Journal of Computer and System Sciences 89 (2017) 328–348

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

On the combinatorial design of data centre network topologies ☆,☆☆

Iain A. Stewart

School of Engineering and Computing Sciences, Durham University, Science Labs, South Road, Durham DH1 3LE, UK

ARTICLE INFO

Article history:
Received 28 April 2016
Received in revised form 26 January 2017
Accepted 29 May 2017
Available online 13 June 2017

Keywords:
Data centre networks
Switch-centric data centre networks
Fat-Trees
Combinatorial designs
Bipartite graphs
Path diversity

ABSTRACT

The theory of combinatorial designs has recently been used in order to build switch-centric data centre networks incorporating a large number of servers, in comparison with the popular Fat-Tree data centre network. We clarify and extend these results and prove that in these data centre networks: there are pairwise link-disjoint paths joining all the servers adjacent to some switch with all the servers adjacent to any other switch; and there are pairwise link-disjoint paths from all the servers adjacent to some switch to any identically-sized collection of target servers where these target servers need not be adjacent to the same switch. In both cases, we always control the path lengths. Our constructions and analysis are undertaken on bipartite graphs with the applications to data centre networks being easily derived. Our results show the potential of the application of results and methodologies from combinatorics to data centre network design.

1. Introduction

1.1. The data centre network context

Data centres are expanding both in terms of their size and their importance as computational platforms for cloud computing, web search, social networking, and so on. There is an increasing demand that data centres incorporate more and more servers but so that overall computational efficiency is not compromised through excessive traffic. A key factor as to the eventual performance of a data centre is the *data centre network* (*DCN*); that is, the interconnection fabric of the servers and switches within the data centre. As we strive to incorporate more and more servers, new topologies are being developed so as to cope with the increase in scale and best utilize the additional computational power. It is with topological aspects of DCNs that we are concerned in this paper.

The traditional design of a DCN is *switch-centric* so that the routing intelligence resides amongst the switches, with the servers behaving only as computational nodes. In switch-centric DCNs, there are no direct server-to-server links; only server-to-switch and switch-to-switch links. Switch-centric DCNs are traditionally tree-like with servers located at the 'leaves' of the tree-like structure. Examples include ElasticTree [1], VL2 [2], HyperX [3], Portland [4], and Flattened Butterfly [5], although the dominating switch-centric DCN is Fat-Tree [6]. Whilst it is generally acknowledged that tree-like, switch-centric

☆ This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) grant 'Interconnection Networks: Practice unites with Theory (INPUT)' [grant number EP/K015680/1].

☆☆ A preliminary version of this paper appeared as an extended abstract in the *Proceedings of 20th International Symposium on Fundamentals of Computation Theory* (A. Kosowski, I. Walukiewicz, eds.), Gdansk, Poland, August 17–19 2015, Lecture Notes in Computer Science, Volume 9210, Springer, 2015, 283–295.

E-mail address: i.a.stewart@durham.ac.uk.

http://dx.doi.org/10.1016/j.jcss.2017.05.015
0022-0000/

DCNs have their limitations when it comes to, for example, scalability, due to the size of routing tables at the switches, switch-centric DCNs remain popular and can usually be constructed from commodity hardware. A more recent paradigm, namely the *server-centric* DCN, has emerged so that deficiencies of the tree-like, switch-centric DCNs might be ameliorated. Server-centric DCNs reflect that the routing intelligence resides within the servers with switches operating only as dumb crossbars. In server-centric DCNs there are only server-to-switch and server-to-server links. However, server-centric DCNs also suffer from deficiencies such as packet relay overheads caused by the need to route packets within the server; moreover, server-centric DCNs have yet to make it into the commercial mainstream (the reader is referred to [7] for an overview of the state of the art as regards DCN architectural design). It is with the construction of switch-centric DCNs that we are concerned here.

It is extremely difficult to design computationally efficient (switch-centric) DCNs so as to incorporate large numbers of servers as there are many additional considerations to take into account. For example, switches and (especially) servers in data centres have a limited number of ports with a consequence being that the more servers there are, the greater the average or worst-case link-count between two distinct servers; hence, there is a packet latency overhead to be borne. Also, so as to better support routing, fault-tolerance, and load-balancing, we would prefer that there are numerous alternative paths within the DCN joining any two distinct servers; that is, that there is *path diversity*. Irrespective of the DCN paradigm within which one works, there are many other design parameters to bear in mind relating to, for example, (incremental) scalability, throughput, cost, oversubscription, energy consumption, latency, and security (see, for example, [8,9] for an overview). The upshot is that the DCN designer has to simultaneously secure a number of performance characteristics, some of which are competing against each other; this makes the DCN design space complex and difficult to work in.

