Noname manuscript No.
(will be inserted by the editor)

All-optical Tree-based Greedy Router Using Optical Logic Gates and Optical Flip-Flops

Sahel Sahhaf[1] · **Abhishek Dixit**[2] · **Wouter Tavernier**[1] · **Didier Colle**[1] · **Mario Pickavet**[1] · **Piet Demeester**[1]

Received: date / Accepted: date

Abstract Due to ever increasing throughput demands the lookup in conventional IP routers based on longest prefix matching is becoming a bottleneck. Additionally, the scalability of current routing protocols is limited by the size of the routing tables. Geometric greedy routing is an alternative to IP routing which replaces longest prefix matching with a simple calculation employing only local information for packet forwarding. For the first time, in this paper we propose a novel and truly all-optical geometric greedy router based on optical logic gates and optical flip-flops. The circuit of the router is constructed through the interconnection of SOAs and directional couplers. The successful functionality of the proposed router is verified through simulation. The circuit enables high data rate throughput.

Keywords All-optical router · SOA gate arrays · geometric greedy routing

1 Introduction

Internet traffic keeps growing and higher performance, capacity and forwarding rates are required in order to meet the ever increasing future demands. In addition, the growth of the size of the routing tables restricts the scalability of the current IP-based routing protocols. There are more than 600K Forwarding Information Base (FIB) entries in current Boarder Gateway Protocol (BGP) router reported in [9]. Geometric greedy routing has been proposed as an alternative to IP routing to solve the routing tables scalability issue.

[1]Department of Information Technology (INTEC), Ghent University-iMinds
[2]Department of Electrical Engineering, IIT Delhi

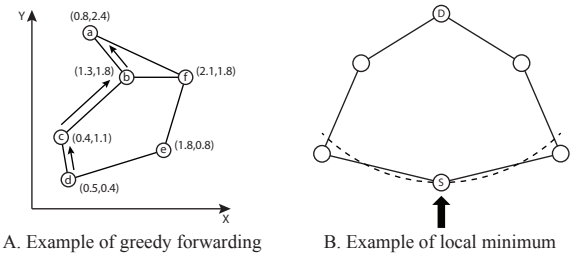

Fig. 1 (A) Example of greedy routing in Euclidean space (B) Example of local minimum

In geometric routing a coordinate is assigned to each node in the network. An obvious option is to attach GPS coordinates to the routers in a network [10]. Network nodes knowing the coordinates of their neighbors could forward the incoming packets to the neighbors which are closer to the packets destination. Following this distance-decreasing policy, the packet can eventually reach the destination. As in every step, the packet is forwarded to the neighbor with maximum decrease in the distance, the scheme is referred to as greedy routing/forwarding. The same idea has been used in [3] however, virtual coordinates are assigned instead. In this scheme, forwarding is only based on local information and thus it is significantly more memory-efficient compared to IP routing. Figure 1A depicts an example of greedy forwarding based on Euclidean coordinates. Starting from node 'd' towards 'a', first the Euclidean distance between the neighbors of 'd' and 'a' is calculated. Node 'c' is closer to 'a' compared to 'e'. Therefore, 'c' is selected as the next hop. The same calculation is performed in nodes 'c' and 'b' until node 'a' is reached.

In geometric routing address lookup, as in IP routing, is replaced with distance calculation. This distance calculation is performed only for a node's neighbors and thus is limited. As IP lookups at wire speed become

more difficult at higher bit rates, using geometric routing will potentially increase the routers speed and thus the throughput which is a significant requirement in the future Internet.

Our contribution. Due to the advances in optical technology, optical components are known to have an excellent potential to be included in high capacity routing systems. Therefore, in this paper, we propose an efficient and scalable all-optical geometric greedy router employing the coordinates derived from a spanning tree of the network. The router is constructed through interconnection of optical logic gates and flip-flops composed of Semiconductor Optical Amplifiers (SOA) and directional couplers. Using SOA gate arrays results in a scalable approach in terms of manufacturing.

It is worth mentioning that the performance of the tree-based greedy geometric routing scheme has been already evaluated in terms of routing quality and convergence behavior in case of network dynamics in our previous works [17,18]. The goal of this paper is to illustrate that a scalable all-optical greedy router can be implemented which enables high data rate throughput. We have implemented XOR, AND, OR gates, multiplexer, two types of flip-flops and more complex components such as counter and comparator to construct the proposed greedy forwarder. All of these components are validated through simulation and the successful functionality of the designed router is demonstrated. To the best of our knowledge, no research has been conducted so far investigating such a design. It is expected that with the advances in the photonic integration research, this router can be integrated in a single platform, which brings down the cost and the power consumption.

The rest of the paper is organized as follows. Section 2 gives an overview on existing work related to i) geometric routing and ii) all-optical designs. Section 3 explains the tree-based greedy routing for which the optical greedy forwarder is designed. The implementation of the designed router is explained in Section 4. The functionality of the proposed circuit is validated through simulation in Section 5. Section 6 discusses future plans. Finally, Section 7 concludes the paper.

2 Related work

A known issue in greedy routing is that packets may get stuck in a local minimum. This means that there is no neighbor closer to the destination than the current node. In Figure 1B node 'S' is an example of a local minimum. Greedy embeddings, as proposed in [13] avoid this situation by ensuring that coordinates are mapped to network nodes in such a way that for every node there is always a neighbor which decreases the distance towards any destination. There are several works which propose greedy embeddings in different metric spaces such as hyperbolic plane [11,1] and multidimensional Euclidean spaces [2,23]. Some works have proposed greedy embeddings based on one or several spanning trees of the network [11,20,6]. Most of the schemes based on multiple trees achieve good routing performance [6–8] at the cost of increased overall overhead and complexity.

Being a recent paradigm, most of the proposed schemes are theoretical works which may be too complex to be used in practice. In order to bridge the gap between geometric routing in theory and its applicability in practice, we propose a hardware design of a greedy forwarder based on a simple greedy embedding. In [17, 18], we investigated this embedding which is based on a single spanning tree. Despite its simplicity, good performance in terms of routing quality and convergence behavior was achieved. As optical technology has an excellent potential to be used in high capacity routing systems, we design the proposed greedy forwarder based on all-optical components. In our previous works [16,15] part of this forwarder, excluding the compound components (i.e., optical counter and comparator), was demonstrated. However, the complete design and the successful operation of the whole system are presented in this paper.

