J Opt (December 2023) 52(4):2188–2198
https://doi.org/10.1007/s12596-023-01142-9

RESEARCH ARTICLE

Efficient implementation of multi-level Dragonfly networks with Hamming graph for future optical networks

Heba A. Hassan[1,3] · Amr A. Al-Awamry[1] · M. B. Abdelhalim[2] · Fathi E. Abd El-Samie[3,4]

Received: 5 December 2022 / Accepted: 17 February 2023 / Published online: 30 May 2023

Abstract With the emergence of cloud computing and virtualized infrastructure in the datacenters, the use of high-radix routers is the most cost-effective alternative for interconnection networks. They are typically utilized as a part of datacenters for High-Performance Computing (HPC). Software Defined Networking (SDN) consolidates the benefits of datacenter virtualization, increasing resource flexibility and utilization and reducing infrastructure costs and overhead. Datacenter networks should be able to ensure high throughput and resiliency. For such reasons, Hamming graphs and Dragonfly networks are suitable for use with high-radix routers. Multilevel Dragonfly networks are used for lower-radix routers to increase the maximum achievable system size with the same router design. This paper introduces Hamming graphs and Dragonfly topologies based on SDN basis. It also presents a novel addressing scheme for Dragonfly topology with simulation experiments. The proposed model will be used for minimal, non-minimal, and adaptive routing in Dragonfly networks to extract a Python code using Mininet, which includes MiniEdit that is used to create and run network simulations. Evaluations show that with global trunking, systems are built with fewer groups than the maximum allowed. Therefore, there is no compelling reason for an additional cost. The proposed recommendations will be useful in the implementation of optical networks.

Keywords Software Defined Networking · Hamming graphs · Dragonfly network · Routing

✉ Heba A. Hassan
Ommolham2010@gmail.com

Amr A. Al-Awamry
amr.awamry@bhit.bu.edu.eg

Fathi E. Abd El-Samie
fathi_sayed@yahoo.com

1 Electrical Department, Benha Faculty of Engineering, Benha University, Banha, Egypt

2 College of Computing and Information Technology, Arab Academy for Science, Technology & Maritime Transport, Cairo, Egypt

3 Department of Electronics and Electrical Communications Engineering, Faculty of Electronic Engineering, Menoufia University, Menouf 32952, Egypt

4 Department of Information Technology, College of Computer and Information Sciences, Princess Nourah Bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia

Introduction

With the accelerated development of technology and bandwidth restrictions, the use of high-radix routers is vital to diminish the diameter, latency, and cost of interconnection networks. The motivation for using high-radix routers is their ability to enhance network performance in large-scale systems such as datacenters. As a result, an adaptable and cost-proficient topology is required to appropriately use high-radix routers. The importance of any topology comes from its ability to set performance bounds for the network by establishing the network diameter as well as the bisection bandwidth. The topology also largely determines the cost of the system. Existing topologies such as Folded-Clos or Fattree [1, 2] are not efficient, because they consume a large bandwidth to balance traffic that is already balanced. So, they pay too high penalty on load-balanced traffic (e.g. uniform) to provide good performance on the adversarial traffic pattern. On the other hand, a conventional Butterfly network is better than a Folded-Clos, because it gives a significantly lower cost (approximately half) than a Folded-Clos gives on balanced traffic. However, a conventional Butterfly network has a disadvantage, because it has no path diversity. So, its performance is severely limited on adversarial traffic patterns. The proposed Dragonfly topology is able to effectively increase the radix through the use of a virtual router or a collection of routers, where the scalability of the flattened Butterfly [3] is restricted by the radix of a single router.

Datacenter networks [4–7] and associated data flow scheduling are important for large-scale data-intensive computing. There are many network challenges in large data centers, but network virtualization offers the solution for meeting these challenges, because it reduces the management cost by allowing Information Technology (IT) administrators to manage the network through an interface without accessing the underlying network infrastructure. Another advantage of network virtualization is that it reduces the downtime of networks and applications, thus making troubleshooting easy. It also makes the network infrastructure more agile and scalable, because logical domains are connected through tunnels. So, IT administrators do not have to physically connect the domains. The SDN [8] achieves a rapid progress in cloud datacenters. The SDN technology also gives promising opportunities for high-throughput and high-volume applications, such as big data deployments in the financial and scientific sectors. There is a similarity between Network Functions Virtualization (NFV) [9] and SDN, as they speed up operation by breaking the bond between proprietary hardware and control/application software. So, the two architectures are optimized for the dynamic cloud environment at the carrier scale. Reducing both Operating Expenses (OpEx) and Capital Expenditures (CapEx) is the principal preferred standpoint of NFV and SDN, as they seek to leverage automation and virtualization to achieve better agility. Hence, the two concepts can enhance the benefits of datacenter virtualization by using data center topologies such as Dragonfly topology in SDN. The Hamming graphs are concerned with error-correcting codes and association schemes, to name two areas. They have also been considered as communication network topologies in distributed computing. So, special codes are used in Hamming graphs to protect connection paths [10, 11].

In this paper, the relationship between the Hamming graph (also known as flattened Butterfly) [3] and the Dragonfly topology [12] is studied, showing that Hamming graphs are extreme cases of Dragonfly networks, but with a large level of trunking that is required to retain an optimal global bandwidth with fewer groups than the maximum allowed. Thus, we can use the Dimension-Ordered deadlock-free Routing (DOR) mechanism that is used in Hamming graphs. It does not depend on Virtual Channels (VCs) for Dragonfly networks. Minimal and non-minimal routing mechanisms decouple the number and use of VCs from deadlock avoidance. So, the paper depends on these mechanisms for Dragonfly networks with trunking $t \geq 2$ and $t \geq 4$. Multiple works try to avoid or reduce the number of VCs in network routers, because the use of multiple VCs increases the area and power requirements of the router, makes some router allocation stages more complex, and entails a significant cost, leading to lower router frequencies and reduced throughput. However, multiple VCs provide deadlock freedom and help to reduce Head-of-Line Blocking (HoLB) [13].

The cost and performance of a scalable multiprocessor are the key elements in interconnection networks. Interconnection networks for low-radix routers, in which few ports are utilized, use low-radix topologies such as 2-D mesh, 3-D mesh, torus and clos (Fattree) topologies. Cray T3D, T3E, and XT3 are examples of machines that employ such networks. Note that low-radix networks provide optimal latency for a given cost because of the low pin bandwidth available in the past [14, 15]. As the pin bandwidth of router chips has increased due to the increase in the signaling rate and the increase in the number of signals, high-radix routers take this advantage of dividing the bandwidth into a larger number of narrow ports [16], where low-radix routers divide the bandwidth into a smaller number of wide ports. Intel's Knights Landing and future Xeon chips are examples of the designs that use on-chip routers, in which the router competes with on-chip cores, memories, and I/O for the chip resources, including the pin bandwidth that leads to lower-radix routers.

