustive method. Without the priority differentiation feature, HNN results in poorer throughput than RHNN for all three priorities, and yields slightly improved throughput for the lowest priority. In Fig. 5(b), we display the throughput based on the RHNN (two priorities) and sequential methods, for a larger TSOPS that is of size 32-by-32 ($N = 4, M = 8$) under heavy load condition. In this simulation, we set $L = 4$ and $F = 1$ and 2, yielding buffer sizes of 4 and 8, respectively. The results show that the RHNN outperforms the sequential method on the throughput for both priorities of packets.

In Fig. 6, we display the performance of RHNN scheduler with respect to QoS differentiation among five different priorities and two different optical buffer sizes. Results in Fig. 6(a) indicate that RHNN scheduler can achieve guaranteed throughput for higher-priority packets even under a handful of optical buffers ($\text{buffer size} = 4$). From both Fig. 6(a) and (b), we show that the RHNN scheduler provides superlatively effective priority-based QoS differentiation with respect to system throughput. Crucially, the higher-priority packets invariably receive almost 100% throughput, regardless of the traffic load.

VI. Conclusion

In this paper, we have proposed a new parallel scheduler, RHNN, for a WDM optical interconnection system, WOPIS. With ranked neurons, RHNN is capable of resolving the QoS scheduling problem, specified by four constraints (*CF*, *SA*, *PF*, and *MD*). Based on two theorems, we determine the neuron weights and coefficients subject to the convergence of the RHNN scheduler to the optimal solution. Simulation results show that the RHNN achieves as superlatively high throughput (almost 100%) as that of the optimal method for higher-priority packets. The computation time is as short as 0.8 μsec for a WOPIS operating at 10-Gb/s per wavelength. Significantly, the RHNN scheduler achieves effective priority-based QoS differentiation, with the result that the higher-priority packets invariably receive almost 100% throughput regardless of any increase in traffic load.

Appendix

We are to determine the coefficients in (8) by considering the state change of a given path/neuron, $x = (I, S_1, S_2, S_3, O, \lambda)$, from iteration, say k, to $k + 1$. From (5), (7), and (8), we have

$$\begin{cases} W_{x,(I',S'_1,S'_2,S'_3,O',\lambda')} \\ = P \times -T(x,(I',S'_1,S'_2,S'_3,O',\lambda')) \\ \quad +Q \times -2 \times \delta(I,I')\delta(S_1,S'_1)\delta(O,O') \\ \quad +R \times T(x,(I',S'_1,S'_2,S'_3,O',\lambda')) \\ \quad \times\{1-u[pr(I',S'_1)-pr(I,S_1)]\} \\ \quad +B \times T(x,(I',S'_1,S'_2,S'_3,O',\lambda')) \\ \quad \times\delta(pr(I',S'_1),pr(I,S_1)) \\ \quad \times\{1-u[d(O,\lambda)-d(O',\lambda')]\} \\ \quad +D \times \delta(I,I')\delta(S_1,S'_1)\delta(O,O') \\ \quad \times\{1-u[d(O,\lambda)-d(O',\lambda')]\} \\ \theta_x = P \times bfs(S_3,O,\lambda) + Q \times \{1-2\times n(I,S_1,O)\}. \end{cases} \tag{9}$$

At iteration k, suppose there are α enabled neurons that belong to the same contention conflicting group of neuron x; there are β enabled neurons that belong to the same contention conflicting group and have the same packet priority as x; and there are γ enabled neurons that belong to the same packet conflicting group of x. Since $T(x,(I',S'_1,S'_2,S'_3,O',\lambda')) = 1$ if x and $(I',S'_1,S'_2,S'_3,O',\lambda')$ belong to the same contention conflicting group; and $\delta(I,I')\delta(S_1,S'_1)\delta(O,O') = 1$ if x and $(I',S'_1,S'_2,S'_3,O',\lambda')$ belong to the same packet conflicting group, then from (9) we attain

$$\begin{aligned} \text{net}_x^{(k)} \equiv & \sum_{(I',S'_1,S'_2,S'_3,O',\lambda')\neq x} W_{x,(I',S'_1,S_{2'},S'_3,O',\lambda')} \\ & \times Y^{(k)}_{(I',S_{1'},S'_2,S_{3'},O',\lambda')} - \theta_x \\ = & P \times (-\alpha) + Q \times (-2\gamma) \\ & + R \times (\alpha-\beta) \\ & \times \{1-u[pr(I',S'_1)-pr(I,S_1)]\} \\ & + B\beta \times \delta(pr(I',S'_1),pr(I,S_1)) \\ & \times \{1-u[d(O,\lambda)-d(O',\lambda')]\} \\ & + D\gamma \times \{1-u[d(O,\lambda)-d(O',\lambda')]\} \\ & - P \times [bfs(S_3,O,\lambda)] \\ & - Q \times [1-2\times n(I,S_1,O)]. \end{aligned} \tag{10}$$