There exist several optical packet and burst switching technologies in the literature [24]. In [12] optical-label swapping technique (OLS) was considered. This work processed the label of a packet in electrical domain while the payload was forwarded through each node without optical-electrical-optical conversion. In other works such as [5,19] the label was also processed in the optical domain. In [14], the first all-optical label swapping router was proposed which performed switching and forwarding functionalities in the optical layer. In their design, the SOA-based MachZehnder Interferometer (SOA-MZI) blocks were used to construct the logic gates and flip-flops which leads to a flexible and scalable approach in terms of manufacturing. In [4] an all-optical switch architecture for Ethernet based on SOA and directional couplers was proposed which provides communication with and without IP. In their approach, the network is abstracted to a binary tree. This is achieved by adding virtual nodes so that every node can be characterized as a binary node. Having binary nodes is beneficial because the forwarding plane is simplified through enabling binary routing and source routing. However, this approach differs from our proposed tree-based geometric greedy routing in the sense that the routing is only on the tree and thus shortcut links[1] are not used.

[1] Links which are not in the spanning tree of the network.

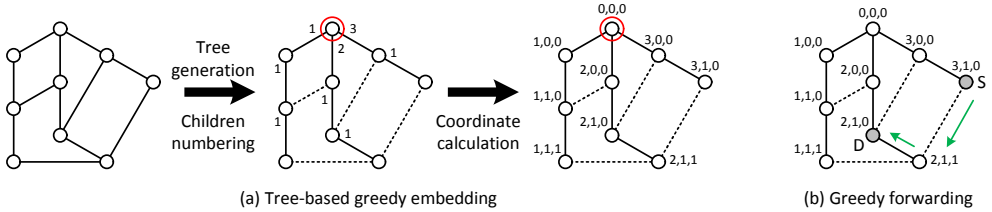

Fig. 2 (A) Tree-based greedy embedding. (B) Greedy forwarding based on tree coordinates

Such an approach leads to high stretch[2] and potential bandwidth bottleneck at the root of the tree. The latter refers to the forwarding of the packets towards the root to reach the intended destination. Up to now, none of the existing works have focused on the design of an all-optical greedy forwarder.

3 Greedy Tree-based geometric Routing (GTR)

In [17], we investigated GTR which is a geometric routing scheme based on a simple but powerful greedy embedding. GTR consists of two components: i) tree-based greedy embedding and ii) greedy forwarding. The former is responsible for constructing a spanning tree and deriving network nodes coordinates. These coordinates are then used by the latter to forward the packets towards the destinations.

In this section, we detail the tree-based greedy embedding and related greedy forwarding. In Section 4, we assume that the embedding has already taken place and nodes are aware of their neighbors coordinates. We then describe the design of different optical components required for the forwarding component of GTR.

Tree-based greedy embedding. In [17] we proposed a simple greedy embedding based on a spanning tree of the network. This scheme reduces the overall complexity and computational overhead. Based on our investigations, this simplification does not penalize the resulting performance. The steps for calculating nodes coordinates based on this embedding scheme are illustrated in Figure 2A: i) A rooted spanning tree of the network is generated. ii) Each node numbers its children from 1 to d (d is the number of children). iii) The root node sets its Coordinate Set (CS) to zero (0,...,0) and every node can calculate the CS of its children by putting the number assigned to each child in place of the first zero coordinate in its own CS. The number of coordinates in the CSs is determined by the depth of the tree.

Greedy forwarding based on Coordinate Sets (CS). Once the CSs are calculated, each node, knowing the CSs of its neighbors, forwards an incoming packet to the neighbor which maximally decreases the distance towards the packet destination. In this context, we use tree-distance as metric which is the hop count on the tree between the nodes: i) first the closest common ancestor to both nodes is found, ii) then the hop counts of each node to the common ancestor is counted, iii) the sum of these two hop counts determines the tree-distance between them. Given the CSs of two nodes, the largest common prefix of the two CSs determines the CS of the closest common ancestor. In each CS, the number of non-zero coordinates after the common prefix determines the hop count of that node to the common ancestor. The sum of the hop counts determines the tree-distance. In case of no common prefix (e.g. nodes (1,1,1) and (2,1,1) in Figure 1A), the common ancestor is the root node (0,0,0) and all the non-zero coordinates of each CS are counted.

Note that in the greedy routing based on the derived CSs the shortcut links are also used which is different from the tree routing[3]. The spanning tree is only used to guarantee a distance decreasing path towards the destination.

Figure 2B illustrates an example of greedy forwarding from S=(3,1,0) to D=(2,1,0) based on the deduced CSs. In node S, the tree-distances (td) are as follows: td((3,0,0),(2,1,0))=3, td((2,1,1),(2,1,0))=1. Therefore, (2,1,1) is selected as the next hop. The path generated by the greedy forwarding is depicted by arrows. This path is much more efficient than tree-routing ($(3,1,0) \rightarrow (3,0,0) \rightarrow (0,0,0) \rightarrow (2,0,0) \rightarrow (2,1,0)$).

4 All-optical tree-based greedy router

4.1 General architecture of the optical greedy router

A greedy router consists of two main components: i) tree-distance calculator and ii) comparator. The former calculates the tree-distance between the CS of a neighbor and the CS of the destination of an incoming packet. The latter selects the neighbor with minimum tree-distance as the next hop.

[2] Stretch is defined as deviation from the shortest path length.

[3] Tree routing refers to a routing when only the tree edges are used.

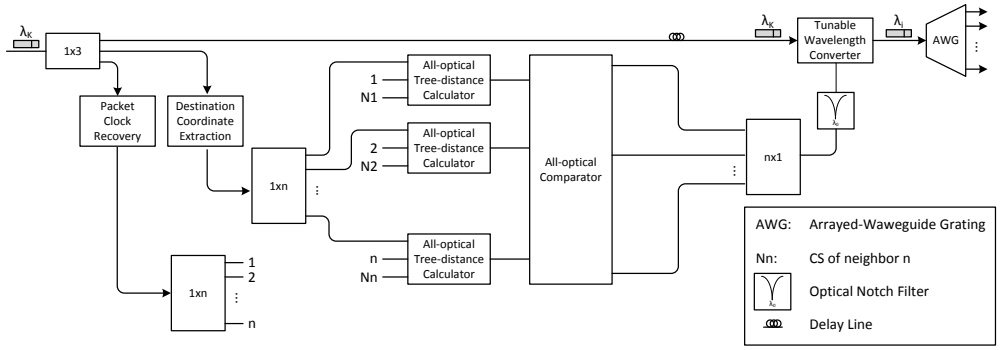

Fig. 3 General architecture of optical greedy router

Figure 3 depicts the architecture of the proposed all-optical greedy router with n ports. For the sake of clarity, the architecture corresponding to one input port is illustrated which can be simply extended for more ports. As we see in the figure, there exists a tree-distance calculator corresponding to each port. Once the tree-distances are calculated one comparator is required to select the next hop.