Scaling to large networks based on low-radix switches can be achieved by using multi-level Dragonfly networks that increase the maximum achievable system size with the same router design. This paper introduces a 2D Hamming graph $K_a \square K_b$ that is used for Dragonfly networks with trunking $t \geq 2$ and $t \geq 4$ for minimal and non-minimal routing, respectively, to build a balanced Dragonfly network that does not reach the maximum achievable size for a given router with a diameter, which can be in the order of millions of nodes. The 3D Hamming graph $K_a \square K_b \square K_c$ is also introduced in this paper. It is comparable to the 2D Hamming graph, but the scalability develops quickly with the number of levels, making configurations with more than 3 levels improbable. The paper also presents a novel addressing scheme for Dragonfly topology with simulation experiments. The proposed recommendations will be useful in the implementation of optical networks [17, 18].

Section 2 introduces the Hamming graphs and Dragonfly topologies. Section 3 gives an explanation of the proposed model for addressing of the Dragonfly topology. Section 4 describes the 2D Hamming graph $K_a \square K_b$ for Dragonfly networks with global trunking. Section 5 introduces two examples concerned with deadlock-free routing mechanisms for Dragonfly networks with trunking based on coloring of the underlying graphs, and then Mininet is used to extract the Python code. Finally, Section 6 gives the concluding remarks of the paper.

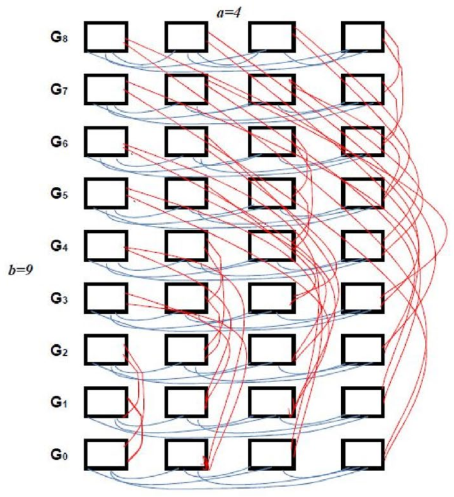
Hamming graphs and Dragonfly topologies

As mentioned in [19], the Hamming graph is characterized as the Cartesian product of complete graphs $K_{m1} \square \ldots \square K_{mn}$, where the Cartesian product is defined as having two vertices connected if and only if, for some component, they are connected in the corresponding factor and the other components are equal.

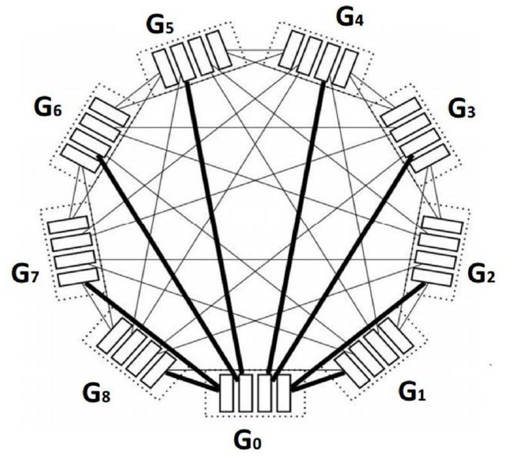
On the other hand, the Dragonfly network is a two-level hierarchical direct network as proposed in [12]. The second level has b groups (0, ……, b−1), each being composed of routers (0, ……., a−1) (first level). Different groups are connected by long, expensive, optical global links, whereas routers within a group are connected by short, cheap, electrical local links. The Dragonfly network is specified by the number of routers per group (a), the number of groups (b), and the global link arrangement (the specific router on each group to which each global link is connected).

The diameter k of the Dragonfly topology includes the diameters of the global topology k_g and local topology k_l, where $k \le k_g + (k_g + 1)k_l = k_g + k_g k_l + k_l$. The degree of the Dragonfly topology Δ also contains two levels, Δ_1 and Δ_2 , where Δ_1 is the number of local links connected to each router and Δ_2 represents the number of global links connected to each router. So, the topology has a degree $\Delta = \Delta_1 + \Delta_2$, and the total number of ports (radix) within routers is $R = \Delta_0 + \Delta_1 + \Delta_2$, where Δ_0 is the number of computed nodes in level 0, $b = a\Delta_2 + 1$ and $a = \Delta_1 + 1$ in any canonical Dragonfly network. The balancing occurs, when local and global links have similar loads under uniform traffic, and hence the condition $2\Delta_2 \approx \Delta_1$ is achieved. The balancing condition proposed in [12] relies on $a = 2\Delta_2$, and $\Delta_0 = \Delta_2$, where Δ_0 nodes can be connected to each router without saturating the network under uniform traffic. Canonical Dragonfly is the Dragonfly network using complete graphs with K_a and K_b in both local and global topologies [20]. In some cases, a canonical Dragonfly topology is a subgraph of the rectangular Hamming graph $K_a \square K_b$. Figure 1 shows an example with $a = 4$, $b = 9$, and $\Delta_2 = 2$.

As we mentioned, the Dragonfly topology is specified by a, b, and the global link arrangement. There are $b^{2O(a\Delta_2)}$ possible arrangements for the global links of a canonical Dragonfly. Consecutive, Palmtree, and circulant-based arrangements are a few specific cases of them, in which the topology is a subgraph of the Hamming graph. This paper depends on Palmtree arrangement [20] for the Hamming graph, which presents the same global connectivity pattern in each group of the system. This arrangement can be implemented by connecting vertex i in group j to vertices $a-1-i$ in groups $j - i\Delta_2 - 1$, $j - i\Delta_2 - 2$,…., and $j - i\Delta_2 - \Delta_2$. Figure 1b shows the Palmtree arrangement of the previous example. Most deadlock-free routing mechanisms proposed for Dragonfly networks require an ordered use of virtual channels for hops allowed through a given type of network link. Canonical Dragonfly depends on a minimal routing mechanism in which the path consists of one local link l to the router with the required global link, then the global link g itself, and finally a local link l to the destination. So, the paths with only two global links gg are avoided.



(a) Hamming graph with a=4, b=9, and $\Delta_2 = 2$.



(b) Palmtree arrangement for thesame example.

Fig. 1 Two layouts of the same Dragonfly topology, which is a subgraph of $K_4 \square K_9$ with $\Delta_2 = 2$, with nodes organized in rows and columns (left, each row corresponds to a different group) or groups (right). Global links leaving group 0 are in bold. (**a**) Hamming graph with $a = 4$, $b = 9$, and $\Delta_2 = 2$. (**b**) Palmtree arrangement for the same example

Since minimal routing [12] locates the global link between the source and destination groups and has paths of type *lgl*, local ports require two different VCs, whereas global ports do not require VCs. This mechanism is costly and complex. So, in Section 4, we will use Dragonfly networks with trunking, because hey do not require VCs.