1.2. Using combinatorial designs to build DCNs

A recent proposal in [10] advocated the use of *combinatorial design theory* in order to design switch-centric DCNs; these DCNs have beneficial properties as regards incorporating more servers and possessing path diversity yet it is possible to limit the worst-case link-length of server-to-server shortest paths (and so, ultimately, achieve better control over packet latency in a DCN). The use of combinatorial designs within the study of general interconnection networks is not new and originated in [11] where the targeted networks involved processors communicating via buses (the reader is referred to [12] for a range of applications of combinatorial design theory within computer science). A hypergraph framework was developed in [11] where the hypergraph nodes represent the processors and the hyperedges the buses. Likewise, an analogous framework was developed in [10] but where the hypergraph nodes and edges both represent switches so that the pendant servers 'hang off' some of the switches (we present a detailed description of this framework in Section 3.3). In [10], the ubiquitous switch-centric Fat-Tree DCN from [6] was used as a yardstick against which to compare the new DCN designs developed in [10] under the normalization that all DCNs are to have the same worst-case link-length of server-to-server shortest paths, namely 6, as this equals the worst-case link-length of server-to-server shortest paths in the Fat-Tree DCN. It was shown that more servers can be incorporated within the new DCNs yet, crucially, the resulting DCNs have good path diversity. It is the algebraic properties (relating to symmetry and balance) possessed by transversal designs that enable the constructions and analysis as described in [10]. One slight difficulty with the original and novel approach taken in [10] is that some of the path diversity results derived there are incorrect (as we explain later in Section 4.1). Not only has combinatorial design theory featured as regards the design of interconnection networks but other aspects of algebra have too; indeed, there has been recent work on the relevance of Cayley graphs, Hamming graphs, and hyperbolicity to DCN design (see, e.g., [13–15]).

1.3. Our contribution

In this paper we return to the framework of [10] and formulate and prove path diversity results for the switch-centric DCNs constructed using the methods of that paper. As our concern is entirely with topological properties of DCNs, henceforth we abstract our DCNs as undirected graphs where the nodes are to represent servers and switches and the edges point-to-point links. The crux of the construction in [10] is (essentially) to build a bipartite graph using a systematic method, called the 3-step method, involving a different 'base' bipartite graph and a transversal design, and to convert the resulting bipartite graph into switch-centric DCNs (in a variety of ways). After explaining how hypergraphs and transversal designs can all be considered as bipartite graphs in Section 2, in Section 3 we provide a detailed description of the 3-step framework from [10] and explain how the bipartite graphs constructed are converted into switch-centric DCNs. Next, we revisit the results from [10]. In particular, in Section 4 we correct and extend the analysis in [10] and affirm that using the 3-step method from [10], we can build switch-centric DCNs: with many more servers than the Fat-Tree DCN yet so that, like the Fat-Tree, every server-to-server shortest path has length at most 6; and so that (assuming some numeric conditions on the base bipartite graph and the transversal design) we can find pairwise link-disjoint paths from all of the servers adjacent to a particular switch to all of the servers adjacent to any other switch. Moreover, we provide an upper bound on the lengths of the paths constructed in terms of the diameter of the base bipartite graph (see Theorem 4). We also deal with a scenario missing from [10] (see part (*b*) of Theorem 4). As we explain, the general situation is more subtle than was assumed in [10].

The DCN path diversity, as we have described it above, comes about from building bipartite graphs (which are subsequently converted to DCNs) so that given any two distinct nodes, there are numerous node-disjoint paths joining these two

nodes; that is, these bipartite graphs have one-to-one path diversity. In Section 5, we go on to show that we can actually build numerous edge-disjoint paths from a source node to different destination nodes in our bipartite graphs; that is, we have one-to-many path diversity (one-to-one and one-to-many path diversity are defined in Section 2.1). The DCNs obtained from these bipartite graphs are such that (assuming some numeric conditions on the base bipartite graph and the transversal design) we can find pairwise link-disjoint paths from all of the servers adjacent to some switch to any identically-sized collection of servers (irrespective of which switch they are adjacent to). Consequently, we show that our DCNs provide support for additional communication patterns that are prevalent within data centre networks. It should be noted that one-to-many and many-to-many communication patterns are commonplace in data centres; for example, in 'big data' processing applications such as MapReduce, Hadoop, Spark, and Storm (see, e.g., the survey [16]).