Upon arrival of a packet in the router, the CS of the destination is extracted from the packet (similar to label extraction in [14]), as shown in Figure 3. The extracted optical CS is then fed to All-optical Tree-distance Calculators. The calculated distance between each neighbor and the destination is then fed to the All-optical Comparator. Note that the neighbors' CSs (N1,...,Nn) are stored in the router and are the inputs of the tree-distance calculators. The comparator produces a high intensity signal at a certain wavelength at the output corresponding to the neighbor with minimum distance. This signal drives a wavelength converter. The packet is then converted to the signal's wavelength and is sent through an arrayed-waveguide grating (AWG). Therefore, the wavelength of the packet determines the outgoing port on which the packet leaves the node.

Optical data processing requires the existence of an all-optical synchronization stage at the input of the node. This synchronization is required to control/power-up subsequent node subsystems. [14] proposed mechanisms for this synchronization. In the absence of such self-synchronization modules, local signal generation or synchronization of incoming packets with local optical oscillators are required. We do not focus on the design of this module and assume the existence of a synchronization mechanism (Packet Clock Recovery block in Figure 3 refers to such module). The captured packet clock signal is then fed to the tree-distance calculator blocks (see Figure 3).

For the sake of directed focus on the all-optical functionalities related to greedy forwarding, the implementation and simulation validations are restricted to the tree-distance calculator and next hop selector (i.e., the All-optical Tree-distance Calculator and All-optical Comparator blocks in Figure 3). In order to support other functionalities required in a router (e.g., destination coordinate extraction from the packet, packet-clock recovery and tunable wavelength conversion as shown in Figure 3) we rely on existing approaches e.g., [14].

4.2 All-optical tree-distance calculator

Figure 4 depicts the proposed circuit for the tree-distance calculator. The inputs of the this block are i) CS of the destination (D), ii) CS of a neighbor (N) and iii) the clock signal (clock). In this figure, the values at different stages of the architecture are marked for sample CSs including 3 coordinates each composed of 2 bits.

Given two CSs, this circuit identifies the first uncommon coordinate and starts counting the non-zero coordinates in both CSs. As we see, the circuit is composed of all-optical gates (XOR, AND, OR), multiplexer, flip-flop and a counter. As the final value is set in the flip-flops of the counter, a parallel to serial circuit is required because the all-optical comparator (see Figure 3) receives serial input signals.

The inputs of the circuit (N and D) are serial bits which are fed to an XOR gate to identify the first uncommon bits in the two CSs. The output of the XOR is fed to an OR gate together with a feedback from a flip-flop. This leads to a signal which is set to '1' from the moment the first uncommon bit is identified and remains '1' up to the last bit of the CS. This signal determines when the counting of non-zero coordinates should begin. As the non-zero coordinates in both CSs should be counted, this signal is replicated through a multiplexer and a fiber delay element. The duration of the fiber delay is equal to the length of a CS (bits). We generate the same signal twice and thus the length of the output signal of the multiplexer is twice the length of a CS. With such signal we can count the hop counts of the two CSs consecutively.

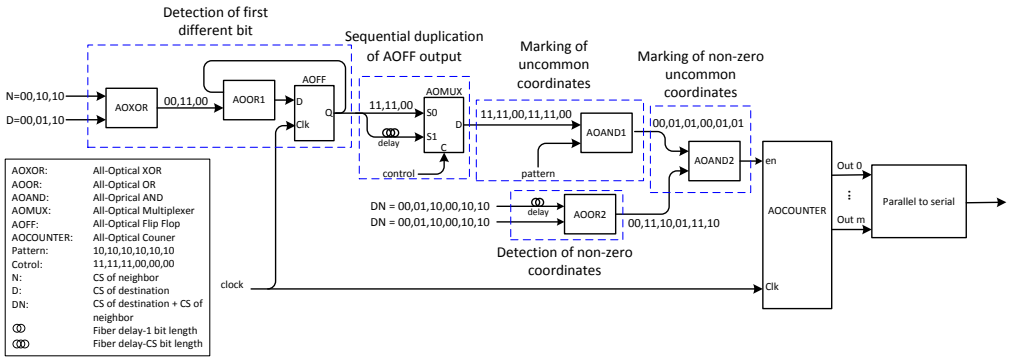

Fig. 4 Architecture of all-optical tree-distance calculator for one port/neighbor. In the provided CSs, the first coordinate is depicted at right and is the first to be fed into the circuitry.

As the coordinates should be counted in each CS and not the bits, we use a pattern which determines the last bit of each coordinate in a CS with a '1'. Feeding this pattern and the output of the multiplexer to an AND gate, we get a signal which is '1' only at the last bit of the coordinates in a CS and only if the first uncommon bit in the CSs is identified. If we apply this signal to the enable pin of a counter, all the coordinates of the two CSs from the location of the first uncommon coordinate are counted up to the end of the CSs. Now we need to make sure that only non-zero coordinates are counted. The design of this part is dependent on the number of bits in a coordinate.

In the lower part of the circuit, the two CSs are applied to the inputs of an OR gate consecutively (DN). As we see, one of the inputs is applied to the OR gate through a fiber delay. The duration of this delay element is equal to one bit. In Figure 4, we assumed that each coordinate of a CS consists of two bits. If there is at least one bit equal to '1' in each coordinate, the output of the AOOR2 would be '1' in the location of the last bit of that coordinate. For CSs with K-bit coordinates, the AOOR2 and the input fiber delay are replaced with K-1 single-bit fiber delays and one K-way OR gate (or (K-1) 2-input OR gates).

Feeding the output of AOAND1 and AOOR2 to an AND gate results in a signal which is '1' at the last bit of each non-zero coordinate in a CS and if only the first uncommon coordinate in the two CSs is detected. Applying this signal to the enable pin of a counter leads to counting the hop count of both CSs to their closest common ancestor which is equal to the tree-distance between them.

4.2.1 All-optical components for counting preparation circuit

In this section, the architecture of all-optical components used in the design of the tree-distance calculator are described. These components are all-optical AND, OR, XOR gates, multiplexer and flip-flop. As shown in Figure 4 these components are used to prepare a signal to enable the counting of coordinates in each CS. Therefore we refer to them as components required for 'counting preparation circuit'.