Proposed model for addressing of Dragonfly topology

There is no addressing for the Dragonfly topology, and hence we assume that we have some groups of switches instead of routers and we put one of them as a core switch, but the other switches act as pod switches. Figure 2 shows an expanded block diagram of the previous example with $a=4$, $b=9$, and $\Delta_2=2$.

We have 9 groups. So, we put one of the groups to act as core switches and the remaining groups to act as pod switches. We consider 8 pod switches. The IP address allocation within the subnetwork is 10.0.0.0/8. Pod switches have address 10.pod.switch.1, where pod and switch in [0, *k*-1] are based on position. Core switches have address 10.*k*.*j*.*i*, where *i* and *j* denote core position in $(k/2)^2$ core switches. Similarly, hosts have address 10.pod.switch.ID, where ID is the host ID in the switch subnet [2, $(k/2)+1$]. Figure 3 shows the previous example with addresses for pod switches and core switches.

The work in this paper depends on a virtual machine known as Mininet [21] to set up an SDN without hardware. The Mininet network simulator incorporates MiniEdit, a simple GUI editor for Mininet. MiniEdit is an experimental tool made to exhibit how Mininet can be extended. MiniEdit is used to make and run network simulations. The paper also demonstrates how to use MiniEdit to build a network, configure network elements, save the topology, and run the simulation experiments. Figure 4 shows that we were able to create a custom network topology graphically using MiniEdit, and then we can extract the Python code.

2D Hamming graph $K_a \square K_b$ for Dragonfly networks with global trunking

As mentioned earlier, canonical Dragonfly topologies can be seen as subgraphs of Hamming graphs with a proper global

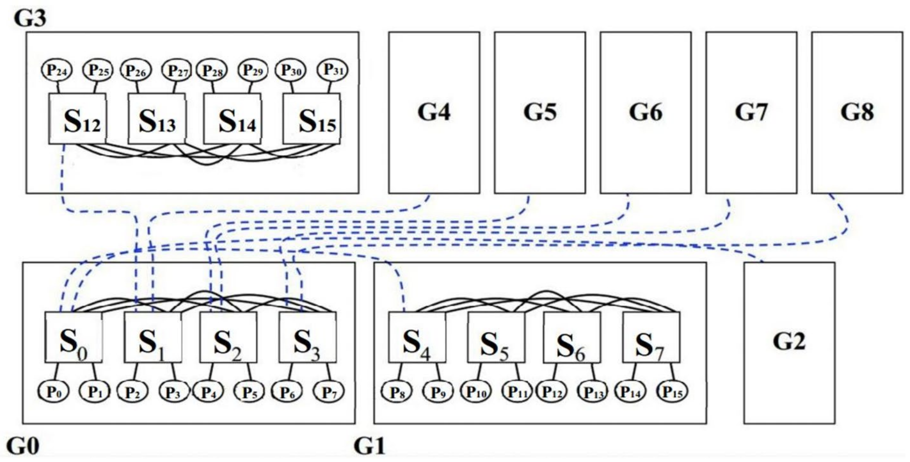


Fig. 2 An expanded block diagram of the previous example

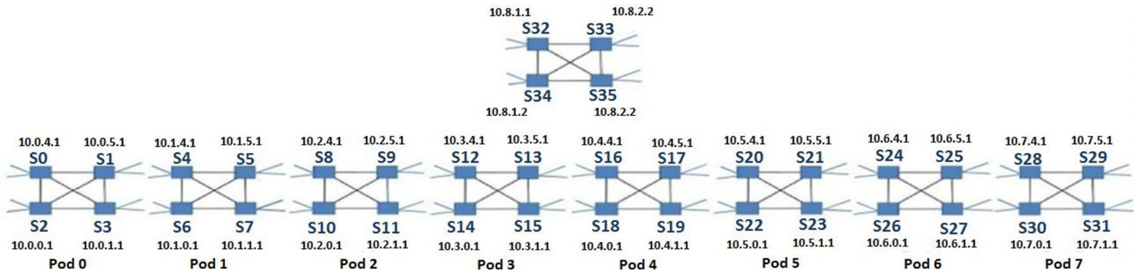


Fig. 3 Addressing for Dragonfly topology

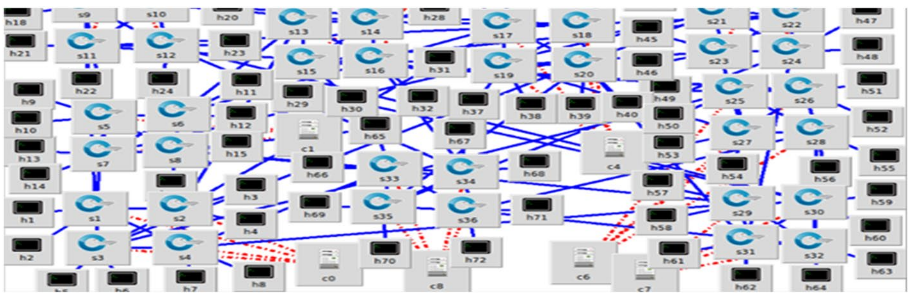


Fig. 4 Custom network in MiniEdit (Dragonfly topology)

Table 1 Several dimensioning examples for groups of a network with $a=4$ routers per group and different levels of trunking

t	Number of groups b using $\alpha=1$ according to $b = 1 + \alpha\frac{a(a-1)}{t}$	Number of groups b for a balanced network according to $b = 1 + \frac{1}{1+\left(\frac{t}{a}-1\right)^2}\frac{a(a-1)}{t}$	Number of groups b for α = 1/2 according to $b = 1 + \alpha\frac{a(a-1)}{t}$
1	13	8.7	7
2	7	5.8	4
3	5	4.8	3
4	4	4	2.5

link arrangement, which means that Hamming graph is an extreme case of Dragonfly network, but with a large level of trunking to retain optimal global bandwidth in systems with fewer groups than the maximum allowed [22]. The trunking level in a topology is the number of parallel links that are employed to increase the aggregated bandwidth, increasing also the number of router ports used. This concept includes two parts, local trunking and global trunking levels. Local trunking in a Dragonfly topology points to parallel links between pairs of routers within a group, whereas global trunking level t refers to the number of global links between every pair of groups. Dragonfly networks with global trunking achieve the relation:

$$a\Delta_2 = t(b-1) \tag{1}$$

It is clear from the relation that Dragonfly networks with trunking depend on the number of routers per group a, the number of groups b, the global links per router Δ_2, the global link arrangement, and the global trunking $t > 1$ ($t=1$ for a canonical Dragonfly without trunking).