This paper is unashamedly theoretical. However, we demonstrate that not only is there interesting combinatorics within the practical world of DCN design but that combinatorial mathematics can potentially contribute to the DCN design space on a practical level. We feel that the mathematical aspects of DCNs have so far remained almost completely unexamined and we advocate a closer theoretical scrutiny of DCNs both as a model of computation and in relation to the vast swathes of research on general interconnection networks. We mention some practical considerations and directions for further research in the Conclusion.

2. Basic concepts

We begin by briefly reviewing some architectural aspects of switch-centric DCNs that are pertinent to our subsequent research. We then move on to the discrete structures featuring in [10,11], namely hypergraphs, bipartite graphs, and transversal designs. So that we might fully describe and understand the constructions in [10,11], as well as our own upcoming analysis of switch-centric DCNs, we eventually amalgamate hypergraphs, bipartite graphs, and transversal designs so that by the end of this section, we will have developed an encompassing bipartite graph framework for the design of switch-centric DCNs. The reader should be aware that it will not be until Section 3.3 that we transform the bipartite graphs that we have been constructing up until then into switch-centric DCNs. As a hint as to this transformation (and so that the reader does not lose sight of our eventual goal), roughly speaking we shall regard the nodes of one of our constructed bipartite graphs as switch-nodes and attach to some of these switch-nodes additional server-nodes in order to get our switch-centric DCN. General graph-theoretic concepts can be obtained in [17].

2.1. Switch-centric DCNs

A switch-centric DCN is abstracted as a graph (which we also refer to as a DCN) where the nodes are partitioned into two sets: there are *server-nodes*; and there are *switch-nodes*. Of course, the server-nodes correspond to servers in the DCN and the switch-nodes to switches; note that immediately there are practical design limitations imposed by the number of *ports* in a real switch and the number of *NIC ports* in a real server (we sometimes refer to the number of ports of a switch-node rather than its degree). Furthermore, in switch-centric DCNs there are no links joining one server-node directly to another server-node (because all routing within a switch-centric DCN falls within the purview of the switches). Of concern to us in this paper will be incorporating a comparatively large number of server-nodes within our DCNs but so that the maximum length of a shortest path joining any two server-nodes, that is, the *diameter* of the DCN, is kept within a given bound, where the *length* of such a path is the number of distinct links on the path. Essentially, we will be comparing DCNs as to how many server-nodes they incorporate but when their diameters are normalized.

However, DCNs must also possess other properties to make them usable within a data centre context. For example, they also need to: be scalable and incrementally scalable (that is, have the capacity to cope with increases in components and data); have low message latency; provide for high overall throughput (under a range of traffic patterns); be able to tolerate (a limited number of) faults; be energy efficient; be both economically and physically viable; and support virtualization (that is, the partitioning of the DCN into virtual networks on a dynamic basis), amongst many other things. Support for some of these properties can be measured using graph theory; for example, the diameter of the DCN gives guidance as regards the expected message latency. Of particular interest to us will be *path diversity* which we define (somewhat informally) as the capacity to send data without inducing additional congestion or so as to cope with existing congestion or faults. There are two contexts of interest to us: the *one-to-one* (or *unicast*) context, when a source server-node wishes to send data to a destination server-node by the utilization of independent paths (we will return to what we mean by 'independent' soon); and the *one-to-many* (or *multicast*) context, when a source server-node wishes to send data to a number of destination server-nodes so that the different transmissions do not induce congestion. Path diversity is highly relevant to a number of the above properties such as latency and scalability, where different paths are used to split and balance loads, and fault tolerance, where different paths provide alternative means of transit in the case of faults. Path diversity is important in both the one-to-one and one-to-many contexts, with this importance accentuated in the latter context when a data centre needs to support data replication and applications like MapReduce [18]. An additional dimension is added with respect to virtualization when we have virtual machines embedded within a data centre that share the same resources but require traffic to be routed via different routes. As we shall soon see, just as with latency, the independence of paths can be considered graph-theoretically.

2.2. Hypergraphs

Hypergraphs provide the original framework for the 3-*step construction* (to be defined later) as employed in [11,10]: in [11], hypergraphs were used to model bus interconnection networks; and in [10], hypergraphs were used to model data centre networks. For the moment, and in order to appreciate the context of [11,10], we retain this hypergraph framework before we phrase all content in this introduction within an encompassing bipartite graph framework.