The design of the proposed all-optical components make use of the differentiated phase shift and gain offered by the SOAs when fed with different power levels. SOA is an optical amplifier based on a semiconductor gain medium. The differentiated phase shift of the SOAs can be utilized to form SOA-based Mach-Zehnder interferometer (SOA-MZI) circuit which acts as a cross and bar state switch. The differentiated gain of the SOAs can be utilized to selectively amplify the signals, which form the basis of the gates like OR gate.

A directional coupler has different phase shifts depending on whether to choose a cross or bar output. The cross output of a directional coupler differs in phase by 180 degrees whereas bar output does not differ in phase with the input. In a SOA-MZI configuration, the combination of couplers is chosen in such a way that even when the SOAs are fed with the identical inputs and offer the same phase and gain, the two inputs going through the different branches of the MZI combine finally with either a phase change of π or 0, leading to destructive and constructive interference. So the whole SOA-MZI circuit acts as a cross and bar switch.

In all the designs in this paper, the splits are done by means of directional couplers. However, in some figures the couplers are not depicted for the sake of simplicity. Figure 5A depicts a multiplexer. The selected area with the dashed line illustrates a SOA-MZI configuration. Depending on which of the inputs is required to be at the output, the inputs to the SOA-MZI configuration and the phase shift in the SOAs are tuned. Feeding the select signal directly to the lower arm of the SOA-MZI configuration causes the difference in the inputs of the SOAs which leads to having in2 in the output of the SOA-MZI if select is '1'. In other case,

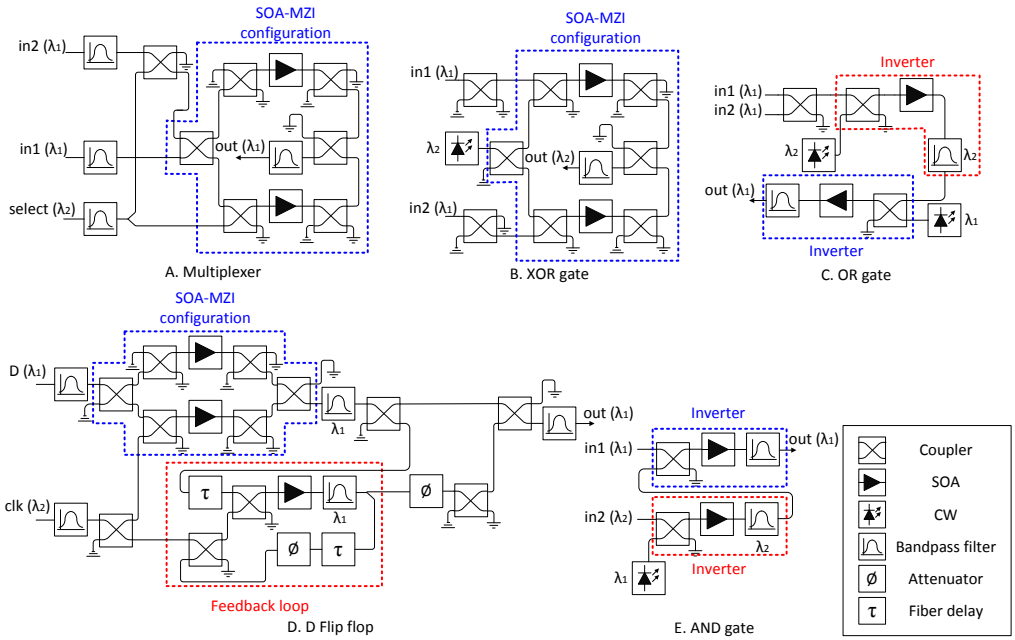

Fig. 5 Architecture of all-optical components. SOA-MZI functions as a cross-bar switch.

i.e. select=‘0’, with proper tuning of the phase shift in the SOAs the in1 appears in the output. An XOR gate is shown in Figure 5B. Similar to the multiplexer, the design of this gate is also based on a SOA-MZI configuration (selected with dashed line). The two inputs of the XOR gate are fed to the arms of the SOA-MZI block. In case both inputs are ‘1’, because of the phase shifts through the directional couplers, the output signal is destructive and thus ‘0’ and it would be ‘1’ only if one of the inputs is ‘1’. The next circuit is an OR gate which is based on two inverters (selected areas). We designed a D flip-flop which is level-triggered and uses a SOA-MZI configuration (selected on top in Figure 5D). The phase shifts through the directional couplers are in such a way that the inputs to the two SOAs are not equal and therefore the output cannot be destructive if clock (clk) signal is ’1’ which causes that the input (D) appears at the output. In the absence of a pulse in clk, the selected area in the bottom of the circuit functions as a feedback loop to maintain the output without any change. The final circuit represents an AND gate (Figure 5E). The functionality is emulated through 2 inverters which are selected with dashed lines (similar to OR gate). The output would be ‘1’ only if the NOT of in2 is ‘0’ and in1 is equal to ‘1’. Note that unlike other designs, the two inputs of the AND gate should be at different wavelengths.

In Figure 4, we demonstrated the need for a pattern (input of AOAND1) to determine the last bit of each coordinate in a CS and a control signal (for the AOMUX) to duplicate the output of the AOFF. In order to generate different patterns/control signals, we can exploit the output of the clock recovery block (Figure 3). Relying on this output, a set/reset pulse at the beginning of a packet is generated which can be used together with delay lines and directional couplers to generate different patterns/signals.

4.2.2 All-optical components for counting circuit

We implemented an all-optical counter based on the design in [22]. However, we changed this design to be able to use our implemented components such as AND gate. We extended this design with an additional AND gate in the input of the circuit to provide an enable pin in addition to a clock signal for the counter to control the counting. The extended counter counts up only if there is a clock pulse and the enable signal is ‘1’. The design of this counter is based on S-R flip-flops. Figure 6A illustrates the circuit for a 2-bit counter and the implemented S-R flip-flop is depicted in Figure 6B. Note that the inputs of the AND gates and the S and R inputs of the flip-flops should be at two different wavelengths. Therefore a wavelength converter is required at the input of the first stage to ensure the proper wavelength for other components. The design of such a converter is explained later.