As proposed in [23], a balanced trunked Dragonfly network achieves the following relation under uniform traffic and minimal routing assumptions:

$$b = 1 + \alpha\frac{a(a-1)}{t} = 1 + \frac{1}{1+\left(\frac{t}{a}-1\right)^2}\frac{a(a-1)}{t} \tag{2}$$

where the parameter α controls the relation between the number of links of each type (global, local) for a balanced network and the number of edges of each type in a balanced network as follows:

$$\alpha = \frac{\text{avg}_2}{\text{avg}_1} = \frac{|E_2|}{|E_1|} = \frac{\frac{tb(b-1)}{2}}{b\frac{a(a-1)}{2}} = \frac{t(b\ -\ 1)}{a(a\ -\ 1)} \tag{3}$$

In addition, α can be approximated with the average distance as:

$$\alpha \approx \frac{1}{1+\left(\frac{t}{a}-1\right)^2} \tag{4}$$

For the maximum trunking case of Hamming graphs with $t=a$, and $\alpha=1$, the balancing condition is $b=a$ or equivalently $\Delta_2=a-1$, whereas in the canonical Dragonfly with $t=1$ (no trunking) and $\alpha=1/2$, the approximate Dragonfly balancing condition is $b=1+a\ (a-1)/2$.

Trunking modifies the balancing conditions of the network, and this is observed in Table 1. Networks with fewer groups require more trunking to be balanced. Table 1 shows examples of dimensioning of the number of groups b that keeps $\alpha \in \left[\frac{1}{2}, 1\right]$.

Figure 5 shows that for $t=1$, the balancing condition approaches $\alpha = {}^{1}\!/_{2}$. On the other hand, for $t=a=4$, the balancing condition approaches $\alpha = 1$. The figure also demonstrates that the fewer the groups of a Dragonfly network that are accessible, the higher the trunking level required for the topology to be balanced is.

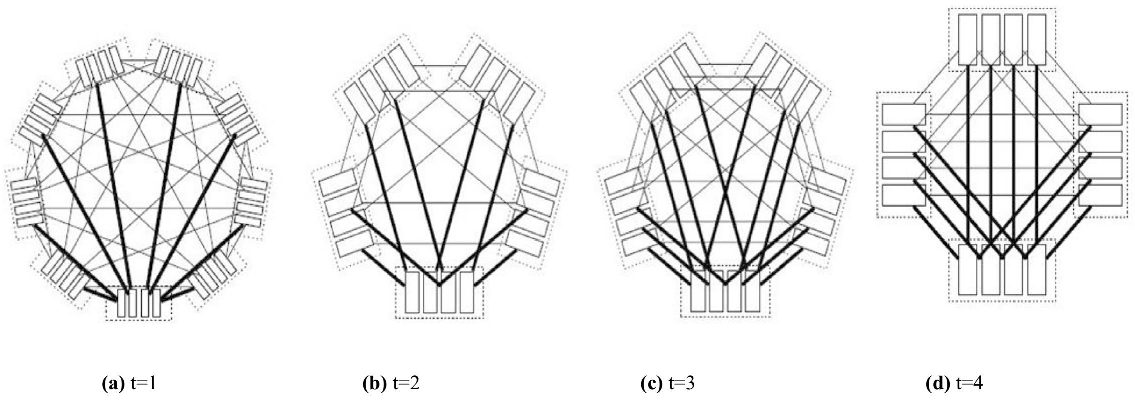


Fig. 5 Dragonfly network with $a=4$ routers per group and b groups according to Table 1. (**a**) $t=1$, (**b**) $t=2$, (**c**) $t=3$, (**d**) $t=4$

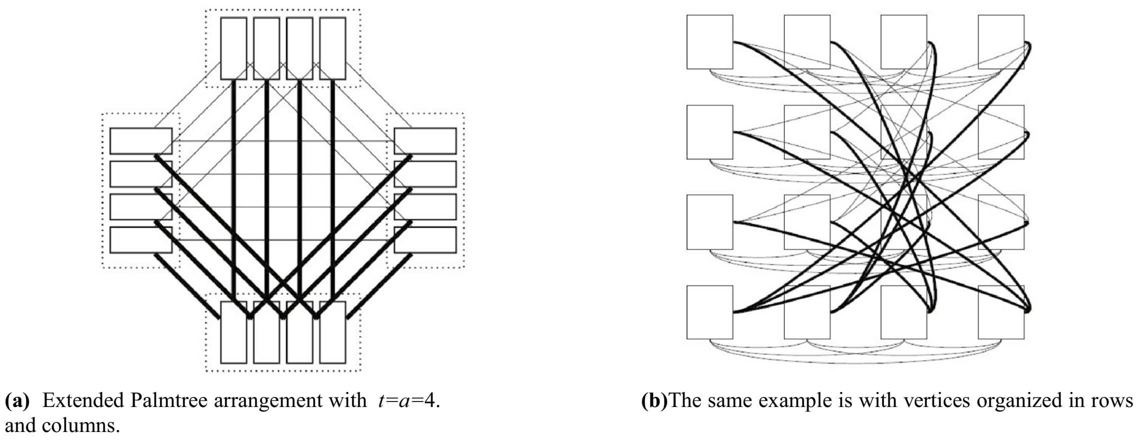

(a) Extended Palmtree arrangement with t=a=4.
and columns.
(b)The same example is with vertices organized in rows

Fig. 6 Dragonfly network with extended Palmtree arrangement with $a=b=4$, $\Delta_2=3$. (**a**) Extended Palmtree arrangement with $t=a=4$. (**b**) The same example with vertices organized in rows and columns

The main characteristic of trunking is that it increases the number of possible arrangements of a Dragonfly network. The previous configurations of global link arrangements for Dragonfly networks without trunking can be applied here, but these configurations are denoted as extended trunking. Figure 6(a) shows the Dragonfly network with extended Palmtree arrangement with $a=b=4$, and $\Delta_2=3$. This case is called maximum trunking as $t=a=4$. The graph obtained for this case is very similar to the Hamming graph, but it is not isomorphic to it. Figure 6(b) shows a representation of such a graph with nodes organized in rows and columns.

Deadlock-free adaptive routing in Dragonfly networks with trunking

The advantage of using Hamming graphs with Dragonfly networks is that they allow for deadlock avoidance mechanisms in the light of route restrictions (DOR). So, the number of VCs allowed for these mechanisms is decreased. A DOR mechanism depends on coloring of all the links in the network with one of two colors, according to their dimensions, and following paths that comply with a certain color order. To achieve a careful selection of the global link connectivity, the paper depends on an extended Palmtree or a subgraph of the Hamming graph for the global link configuration, and hence coloring of routers is possible.