A *hypergraph* $H = (V, E)$ consists of a finite set V of *nodes* together with a finite set E of *hyperedges* where each hyperedge is a non-empty set of nodes and each node appears in at least one hyperedge. The *degree* of a node is the number of hyperedges containing it and the *rank* of a hyperedge is its size as a subset of V. A hypergraph is *regular* (resp. *uniform*) if every node has the same degree (resp. every hyperedge has the same rank) with this degree (resp. rank) being the *degree* (resp. *rank*) of the hypergraph. Every graph $G = (V, E)$ has a natural representation as a hypergraph: the nodes of the hypergraph are V; and the hyperedges are E, where the hyperedge e consists of the pair of nodes incident with the edge e of G.

2.3. Hypergraphs and bipartite graphs

We can represent any hypergraph $H = (V, E)$ as a bipartite graph: the node set of the bipartite graph is $V \cup E$; and there is an edge (v, e), for $v \in V$ and $e \in E$, in the bipartite graph if, and only if, $v \in e$ in the hypergraph. It is clear that this yields a one-to-one correspondence between hypergraphs and bipartite graphs (without isolated nodes) that come complete with a partition of the elements into a 'left-hand side', which will correspond to the nodes of the hypergraph, and a 'right-hand side', which will correspond to the hyperedges of the hypergraph (remember that in a hypergraph, every node is in at least one hyperedge and every hyperedge contains at least one node, so we cannot have isolated nodes in our bipartite graphs). We assume (henceforth) that every bipartite graph comes equipped with such a partition and for clarity from now on we refer to the nodes on the left-hand side as *nodes* and the nodes on the right-hand side as *blocks* (this is in keeping with our upcoming realisation of transversal designs as bipartite graphs). Likewise, we refer to the degree of a node as its *degree* and the degree of a block as its *rank*. A bipartite graph corresponding to a regular, uniform hypergraph of degree d and rank Δ is called a (d, Δ)-*bipartite graph*. Every bipartite graph (and so every hypergraph) also describes its *dual bipartite graph* (or alternatively its dual hypergraph) where the roles of the nodes on the left-hand side and the blocks on the right-hand side of the partition are reversed in the definition of the bipartite graph; so, for example, the dual bipartite graph of a (d, Δ)-bipartite graph is regular of degree Δ and uniform of rank d.

Note that if G is a bipartite graph then it corresponds to a hypergraph via our representation above and it also corresponds to a hypergraph via the natural representation highlighted in Section 2.2. The two hypergraphs corresponding to the same bipartite graph are different and we are never interested in the representation of a bipartite graph as a hypergraph via the natural representation of Section 2.2.

2.4. Paths in hypergraphs

A *path* in some hypergraph $H = (V, E)$ (or the corresponding bipartite graph) is an alternating sequence of nodes and hyperedges so that all nodes are distinct, all hyperedges are distinct, and a node $v \in V$ follows or precedes a hyperedge $e \in E$ in the sequence only if $v \in e$ in the hypergraph (or (v, e) is an edge in the corresponding bipartite graph). The first element of some path is the *source* and the final element the *destination*. The *length* of any path is its length in the bipartite graph corresponding to the hypergraph, and the *distance* between two distinct elements of $V \cup E$ is the length of a shortest path joining these two elements in the corresponding bipartite graph. The *diameter* of H is the maximum of the distances between every pair of distinct nodes of V, and the *line-diameter* of H is the maximum of the distances between every pair of distinct hyperedges of E.

We have two remarks. First, we have traditional notions of diameter and line-diameter in any bipartite graph. Note that our notion of diameter in a bipartite graph, which is the longest shortest node-to-node path (and so ignores node-to-block and block-to-block paths), is different from the usual graph-theoretic notion of diameter in a bipartite graph (the same comment can be made as regards line-diameter). When we talk of the diameter or line-diameter of a bipartite graph, we mean with respect to *our* notion of diameter or line-diameter, respectively; if we need to talk of the traditional notion of graph diameter then we will make this clear. Second, our notion of path length in a hypergraph differs from that in [10] where the length is the number of nodes (resp. hyperedges) in a hyperedge-to-hyperedge (resp. node-to-node) path. There is no real consequence to this difference; essentially, our notion of path length is double that in [10]. However, we shall soon move to an exclusively bipartite graph-theoretic formulation in which our notion of length is the natural one to adopt.