As we see in Figure 6A the counter consists of two identical stages (except the wavelength converter in the first stage). Each stage includes an S-R flip-flop, an AND gate, directional couplers and fiber delay elements. There are 3 ports in each stage: i) input pulse, ii) output (out-i) and output carry signal (carry-i). Note that the carry1 from the first stage is the input of the sec-

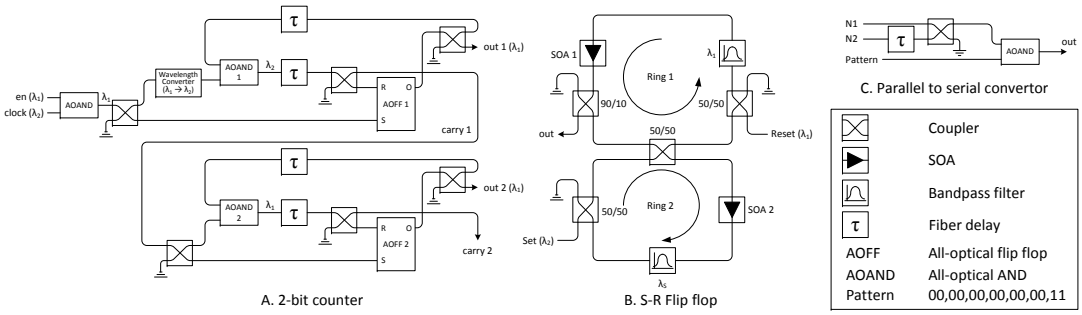

Fig. 6 Architecture of all-optical counter

ond stage. The outputs of the two flip-flops (out1 out2) are the outputs of the counter. At the beginning both flip-flops are at state '0'. When the input of first stage is '1', it cannot pass the AOAND1 because out1 is still '0'. Therefore, only the 'S' input of the AOFF1 is '1' which causes the flip-flop to change state to '1'. Upon arrival of the second pulse in the input of the counter, the 'S' input of the AOFF1 again receives the pulse and thus AOFF1 remains at state '1', however this time, the 'R' input of the AOFF1 will also receive the pulse but with a delay. So AOFF1 is set to '0'. The output of AOAND1 is fed to the 'S' input of AOFF2 which sets it to '1'. With the third pulse in the input, the AOFF1 is set to '1' but it does not pass the AOAND1 and AOFF2 remains at state '1' as well. Finally when the fourth pulse arrives, the output of AOAND1 is '1' and the 'R' input of AOFF1 receives it later than 'S' therefore, the AOFF1 changes its state to '0'. Similarly the AOFF2 is set to '0'. This circuit can simply be extended by replicating each stage and feeding the carry out signal to the next stage as the input pulse.

The S-R flip-flop is composed of 2 SOAs operating at two different wavelengths, namely λ_1 and λ_2. A pulse injected to 'Set' input will saturate SOA2, suppress Ring2, and then set the flip-flop to '1'. Similarly, a pulse injected to 'Reset' input saturates SOA1 and sets the flip-flop to '0'.

We need a parallel to serial circuit because the inputs of the all-optical comparator should be serial. Figure 6C depicts such a circuit for two bits. In this circuit, N1 and N2 are the outputs of the all-optical counter, out1 and out2 respectively. Feeding these two bits through a fiber delay element and a coupler to an AND gate together with a pattern as shown, we can have N1 and N2 consecutively.

4.3 All-optical comparator

We designed a 2-input all-optical comparator. Its functionality is different from typical comparators as it works similar to a selector. Therefore, the existing designs could not be used in the proposed greedy forwarder. Given two values (in serial) as input, the smaller value is selected by emitting a high intensity signal in the corresponding output. We explain the design in 3 different blocks.

The first block of the comparator, shown in Figure 7A, has two input (A and B) and two output ports (E and F). This block works in such a way that the outputs, E and F, are '0' if the inputs are the same (both '1' or '0') and only if one input is '1' the corresponding output will be '1' as well. Note that it is not possible that both outputs are '1' at the same time. As we see in this figure, the circuit works only if the two inputs are at two different wavelengths (λ_1 and λ_2). Assuming that the inputs (A and B) are given by two tree-distance calculators with the same circuit, working at the same wavelength, we need a wavelength converter at the input of the comparator to make sure that the two inputs have different wavelength (see Figure 7). The design of such a converter is depicted in Figure 7B. In this design, it is better to select the third wavelength λ_3 between the other two ($\lambda_1 < \lambda_3 < \lambda_2$).

The final block of the comparator is illustrated in Figure 7C. The goal of this design is to have a similar functionality as an S-R flip-flop but with the capability of having 3 states instead of 2 (i.e. states corresponding to two sets and one reset). As shown in the figure, this modified S-R flip-flop has 2 set inputs (E and F) and 1 reset input. out1 and out2 present the output of the comparator. It is composed of 3 SOAs, operating at different wavelength, λ_1, λ_2 and λ_R ($\lambda_1 < \lambda_R < \lambda_2$). The operation of this flip-flop is as follows: if E is set to '1', this state can build up in the outer ring (Ring2) due to the amplification in SOA2 and the allowed wavelength (λ_1) by the bandpass filter and thus, out2 will be set to '1' and the flip-flop is in state '1' via E. Note that the other bandpass filters are set in such a way ($\lambda_R - \lambda_2$ and $\lambda_1 - \lambda_R$) that this state cannot build up in the inner ring (Ring1). Similarly out1 is set to '1' if input F is '1' (this state builds up in the inner ring i.e., Ring1). Note that E and F cannot be '1' at the same time. This is guaranteed by the first block of the comparator ex-

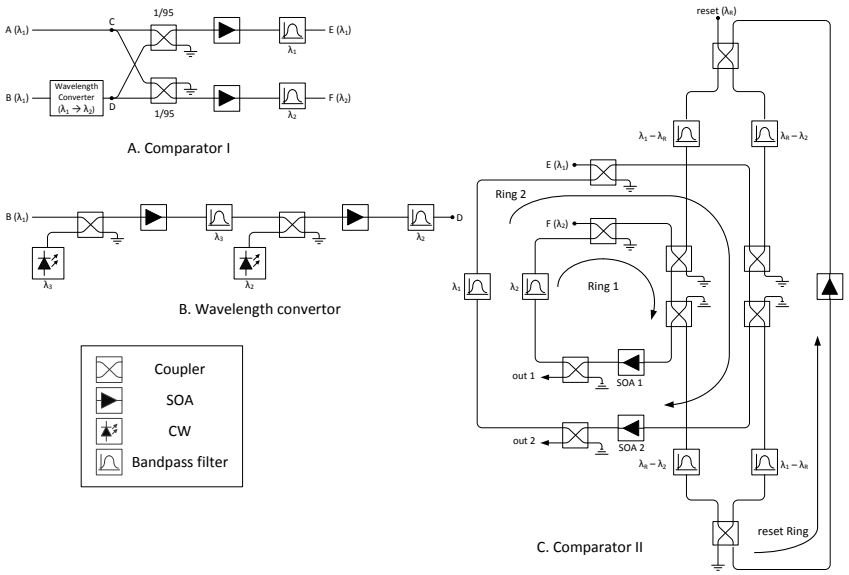

Fig. 7 Architecture of all-optical comparator

plained earlier. Thus out1 and out2 are not '1' at the same time either.