There are three alternative routing mechanisms for Dragonfly networks with global trunking in view of a variety of the route restriction mechanism utilized in Hamming graphs. The first routing mechanism is minimal routing, and it requires $t\geq 2$ based on 2 router colors, and it can be implemented without VCs. The second routing mechanism is a variant of Valiant routing (non-minimal), which sends traffic to an intermediate network router. It

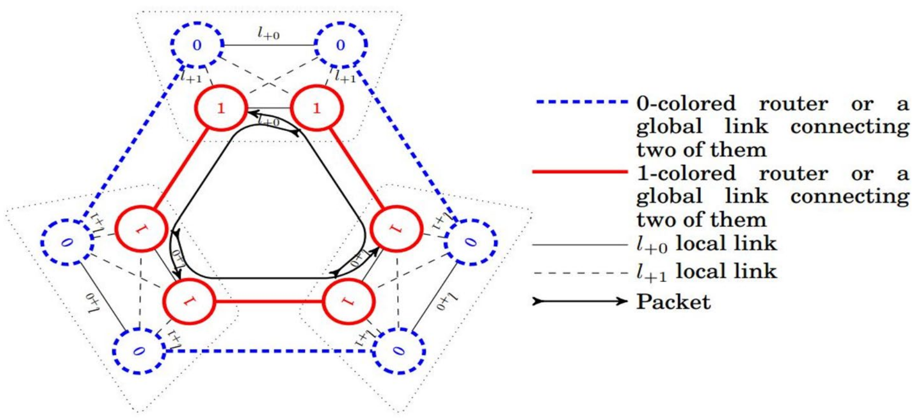

0
1
l_{+0}
l_{+1}
0-colored router or a global link connecting two of them
1-colored router or a global link connecting two of them
l_{+0} local link
l_{+1} local link
Packet

Fig. 7 Coloring of routers with 0 or 1 and the local links with + 0 or + 1

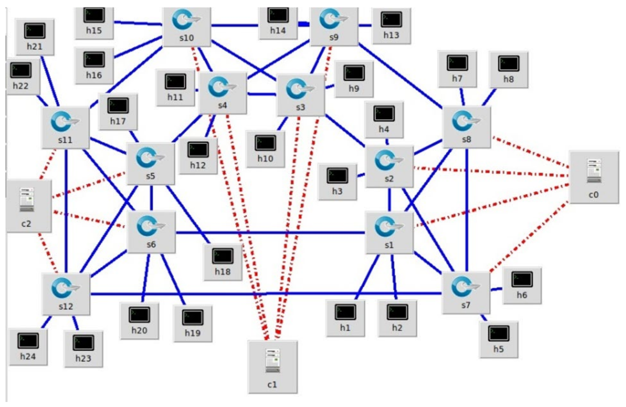


Fig. 8 Custom network in MiniEdit (minimal routing for $t \geq 2$)

is used for adverse traffic patterns requiring $t \geq 4$ and can be implemented without VCs. Finally, the third routing mechanism is an adaptive mechanism. It also requires $t \geq 4$, and selects between the minimal or Valiant paths depending on network conditions.

Minimal deadlock-free routing for $t \geq 2$

As mentioned, DOR mechanism based on the “2-color minimal” like *lgl* route is required for Dragonfly networks with $t \geq 2$ global links between pairs of groups. The cyclic dependency presented would be avoided using the color-ordering rules that decide which of the t global links to use each time [19]. Figure 7 introduces a simple three-group example with 0 or 1 for coloring of routers and + 0 or + 1 for local links.

It is clear from the figure that there are 4 routers per group, and two of them have the first color 0 and the remaining have the second color 1. For global links, vertices with the same color should be connected by the proper arrangement of global links, but this leads to a restriction in the global link arrangement. To satisfy this restriction, the paper idea depends on an extended Palmtree for $a \geq 4$ and any subgraph of the Hamming graph for $a \geq 2$ to divide vertices into several classes. Local links l_{+0} and l_{+1} are used to connect vertices with the same or different colors, respectively, according to the difference of their endpoints.

As shown in the figure, the respective colors of the source and destination routers will vary the routing mechanism. For routers with the source and destination of different colors, the l_{+0} link is always used in the source group and l_{+1} in the destination group and the path contains one l_{+0} and one l_{+1}, whereas the global link will have the same color as the source router, and hence cyclic dependencies from l_{+1} l_{+0} on local links are prevented. For routes in which endpoints have the same color, l_{+0} local links are always used, when the destination group index is larger than the source index and the l_{+1} local links. Otherwise, the path must contain two l_{+0} or two l_{+1} local links, whereas the global link will have the same color as the source router.

Figure 7 shows that there are 3 paths between routers of different groups in which l_{+0} local links are used, but at least one of the paths will decrease the group index. So, l_{+1} links will be used. Finally, all links are used similarly under uniform traffic. Using MiniEdit, the custom network will be as shown in Fig. 8. The proposed model for addressing Dragonfly topology will be used, and hence the Python code can be extracted.

Minimal and non-minimal deadlock-free routing for $t \geq 4$

The DOR mechanism based on "4-color non-minimal" is required for Dragonfly networks with $t \geq 4$ global links between pairs of groups. Non-minimal valiant routes like *lgllgl* are used for adverse traffic patterns, which do not need VCs for deadlock-freedom [23]. The 4-color non-minimal can be converted to 4-color minimal, by traversing only the first or the second half of the allowed path with a single global hop despite using one more local hop in some cases, but the 4-color minimal is less restrictive than the previous mechanism for $t = 2$. Figure 9 shows the 4-color non-minimal, with $t \geq 4$. This mechanism is also based on coloring of the routers to order local and global links, since local links must be ordered.

Figure 9 shows that there are 4 colors for routers, 2 colors for global links, and 8 colors for local links. $\{0A; 0B; 1A; 1B\}$ are four labels for routers used in this mechanism. For global links, vertices with the same color could be connected to the extended Palmtree with $a \geq 8$ and to subgraphs of the Hamming graph with $a \geq 4$. gA, and gB are two labels for global links according to their endpoints. For local links, there are eight labels $\{l_{+0AA}, l_{+0BA}, l_{+1AA}, l_{+0AB}, l_{+1AB}, l_{+1BB}, l_{+1BA}, l_{+0BB}\}$. They are determined according to the source and destination. An ordering of links is required to achieve a deadlock-free routing. Figure 10 introduces the complete ordering for local links, which are allowed for the paths. Allowed paths flow from left to right, and parallel routes represent different alternatives, one of which is chosen depending on the labels of the source and destination routers.