Moreover, path x should be enabled, i.e., $\text{net}_x^{(k)} \geq 0$, iff the following five conditions are simultaneously satisfied: (a) $bfs(S_3,O,\lambda) = 0$ (the corresponding buffer space is unoccupied); (b) $n(I,S_1,O) = 1$ (there exists an in-out request for x); (c) $1-u[pr(I',S'_1)-pr(I,S_1)] = 1$ (x has the highest packet priority); (d) $\delta(pr(I,S_1),pr(I',S'_1)) \times \{1-u[d(O,\lambda) - d(O',\lambda')]\} = 1$ (x has the smallest buffering delay among those of the same priority); and (e) $1-u[d(O,\lambda)-d(O',\lambda')] = 1$ (x has the smallest buffering delay). Therefore,

$$\begin{aligned} \text{net}_x^{(k)} = -\alpha P - (2\gamma-1)Q + (\alpha-\beta)R + \beta B + \gamma D \geq 0, \\ \text{for } \alpha \geq 0, \alpha-\beta \geq 0, \text{ and } \gamma \geq 0. \end{aligned} \tag{11}$$

On the other hand, path x should be disabled, i.e., $\text{net}_x^{(k)} < 0$, if any one of the above five conditions (a)–(e) is not satisfied. Accordingly, from (10) we get

$$\begin{aligned} \text{net}_x^{(k)} = -(\alpha+1)P-(2\gamma-1)Q+(\alpha-\beta)R+\beta B+\gamma D < 0, \\ \text{for } \alpha \geq 0, \alpha-\beta \geq 0, \text{ and } \gamma \geq 0 \end{aligned} \tag{12}$$

$$\begin{aligned} \text{net}_x^{(k)} = -\alpha P-(2\gamma+1)Q+(\alpha-\beta)R+\beta B+\gamma D < 0, \\ \text{for } \alpha \geq 0, \alpha-\beta \geq 0, \text{ and } \gamma \geq 0 \end{aligned} \tag{13}$$

$$\begin{aligned} \text{net}_x^{(k)} = -\alpha P-(2\gamma-1)Q+(\alpha-\beta-1)R+\beta B+\gamma D < 0, \\ \text{for } \alpha \geq 0, \alpha-\beta \geq 1, \text{ and } \gamma \geq 0 \end{aligned} \tag{14}$$

$$\begin{aligned} \text{net}_x^{(k)} = -\alpha P-(2\gamma-1)Q+(\alpha-\beta)R+(\beta-1)B+\gamma D < 0, \\ \text{for } \alpha-\beta \geq 0, \beta \geq 1, \text{ and } \gamma \geq 0 \end{aligned} \tag{15}$$

$$\begin{aligned} \text{net}_x^{(k)} = -\alpha P-(2\gamma-1)Q+(\alpha-\beta)R+\beta B+(\gamma-1)D < 0, \\ \text{for } \alpha \geq 0, \alpha-\beta \geq 0, \text{ and } \gamma \geq 1. \end{aligned} \tag{16}$$

Notice that if condition (c) is unsatisfied, we attain $(\alpha-\beta)R \times \{1-u[pr(I',S'_1)-pr(I,S_1)]\} \leq (\alpha-\beta-1)R$, leading to (14). In addition, if conditions (d) and (e) are not satisfied, we get $\beta R \times \{1-u[d(O,\lambda)-d(O',\lambda')]\} \leq (\beta-1)R$, and $\gamma D \times \{1-u[d(O,\lambda)-d(O',\lambda')]\} \leq (\gamma-1)D$, yielding inequalities (15) and (16), respectively. By solving (11) to (16), we arrive at a sufficient condition, $P = 2Q = R = B = D > 0$, for RHNN to converge to the optimal solution.

REFERENCES

[1] A. K. Kodi and A. Louri, "Multidimensional and reconfigurable optical interconnects for high-performance computing (HPC) systems," *J. Lightw. Technol.*, vol. 27, no. 21, pp. 4634–4641, Nov. 2009.

[2] Z. Zhang and Y. Yang, "WDM optical interconnects with recirculating buffering and limited range wavelength conversion," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 5, pp. 466–480, May 2006.

[3] M. Yuang, Y. Lin, J. Shih, J. J. Chen, P. Tien, S. S. W. Lee, and S. Lin, "A QoS optical packet switching system: Architectural design and experimental demonstration," *IEEE Commun. Mag.*, vol. 48, no. 5, pp. 66–75, May 2010.