A pulse injected to the reset input of the flip-flop saturates SOA1 and SOA2, suppresses both Ring1 and Ring2, and sets the flip-flop to '0' (i.e. out1=out2='0').

5 Experimental results

The operation of the proposed circuits, namely tree-distance calculator and the comparator is verified through simulation using the commercially available simulation tool VPI. In this simulation, the example depicted in Figure 2B is considered. As explained earlier, in order to greedy forward the packets from S to D, in node S=(3,1,0), the tree-distance between every neighbor, namely (3,0,0) and (2,1,1), and D=(2,1,0) is calculated and the neighbor with minimum distance is selected i.e., (2,1,1). We have simulated this scenario (only in node S) and the expected values at different stages of the proposed design are depicted in Figure 8. As there are two neighbors/ports in node S, we need two blocks of tree-distance calculator, one for each port. The outputs of these two blocks are fed to the proposed all-optical comparator and the output of the comparator correspondent to the port with smaller distance rises to '1' when this value is detected. In the simulation results, we report the outputs of different stages of the circuit.

In this simulation, we consider wavelength sources with a center frequency of 193.12 THz (λ_1), 201.12 THz (λ_2) and 197.12 THz (λ_3). In tree-distance calculator (see Figure 4), we assume DN sources at λ_1 and all others at λ_2. In components where 3 wavelengths are needed the third source is assumed at λ_3 i.e., λ_3 in the wavelength converter and λ_R in the comparator are equal to 197.12 THz. The continuous wave (CW) light sources within different components are chosen appropriately. The clock signal has a data rate of 20 Gbps.

All the individual all-optical components including logical gates, multiplexer, flip-flops, counter and comparator are tested with a data rate of 10 Gbps. In order to emulate the delay elements of the design, a simple fiber-based delay element is used. The fiber length induces a delay equal to the duration of a data bit.

The first reported values correspond to an intermediate stage which is the input to the enable pin of the counter ('counting preparation circuit'). Figure 9 presents simulation input values for both tree-distance calculators (N1, D1, N2 and D2) and the output of the preparation circuits, namely en1 and en2. Feeding en1 and en2 to the enable pin of the corresponding counters, the tree-distances of 3 and 1 should be counted respectively. These values which are fed to the in1 and in2 of the comparator (see Figure 8) are reported in Figure 10. Note that the simulation time in Figure 10 starts at 1.2 ns. This is because all the bits of the two CSs should pass the counter (serially) to have the correct value of the tree-distance. We report the values of another intermediate stage which are E and F values in the comparator block (see Figure 7C). Finally the outputs of the comparator are reported in Figure 10 which is a high signal in out2 i.e., the output correspondent to the input with the smaller value.

In this simulation, the rise/fall time of the output signal is less than 10 picoseconds. This indicates that the proposed design is capable of functioning appropriately in higher data rates than the simulated one (i.e., 10 Gbps).

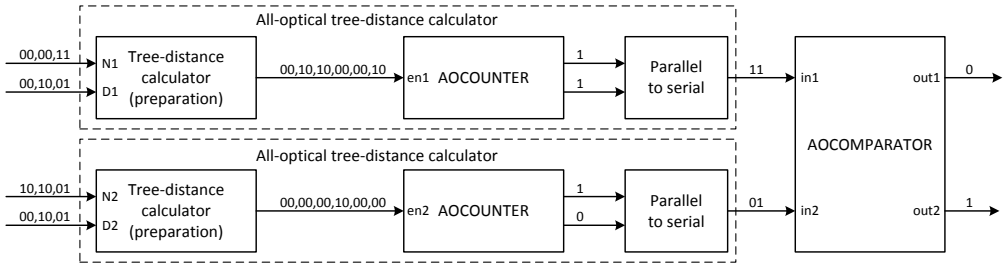

Fig. 8 Scenario used for experimental validation

5.1 Power consumption analysis

As explained earlier, all the all-optical components are based on SOAs and directional couplers. Since couplers are passive components, SOAs dominate the total power consumption of the proposed optical greedy router. Required number of SOAs in different components are reported in Table 1.

The proposed design scales with the degree of the nodes in the network. Assuming a node has degree 'm', 'm' tree-distance calculators and one m-way comparator are required to find the minimum value among the 'm' calculated distances. An alternative to an m-way comparator is to use 'm-1' 2-input comparators. Although cascading 2-input comparators may require additional components, we assume that these comparators can be connected directly. Based on this assumption, we analyze the required number of SOAs in the proposed design for a port in the router considering the Internet topology. Based on the number of the SOAs, we roughly estimate the power consumption of the proposed design.

The Internet is composed of almost 60K Autonomous Systems (AS) in the core. In the AS-level graph, the maximum degree of an AS is approximately 3K. These ASes are reached through several Internet exchange points (IXP) world-wide. Since there are roughly 70 IXPs in different areas, each IXP has almost 40-50 (3K/70) neighbors. According to these numbers, we consider 40-50 peers in the AS border routers. In the proposed scheme, each coordinate in the CS should be large enough to enable numbering of all the peers of an AS. Based on the above explanation, 6 bits are sufficient for each coordinate.

There is a segment in the tree-distance calculator design which is dependent on the number of bits in a coordinate. In Figure 4, the number of OR gates for detection of non-zero coordinates equals to the number of bits-1. Therefore, 5 OR gates are required in this segment. We consider a 5-bit counter for each tree-distance calculator as the maximum tree-distance between two nodes is less than 32. The latter refers to considering maximum depth of 16 for the spanning tree of the network. Each level of an all-optical counter consists of an AND and a S-R flip flop, thus in total 4 SOAs are required for each level. Looking at the design in Figure 6A, extra components such as a wavelength converter and an AND gate at the first level are needed as well. Considering all these extra components, a 5-bit counter requires 28 SOAs in total. In the greedy router, we consider 64 tree-distance calculators and thus 63 2-input comparators for each port are required. Based on the numbers reported in Table 1 and the assumptions made, the required number of SOAs for the proposed circuit for one port in the greedy router is 3897. Assuming the power consumption of 100mW for each SOA, the total power consumption of the proposed circuit is 389.7W. This value is independent of the data rate. Note that this power estimation was based on the discrete design of the components and applying the photonic integration technology will significantly improve the energy-efficiency of the router. Having this in mind, the estimated power consumption of the proposed circuit is comparable with the 40Gbps slot card with power consumption of 394W (based on Juniper T-series router [21]). Although this is not a precise power estimation, it gives an insight on the energy-efficiency of the proposed design.