Non-minimal routing uses the complete ordering $l_{+0\mathrm{AA}}$; $l_{+0\mathrm{BA}}$ ‹ gA ‹ l_{+1AA} ‹ ${}_1 l_{+1AB}$ ‹ l_{+0AB} ‹ l_{+1BB} ‹ g B ‹ $l_{+1\mathrm{BA}}$; l_{+0BB}. Global links will always be gA ‹ gB, which means that

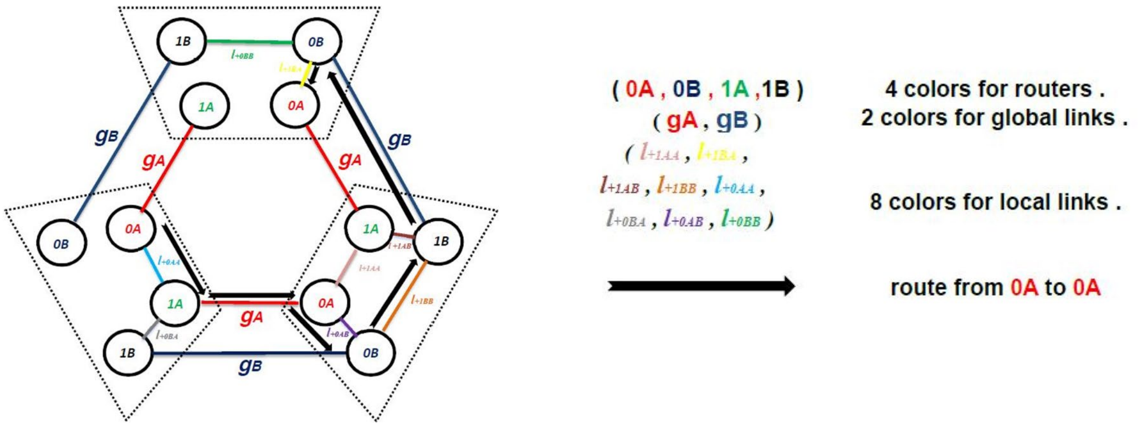


Fig. 9 Coloring of routers with $\{0A; 0B; 1A; 1B\}$, and the local links with $\{ l_{+0AA}, l_{+0BA}, l_{+1AA}, l_{+0AB}, l_{+1AB}, l_{+1BB}, l_{+1BA}, l_{+0BB}\}$

Fig. 10 A precedence of links using $t=4$, which allows for routes *lgl* and *lgllgl*, and every node represents a link in the path

$$\begin{tikzcd}
l_{+0AA} \arrow[dr] & & l_{+1AA} \arrow[r] & l_{+1AB} \arrow[dr] & & l_{+1BA} \\
 & g_A \arrow[ur] \arrow[dr] & & & g_B \arrow[ur] \arrow[dr] & \\
l_{+0BA} \arrow[ur] & & l_{+0AB} \arrow[r] & l_{+1BB} \arrow[ur] & & l_{+0BB}
\end{tikzcd}$$

Fig. 11 Custom network in MiniEdit (minimal and non-minimal routing for $t \geq 4$)

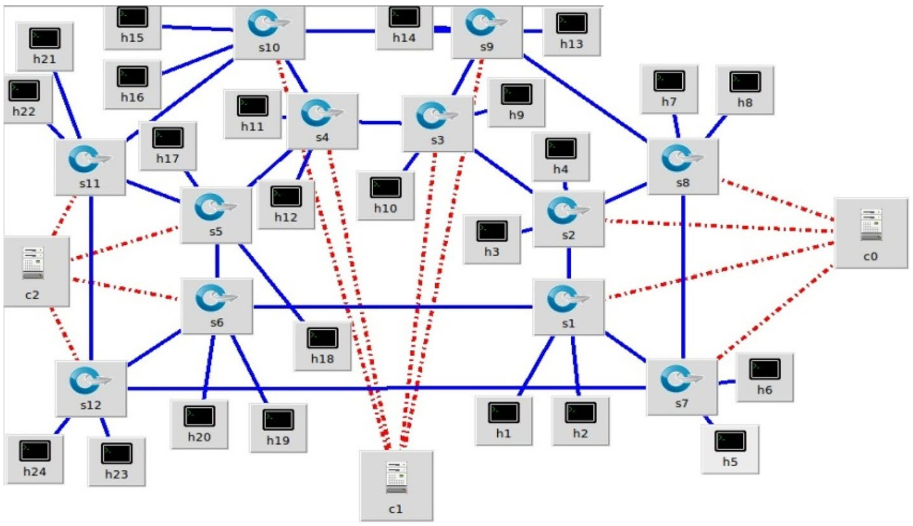


Table 2 Characteristics of 2D balanced Dragonfly networks with a, and b routers per dimension, Δ_0 computing nodes per router, router radix (R ports), and the approximate number of computing nodes

2-level balanced Dragonfly networks with a, and b routers per dimension and Δ_0 computing nodes per router	Canonical Dragonfly $t = 1$	Hamming graph $K_a \square K_b$ $t = a$
Balancing conditions	$b = 1 + a(a-1)/2$	$b = a$
Link use relations	$\alpha \approx 1/2$	$\alpha = 1$
Routers $= ab$	$a^3/2$	a^2
Radix $= R = \Delta + \Delta_0$	$R = (a-1)(1+2\alpha) = 2(a-1)$	$R = 3(a-1)$
Computing nodes $= ab\ \Delta_0$	$(R^4 + 4R^3 + 12R^2)/2^6$	$(R+2)^3/3^3$
General route	lgl	lg

the links of class gA will appear earlier in the route than the global link of class gB. It is obvious from the ordering that local links are selected depending on the label A or B of the source router at the beginning of the path of ordering and depending on the label A or B of the destination router at the end of the path of ordering, while in the middle depending on the change $+0/+1$ of the whole path. Based on the previous ordering, the route from the $0A$ source router to the $0A$ destination router in different groups is l_{+0AA}; gA; l_{+0AB}; l_{+1BB}; gB; l_{+1BA}. It is clear that this mechanism uses any of the $0B$ routers in the network as the intermediate router of the Valiant path. However, if we divide the previous path, we note that packets going from a $0A$ router to a $1A$ router have a route l_{+0AA}; gA; l_{+1AA} in which packets go, minimally. Similarly, packets go minimally from a $0A$ to a $0A$ router, and the path is l_{+1AB}; gB; l_{+1BA}. Finally, adaptive routing can be used, since minimal and non-minimal routes are allowed with the same ordering of links. It is based on selecting one of them at the source. So, UGAL method [24] is used, because it depends on some decision mechanisms and congestion information from neighbors. Similarly, after using MiniEdit, the custom network will be as shown in Fig. 11.