We shall be interested in building sets of paths in some hypergraph H so that the paths might have the same sources or destinations; moreover, we shall require that these paths do not 'interfere' with one another (or are 'independent' as we mentioned earlier). We say that a set P of paths in H is:

- *pairwise internally-disjoint* if any source or destination of some path of P only appears as a source or destination on any path of P, and any node or hyperedge that is not a source or destination appears on at most one path of P

- *pairwise edge-disjoint* if every pair $(v, e) \in V \times E$ is such that v follows or precedes e on some path at most once across all paths from the set P.

2.5. Hypergraphs as switch-centric DCNs

Given some hypergraph $H = (V, E)$, our intention is to ultimately transform this hypergraph into a DCN by considering both the nodes and the hyperedges as switch-nodes so that the switch-nodes corresponding to the nodes (which we shall later call the level-1 switch-nodes, with the switch-nodes corresponding to the hyperedges the level 2-switch-nodes) also have adjacent server-nodes, which we have yet to define (this intention is best appreciated by working with the corresponding bipartite graph rather than the hypergraph; the upcoming Fig. 5 provides a visualization of what we mean). Consequently, we can regard a hypergraph H as modelling a switch-centric DCN N where there are two levels of switch-nodes.

Suppose that we have a set P of pairwise internally-disjoint paths from a node u of H to another node v of H. This translates to a set P' of pairwise internally-disjoint paths in N from the corresponding level-1 switch-node u to the corresponding level-1 switch-node v. We can use the paths of P' for the simultaneous transfer of data from server-nodes adjacent to the level-1 switch-node u to server-nodes adjacent to the level-1 switch-node v (see Fig. 5). In order to facilitate this data transfer we require that level-1 switch-nodes are non-blocking whereas the level-2 switch-nodes can be blocking; recall that a switch-node is *non-blocking* when no contention arises when simultaneously sending data through the switch-node on two distinct input links and out on two distinct output links, and *blocking* otherwise. This is because we need to be able to simultaneously move data from all servers adjacent to the level-1 switch-node u in N across the switch-node and out along different links (the same can be said for v). If our paths in H are only pairwise edge-disjoint then we require that level-1 and level-2 switch-nodes of N are non-blocking (as we might have switch-nodes appearing on more than one path of P', even though no link does).

2.6. Transversal designs

The notion of a transversal design is crucial to what follows.

Definition 1. Let $k, \Delta \geq 2$. A $[\Delta, k]$-*transversal design* T is a triple $(\mathcal{X}, \mathcal{D}, \mathcal{U})$ where: $|\mathcal{X}| = \Delta k$; $\mathcal{D} = (D_1, D_2, \ldots, D_\Delta)$ is a partition of $\mathcal{X}$ into Δ equal-sized *groups* (each of size k); and $\mathcal{U} = \{U_j : j = 1, 2, \ldots, k^2\}$ is a family of k^2 subsets of $\mathcal{X}$, each of size Δ and called a *block*, so that

- $|D_i \cap U_j| = 1$, for $i = 1, 2, \ldots, \Delta$, $j = 1, 2, \ldots, k^2$
- each pair of elements $\{x_i, x_j\}$, where $x_i \in D_i$, $x_j \in D_j$ and $i \neq j$, is contained in exactly 1 block.

We adopt a graph-theoretic perspective on transversal designs as defined in Definition 1: we think of the $[\Delta, k]$-transversal design T as a bipartite graph where the elements of $\mathcal{X}$ (resp. $\mathcal{U}$) lie on the left-hand side (resp. right-hand side) of the partition, and so are called nodes (resp. blocks) within the bipartite graph, and so that in this bipartite graph there is an edge (p, Q), for $p \in \mathcal{X}$ and $Q \in \mathcal{U}$, if, and only if, in the transversal design the element p is in the block Q. Note that the bipartite graph corresponding to the transversal design from Definition 1 is a (k, Δ)-bipartite graph. Henceforth, we adopt our bipartite graph framework and regard both hypergraphs and transversal designs as bipartite graphs (unless we state otherwise).

There is an intimate relationship involving transversal designs, *orthogonal arrays* and *mutually orthogonal latin squares*, although there is no need to give definitions here. However, it is well known: that there are Δ mutually orthogonal latin squares of order k if, and only if, there is a $[\Delta + 2, k]$-orthogonal array if, and only if, there is a $[\Delta + 2, k]$-transversal design; and that there are at most $k - 1$ mutually orthogonal latin squares of order k (see, for example, [19]). Hence, if we have a $[\Delta, k]$-transversal design then $\Delta \leq k + 1$. Also, if k is a prime power then a $[\Delta, k]$-transversal design exists whenever $2 \leq \Delta \leq k + 1$ (again, see [19]). We shall use these facts later on. The study of the existence of $[\Delta, k]$-transversal designs, for various Δ and k, is a long-standing area of research.