6 Discussion and future work

In this paper, we designed and implemented several all-optical components required in the main blocks of a greedy forwarder, namely tree-distance calculator and comparator/selector. The simulation results confirmed the feasibility of the design for a sample input. However, in the design of the optical comparator, two inputs were considered. Knowing that the proposed design scales with the degree of the nodes, the comparator should be capable of selecting among more than two inputs. A solution is to use $(N-1)$ 2-input comparators to select among N inputs. However, in order to cascade the 2-input comparators, extra components such as multiplexers, AND gates, etc. are required between different stages. Cascading multiple blocks (specially blocks with complex designs such as the proposed comparator) leads to long delays which results in low data rates. A

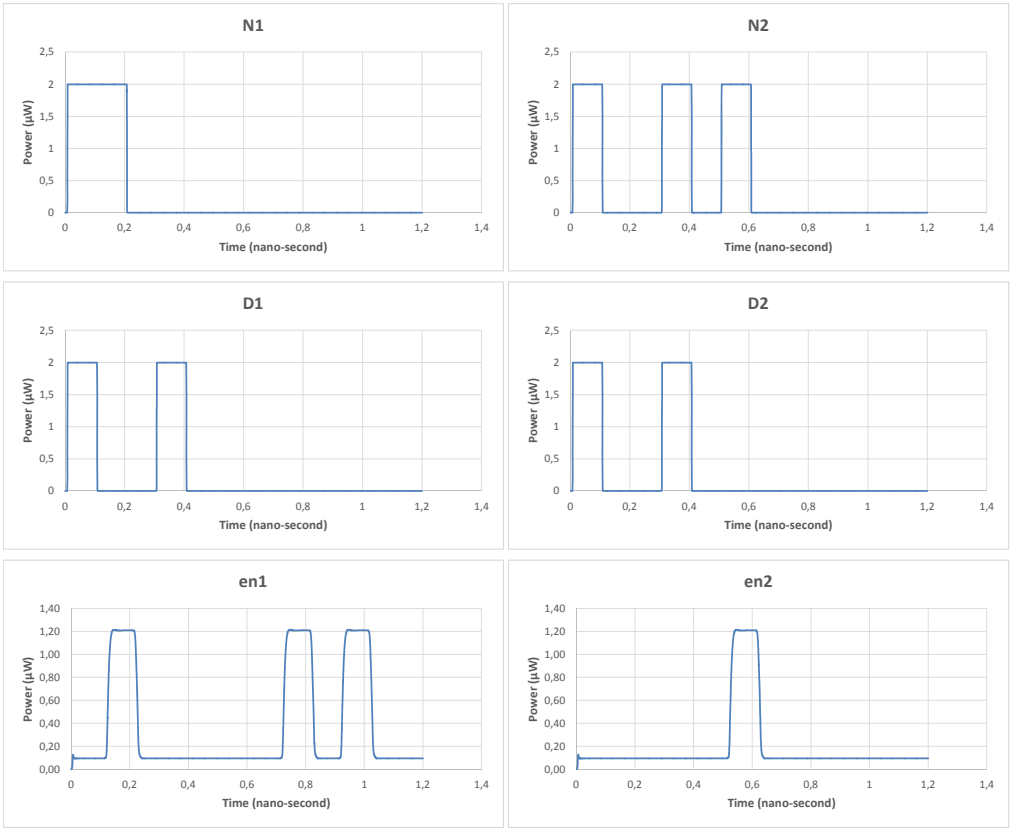

Fig. 9 Inputs of tree-distance calculator and input of enable pin in the counter

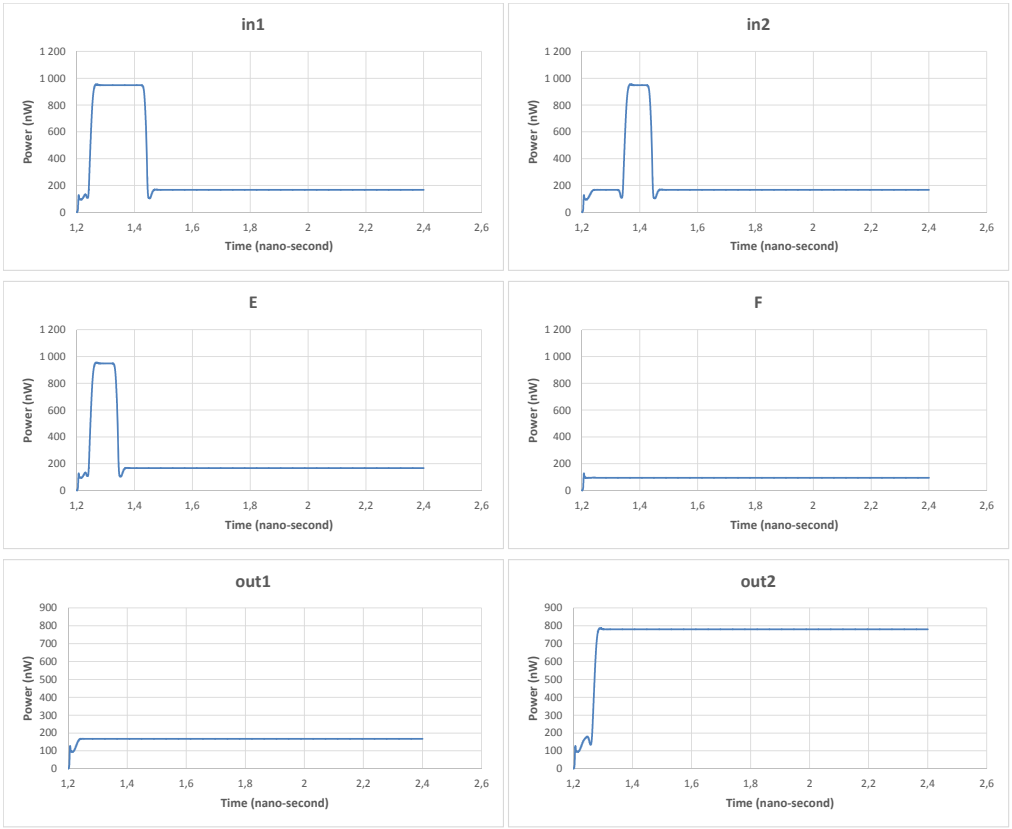

Fig. 10 Inputs and outputs of comparator

All-optical component	Number of SOAs
AND	2
OR	2
XOR	2
Multiplexer	2
D-Flip flop	3
Wavelength converter	2
5-bit counter	28
2-input comparator	5

Table 1 Required number of SOAs in all-optical components

future work includes designing an efficient circuit supporting selection among N values.