Table 2 shows the maximum number of computing nodes in a network ($ab\Delta_0$) for a given router radix R in which

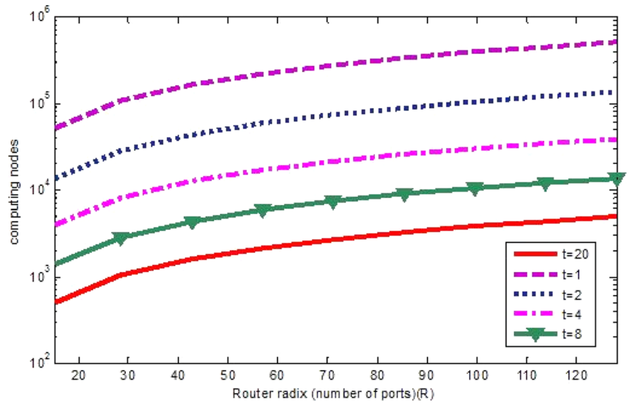


Fig. 12 Scalability for 2-level Dragonfly networks with trunking level t

$R = \sum_{i=0}^{n} \Delta_i = \Delta_0 + \Delta$ and for 2-level networks with $R = (a-1)(1+2\alpha)$ and α obtained from Eq. (3). This paper deals with Moore graphs in which the degree Δ and diameter k give the maximum number of vertices, namely $k = 2$ with $\Delta = 2,3,7$ and $k = 3$ with $\Delta = 2$ [24].

Figure 12 shows the scalability for 2-level Dragonfly networks with trunking level t by evaluating the system size for different router radii and trunking levels. The figure includes the 2D Hamming graph ($t = a$, $k = 2$), canonical 2-level Dragonfly topology ($t = 1$, $k = 3$), and multiple alternatives with variable trunking levels and smaller size than the canonical Dragonfly. It is shown from the figure that with global trunking, systems do not reach the maximum size for a given router and a diameter which can be in the order of millions of nodes. Thus, trunking is required to build systems with size smaller than the maximum achievable size.

3-Level Dragonfly Networks

By increasing the number of hierarchy levels, larger Dragonfly networks can be built with high scalability. This section gives the properties of 3-level Dragonfly networks, which are comparable to 2-level Dragonfly networks, but the scalability develops quickly with the number of levels making configurations with more than 3 levels improbable.

The 3-level Dragonfly networks [19] include a 1-level group that consists of routers, a 2-level group that consists of b 1-level groups, and c 2-level groups that exist in the whole network. The 3-level Dragonfly networks include local (l or 1), medium (m or 2), and global (g or 3) links. Hence, the degree will be extended to $\Delta = \Delta_1 + \Delta_2 + \Delta_3$, and there are two trunking levels (t_2, t_3).

For this case, t_2 is the number of links between every pair of 1-level groups, and t_3 is the number of links between every pair of 2-level groups. Similar to the 2-level trunked Dragonfly networks, the balancing conditions for 3-level Dragonfly networks can be obtained from the next relations:

Table 3 Characteristics of 3-level balanced Dragonfly networks with a, b, and c routers per dimension, Δ_0 computing nodes per router, router radix (R ports), and the approximate number of computing nodes

3-levels balanced Dragonfly networks with a, b and c routers per dimension, Δ_0 computing nodes per router	**3**-level canonical Dragonfly $t_2=1$, $t_3=1$	Hamming graph $K_a \square K_b \square K_c$ $t_2=a$, $t_3=ab$	Cascade-like $t_2=a$, $t_3=1$	Fourth case $t_2=1$, $t_3=b$
Balancing conditions	$b=1+a(a-1)/2$, $c=1+b(b-1)/2$	$a=b=c$	$a=b$ $c-1=a^2(a\text{-}1)/2$	$c\text{-}1=b-1\approx a(a-1)/3$
Link use relations	$\alpha \approx \beta \approx 1/2$	$\alpha = \beta = 1$	$\alpha = 1$ $\beta \approx 1/2$	$\alpha \approx 1/3$ $\beta = 1$
Routers $= abc$	$a^7/16$	a^3	$\approx a^5/2$	$a^5/3^2$
Radix $= R = \Delta + \Delta_0$	$R = (a-1)(1+\alpha+2\alpha\beta) = 2(a-1)$	$R = 4(a-1)$	$R = 3(a-1)$	$R = 2(a-1)$
Computing nodes $= abc\ \Delta_0$	$R^4(R+2)^4/2^{14}$	$(R+3)^4/2^8$	$(R^6+12R^5+54R^3)/(2^2\times 3^6)$	$R^6/(2^6\times 3^3)$
General route	*lmlglml*	*lmg*	*lmglm*	*lmlgl*

$$a\Delta_2 = t_2(b-1) \approx a(a-1)\alpha \quad (5a)$$

$$ab\Delta_3 = t_3(c-1) \approx t_2 b(b-1)\beta \quad (5b)$$

Considering that α and β are the relations between the average distance on each type of link, where $\alpha = \frac{avg_2}{avg_1}$ and $\beta = \frac{avg_3}{avg_2}$, the degrees are obtained from the next relations:

$$\Delta_2 \approx (a-1)\alpha \approx \Delta_1\alpha \quad (6a)$$

$$\Delta_3 \approx (a-1)\alpha\beta \approx \Delta_2\beta \quad (6b)$$

Table 3 shows the maximum number of computing nodes in a network ($abc\ \Delta_0$) for a given router radix. As we mentioned earlier, $R = \sum_{i=0}^{n} \Delta_i = \Delta_0 + \Delta$, and for 3-level balanced networks, $R = (a-1)(1+\alpha+2\alpha\beta)$. The table includes four important cases, which are exceptionally significant. The first case is the canonical 3-level Dragonfly network with no trunking. The second case is 3D Hamming graph $K_a \square K_b \square K_c$, which is balanced for $a=b=c$. The third case is a 3-level

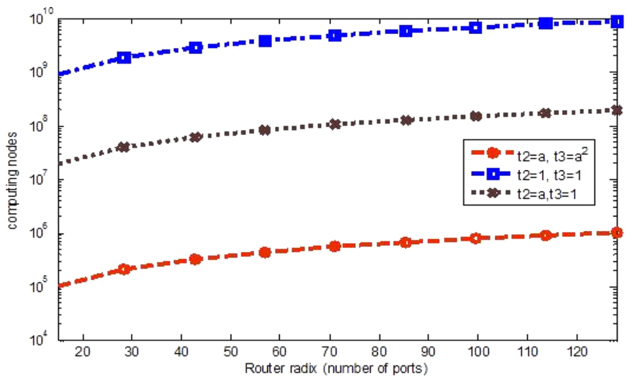


Fig. 13 Scalability for 3-level Dragonfly networks with trunking levels t_2 and t_3

cascade-like Dragonfly network, which has $t_2 = a$, and it is equal to a 2-level Dragonfly network in which a Hamming graph is utilized for the local group topology. The remaining case in the table depends on $t_3 = b$, the same number of 3-level links as in 2-level groups. With such trunking, minimal routes are shortened to *lmlgl*. Larger values of trunking up to $t_3 = ab$ could shorten paths to *lmlg*, yet this obviously leads to over dimensions of the system.