[4] S. Liew, G. Hu, and H. Chao, "Scheduling algorithms for shared fiber-delay-line optical packet switches—Part II: The three-stage CLOS-network case," *J. Lightw. Technol.*, vol. 23, no. 4, pp. 1601–1609, Apr. 2005.

[5] B. Sarker, T. Yoshino, and S. Majumder, "All-optical wavelength conversion based on cross-phase modulation (XPM) in a single-mode fiber and a Mach–Zehnder interferometer," *IEEE Photon. Technol. Lett.*, vol. 14, no. 3, pp. 340–342, Mar. 2002.

[6] G. Papadimitriou, C. Papazoglou, and A. Pomportsis, "Optical switching: Switch fabrics, techniques, and architectures," *J. Lightw. Technol.*, vol. 21, no. 2, pp. 384–405, Feb. 2003.

[7] K. A. Williams, G. F. Roberts, T. Lin, R. V. Penty, I. H. White, M. Glick, and D. McAuley, "Integrated optical 2×2 switch for wavelength multiplexed interconnects," *IEEE J. Sel. Topics Quantum Electron.*, vol. 11, no. 1, pp. 78–85, Jan./Feb. 2005.

[8] T. Tanemura, M. Takenaka, A. A. Amin, K. Takeda, T. Shioda, M. Sugiyama, and Y. Nakano, "InP-InGaAsP integrated 1×5 optical switch using arrayed phase shifters," *IEEE Photon. Technol. Lett.*, vol. 20, no. 12, pp. 1063–1065, Jun. 2008.

[9] A. Jajszczyk, "Nonblocking, repackable, and rearrangable clos networks: Fifty years of the theory evolution," *IEEE Commun. Mag.*, vol. 41, no. 10, pp. 28–33, Oct. 2003.

[10] C. Lin and C. S. Lee*, Neural Fuzzy Systems A Neuro-Fuzzy Synergism to Intelligent Systems*. Upper Saddle River, NJ: Prentice-Hall, 1996.

[11] X. Gao and L. Liao, "A novel neural network for a class of convex quadratic minimax problems," *Neural Computing*, vol. 18, no. 8, pp. 1818–1846, Aug. 2006.

[12] X. Hu and J. Wang, "A recurrent neural network for solving a class of general variational inequalities," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 3, pp. 528–539, Jan. 2007.

[13] Q. Liu and J. Wang, "A one-layer recurrent neural network with a discontinuous hard-limiting activation function for quadratic programming," *IEEE Trans. Neural Netw.*, vol. 19, no. 4, pp. 558–570, Apr. 2008.

[14] X. Hu and B. Zhang, "A new recurrent neural network for solving convex quadratic programming problems with an application to the k-winners-take-all problem," *IEEE Trans. Neural Netw.*, vol. 20, no. 4, pp. 654–664, Apr. 2009.

[15] Y. Li, Z. Tang, G. Xia, and R. Wang, "A positively self-feedbacked hopfield neural network architecture for crossbar switching," *IEEE Trans. Circuits Syst.*, vol. 52, no. 1, pp. 200–206, Jan. 2005.

[16] X. Hu and J. Wang, "An improved dual neural network for solving a class of quadratic programming problems and its K-winners-take-all application," *IEEE Trans. Neural Netw.*, vol. 19, no. 12, pp. 2022–2031, Dec. 2008.

[17] T. P. Troudet and S. M. Walters, "Neural network architecture for crossbar switch control," *IEEE Trans. Circuits Syst.*, vol. 38, no. 1, pp. 42–56, Jan. 1991.

[18] K. Symington, A. Waddie, M. Taghizadeh, and J. Snowdon, "A neural-network packet switch controller: Scalability, performance, and network optimization," *IEEE Trans. Neural Netw.*, vol. 14, no. 1, pp. 28–34, Jan. 2003.

[19] M. Chia, D. Hunter, I. Andonovic, P. Ball, I. Wright, S. Ferguson, K. Guild, and M. O'Mahony, "Packet loss and delay performance of feedback and feed-forward arrayed-waveguide gratings-based optical packet switches with WDM inputs-outputs," *J. Lightw. Technol.*, vol. 19, no. 9, pp. 1241–1254, Sep. 2001.

Po-Lung Tien (M'10) received the B.S. degree in applied mathematics, M.S. degree in computer and information science, and Ph.D. degree in computer science and information engineering from the National Chiao Tung University, Hsinchu, Taiwan, in 1992, 1995, and 2000, respectively.

In 2005, he joined National Chiao Tung University, Hsinchu, Taiwan, where he is currently an Assistant Professor with the Department of Electrical Engineering. His current research interests include optical networking, computational intelligence, network optimization, multimedia communications, and performance modeling.

Bo-Yu Ke (S'10) received the B.S. degree in communication engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2009, where he is currently working toward the Ph.D. degree.

His research interests include computational intelligence and network optimization.