Regarding power consumption, the photonic integration technology is one way to improve the energy-efficiency of the proposed circuit. Additionally, sharing the proposed circuit among the ports of a greedy router is an alternate solution to reduce the total power consumption. These are interesting future research directions which require further investigation.

7 Conclusion

In this paper, for the first time, a novel scalable and truly all-optical tree-based greedy forwarder was designed. The successful operation of this design was demonstrated through simulation. The proposed greedy forwarder is more memory-efficient than conventional lookup-based IP routers and it scales with degree of nodes in the network. In the proposed design multiple all-optical components such as logic gates (XOR, AND, OR), multiplexer, flip-flops and more complex modules such as counter and comparator were constructed through interconnection of SOAs and directional couplers. SOA-MZI configuration as a basic block of several components was useful in order to achieve a scalable approach in terms of manufacturing. These components were implemented in simulation for a sample scenario with 6-bit inputs. The experimental results verified the feasibility of the design for data rate of 10 Gbps. The low rise/fall time of the simulation output indicates the capability of the proposed design in functioning in higher data rates.

References

1. Cvetkovski, A., Crovella, M.: Hyperbolic embedding and routing for dynamic graphs. In: INFOCOM 2009, IEEE, pp. 1647–1655 (2009)
2. Flury, R., Pemmaraju, S., Wattenhofer, R.: Greedy routing with bounded stretch. In: INFOCOM 2009, IEEE, pp. 1737–1745 (2009)
3. Fonseca, R., Ratnasamy, S., Zhao, J., Ee, C., Culler, D., Shenker, S., Stoica, I.: Beacon vector routing: Scalable point-to-point routing in wireless sensornets. In: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2, pp. 329–342. USENIX Association (2005)
4. Gumaste, A.A., Mehta, S., Rana, S., Das, T.: Obis: Optical bit switching-a switch architecture for optical omnipresent ethernet. In: Optical Fiber Communication Conference, p. OTuG3. Optical Society of America (2010)
5. Herrera, J., Raz, O., Tangdiongga, E., Liu, Y., Mulvad, H.C.H., Ramos, F., Marti, J., Maxwell, G., Poustie, A., Hill, M., et al.: 160-gb/s all-optical packet switching over a 110-km field installed optical fiber link. Lightwave Technology, Journal of **26**(1), 176–182 (2008)
6. Herzen, J., Westphal, C., Thiran, P.: Scalable routing easy as pie: A practical isometric embedding protocol. In: Network Protocols (ICNP), 2011 19th IEEE International Conference on, pp. 49–58. IEEE (2011)
7. Houthooft, R., Sahhaf, S., Tavernier, W., De Turck, F., Colle, D., Pickavet, M.: Fault-tolerant greedy forest routing for complex networks. In: Proceedings of 6th International Workshop on Reliable Networks Design and Modeling (RNDM) (2014)
8. Houthooft, R., Sahhaf, S., Tavernier, W., De Turck, F., Colle, D., Pickavet, M.: Robust geometric forest routing with tunable load balancing. In: Proceedings of INFOCOM 2015 (2015)
9. Huston, G.: BGP routing table reports (2015). Http://bgp.potaroo.net/
10. Karp, B., Kung, H.: GPSR: greedy perimeter stateless routing for wireless networks. In: Proceedings of the 6th annual international conference on Mobile computing and networking, pp. 243–254. ACM (2000)
11. Kleinberg, R.: Geographic routing using hyperbolic space. In: INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE, pp. 1902–1909 (2007)
12. Koch, B.R., Hu, Z., Bowers, J.E., Blumenthal, D.J.: Payload-envelope detection and label-detection integrated photonic circuit for asynchronous variable-length optical-packet switching with 40-gb/s rz payloads and 10-gb/s nrz labels. Journal of lightwave technology **24**(9), 3409 (2006)
13. Papadimitriou, C., Ratajczak, D.: On a conjecture related to geometric routing. Theoretical Computer Science **344**(1), 3–14 (2005)
14. Ramos, F., Kehayas, E., Martinez, J.M., Clavero, R., Marti, J., Stampoulidis, L., Tsiokos, D., Avramopoulos, H., Zhang, J., Holm-Nielsen, P.V., et al.: Ist-lasagne: Towards all-optical label swapping employing optical logic gates and optical flip-flops. Journal of Lightwave Technology **23**(10), 2993 (2005)
15. Sahhaf, S., Dixit, A., Tavernier, W., Colle, D., Pickavet, M., Demeester, P.: Scalable and energy-efficient optical tree-based greedy router. In: Transparent Optical Networks (ICTON), 2013 15th International Conference on, pp. 1–4. IEEE (2013)
16. Sahhaf, S., Dixit, A., Tavernier, W., Colle, D., Pickavet, M., Demeester, P.: All-optical tree-based greedy router. In: Optical Fiber Communications Conference and Exhibition (OFC), 2014, pp. 1–3. IEEE (2014)
17. Sahhaf, S., Tavernier, W., Colle, D., Pickavet, M., Demeester, P.: Experimental validation of resilient tree-based greedy geometric routing. Computer Networks (2015)
18. Sahhaf, S., Tavernier, W., Colle, D., Pickavet, M., Demeester, P.: Efficient geometric routing in large-scale

complex networks with low-cost node design. IEICE Transactions on Communications **99**(3), 666–674 (2016)
19. Seddighian, P., Baby, V., Habib, C., Chen, L., Rusch, L., LaRochelle, S.: All-optical swapping of spectral amplitude code labels for packet switching. In: Proc. Photonics in Switching, pp. 143–144 (2007)
20. Tang, M., Chen, H., Zhang, G., Yang, J.: Tree cover based geographic routing with guaranteed delivery. In: Communications (ICC), 2010 IEEE International Conference on, pp. 1–5. IEEE (2010)
21. Van Heddeghem, W., Idzikowski, F.: Equipment power consumption in optical multilayer networks-source data. Photonic Network Comm (2012)
22. Wang, J., Wang, J., Meloni, G., Berrettini, G., Potì, L., Bogoni, A.: All-optical counter based on optical flip-flop and optical and gate. ECOC 2009 (2009)
23. Westphal, C., Pei, G.: Scalable routing via greedy embedding. In: INFOCOM 2009, IEEE, pp. 2826–2830 (2009)
24. Yoo, S.: Optical packet and burst switching technologies for the future photonic internet. Lightwave Technology, Journal of **24**(12), 4468–4492 (2006)