Figure 13 shows the scalability for 3-level Dragonfly networks with trunking levels t_2 and t_3. The figure also includes the 3D Hamming graph $\left(t_2 = a, t_3 = a^2, k = 3\right)$, a canonical 3-level Dragonfly network without trunking ($k = 7$), and a cascade-like Dragonfly network $\left(t_2 = a, t_3 = 1, k = 5\right)$. It is also shown from the figure that with global trunking, systems do not reach the maximum size for a given router and a diameter, which can be in the order of millions of nodes.

Conclusion

In this work, we have studied the relationship between Hamming graphs and Dragonfly networks with a proper global link arrangement and applied them in SDN. Hamming graphs represent an extreme case of a Dragonfly network, but with a large level of trunking. Networks based on Hamming graphs allow for deadlock avoidance mechanisms based on route restrictions (DOR) and do not require VCs. So, Dragonfly networks with trunking based on coloring and ordering of the network resources use DOR mechanisms to reduce the number of VCs. Trunking is required to build systems that do not reach the maximum achievable size for a given router and a diameter, which can be in the order of millions of nodes. 2D Hamming graph $K_a \square K_b$ is used for Dragonfly networks with trunking $t \geq 2$ and $t \geq 4$ that allow for 3-hop paths, and 6-hop paths, respectively, in addition to traffic randomization in both cases without a restriction on the number or use of VCs in the system. 3D Hamming graph $K_a \square K_b \square K_c$ has also been studied in this paper. It is comparable to the 2D Hamming graph, but the scalability develops quickly with the number of levels making configurations with more than 3 levels improbable. Evaluations show that with global trunking, systems are built with fewer groups than the maximum allowed. Therefore, there is no need for an additional cost.

This work on high-radix routers and networks will be very important over time, because the size of networks will continue to increase. So, interconnection networks will turn out to be more critical to system performance. The leverage resulting from 2-level and 3-level Dragonfly router designs will extend to multi-level Dragonfly networks to increase the maximum achievable system size with the same router design.

References

1. F.S. Fizi ,S. Askar in, A novel load balancing algorithm for software defined network based datacenters", International Conference on Broadband Communications for Next Generation Networks and Multimedia Applications (CoBCom), (Graz, Austria 2016), p. 1–6. https://doi.org/10.1109/COBCOM.2016.7593506
2. M. Al-Fares, A. Loukissas, A. Vahdat, A scalable, commodity data center network architecture. ACM SIGCOMM Compt. Commun. Rev. **38**(4), 63–74 (2008)
3. J. Kim, W. J. Dally, D. Abts, in *Flattened butterfly: a cost-efficient topology for high-radix networks*, Proceedings of the 34th annual international symposium on Computer architecture. (2007)
4. M.F. Bari, R. Boutaba, R. Esteves, Data center Network virtualization: a survey. IEEE Commun. Surv. Tutor. **15**(2), 909–928 (2013). https://doi.org/10.1109/SURV.2012.090512.00043
5. T.A. Yang, N. Joshy, E. Rojas, S. Anumula, J. Moola, Virtualization and data center design. Glob. J. Technol. **9**, 36–54 (2015)
6. E.-S. Jung, V. Vishwanath, R. Kettimuthu, in *Distributed Multipath Routing Algorithm for Data Center Networks*, International Workshop on Data Intensive Scalable Computing Systems, New Orleans, (LA, USA, 2014), p. 49–56. https://doi.org/10.1109/DISCS.2014.14
7. N. Maksic, A. Smiljanic, Improving utilization of data center networks. IEEE Commun. Mag. **51**(11), 32–38 (2013). https://doi.org/10.1109/MCOM.2013.6658649
8. Z. Han, W. Ren, A novel wireless sensor networks structure based on the SDN. Int. J. Distrib. Sens. Netw. **10**(3), 874047 (2014)
9. N. McKeown, T. Anderson, H. Balakrishnan, OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)
10. D.R. Hawtin, s-Elusive codes in Hamming graphs. Des. Codes Crypt. **89**, 1211–1220 (2021)
11. J. Barta, R. Montemanni (2017) Hamming Graphs and Permutation Codes Fourth International Conference on Mathematics and Computers in Sciences and in Industry (MCSI), Corfu Greece. https://doi.org/10.1109/MCSI.2017.35
12. J. Kim, W.J. Dally, S. Scott, D. Abts, Technology-driven, highly-scalable dragonfly topology. ACM SIGARCH Computer Architecture News. **36**(3), 77–88 (2008)
13. B. Arimilli, R. Arimilli, V. Chung, S. Clark, W. Denzel, B. Drerup, R. Rajamony, in *The PERCS high-performance interconnect*, 18th IEEE Symposium on High Performance Interconnects. (2010)
14. A. Agarwal, Limits on interconnection network performance. IEEE Trans. Parallel Distrib. Syst. **2**(4), 398–412 (1991). https://doi.org/10.1109/71.97897
15. W.J. Dally, Performance analysis of k-ary n-cube interconnection networks. IEEE Trans. Comput. **39**(6), 775–785 (1990). https://doi.org/10.1109/12.53599
16. J. Kim, W. J. Dally, B. Towles, A. K. Gupta, *Microarchitecture of a high radix router*, 32nd International Symposium on Computer Architecture (ISCA'05), (Madison, WI, 2005). p. 240–431. https://doi.org/10.1109/ISCA.2005.35
17. P. P. Sahu, A new shared protection scheme in optical network, Current science. p. 1176–1183 (2006)
18. F.L. Verdi, C. Carvalho, M.F. Magalhães, Policy-based grooming in optical networks. J. Netw. Syst. Manage. **16**(4), 325–349 (2008)
19. C. Camarero, E. Vallejo, R. Beivide, Topological characterization of hamming and dragonfly networks and its implications on routing. ACM Trans. Archit. Code Optim. (TACO). **11**(4), 1–25 (2014)
20. M. García, E. Vallejo, R. Beivide, in *On-the-Fly Adaptive Routing in High-Radix Hierarchical Networks*, 41st International Conference on Parallel Processing, (Pittsburgh, PA, 2012), p. 279–288. https://doi.org/10.1109/ICPP.2012.46
21. B. Lantz, B. Heller, N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks", Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks. (2010)
22. G. Faanes, A. Bataineh, D. Roweth, in *Cray Cascade: A scalable HPC system based on a Dragonfly network*, Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, (Salt Lake City, UT, 2012). p. 1–9. https://doi.org/10.1109/SC.2012.39
23. A. Singh, in *Load-Balancing Routing in Interconnection Network*, Diss. PhD Dissertation. Standford University. Advisor (s) William J. Dally. (2005)
24. A.J. Hoffman, R.R. Singleton, On moore graphs with diameters 2 and 3. IBM J. Res. Dev. **4**(5), 497–504 (1960). https://doi.org/10.1147/rd.45.0497