We require one final bit of notation. If T is some transversal design, as in Definition 1, and x and y are nodes in distinct groups then we refer to the unique block adjacent to both x and y as the block *generated* by x and y.

3. The 3-step construction and its extensions

We now describe the 3-*step construction* for building bipartite graphs (or, equivalently, hypergraphs) by using a 'base' bipartite graph and a transversal design (which we think of as a bipartite graph). This construction originated in [11] and was used in [10]. We then explain how this construction was subsequently extended in [10] both by iteration and by composition so as to yield switch-centric DCNs.

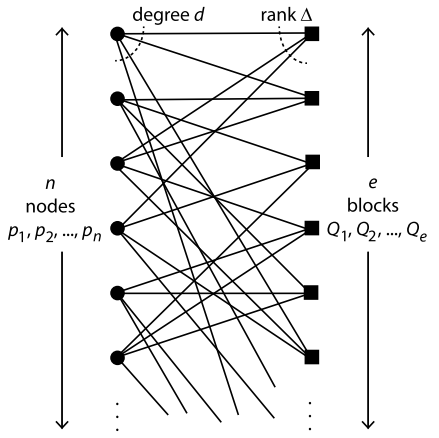

Fig. 1. A (d, Δ)-bipartite graph H_0.

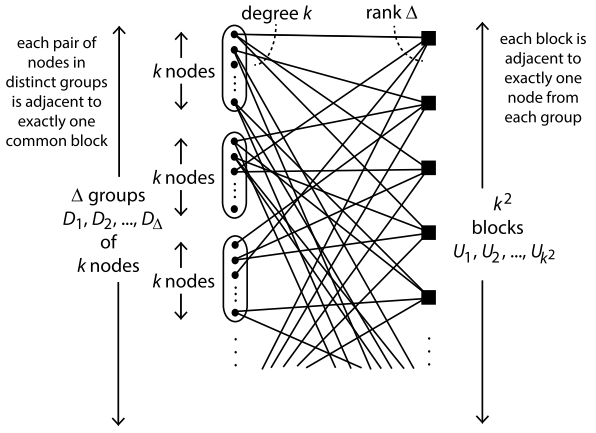

Fig. 2. A $[\Delta, k]$-transversal design T.

3.1. *The* 3*-step construction*

The 3-step construction proceeds as follows.

Step 1: Let H_0 be a connected (d, Δ)-bipartite graph so that there are n nodes (on the left-hand side of the partition, each of degree d) and e blocks (on the right-hand side, each of rank Δ). Such an H_0 can be visualized as in Fig. 1 (ordinarily, we represent nodes as circles and blocks as squares).

Step 2: Let T be a $[\Delta, k]$-transversal design. In particular, there are Δ groups of k nodes (on the left-hand side) as well as k^2 blocks (on the right-hand side). Such a T can be visualized as in Fig. 2. Build the bipartite graph H as follows. For every node p of H_0, introduce a group G_p of k nodes of H; we say that the group of nodes G_p of H is *associated with* the node p of H_0. For every block Q of H_0, adjacent to the nodes $p_1, p_2, \ldots, p_\Delta$ in H_0, introduce a copy of T, denoted T_Q, rooted on the Δ groups of nodes $G_{p_1}, G_{p_2}, \ldots, G_{p_\Delta}$; so, *associated with* the block Q of H_0, we have a set B_Q of k^2 blocks in H. We refer to the Δ groups of nodes $G_{p_1}, G_{p_2}, \ldots, G_{p_\Delta}$ as the *roots* of the copy T_Q of T in H. Such a bipartite graph H can be visualized as in Fig. 3 where two of the copies of T are partially shown (note that they might have some roots in common but their respective sets of blocks are always disjoint as are their sets of edges). The bipartite graph H_0 provides a template as to how we introduce copies of T to form H.

Note that:

- each node of H can be indexed as $a_{p,j}$, where $p \in \{p_i : i = 1, 2, \ldots, n\}$ and $j \in \{1, 2, \ldots, k\}$, so that p is the node of H_0 to which the group G_p in which $a_{p,j}$ sits is associated and j is the index of the node $a_{p,j}$ in this group
- each block of H can be indexed as $B_{Q,U}$, where $Q \in \{Q_i : i = 1, 2, \ldots, e\}$ and $U \in \{1, 2, \ldots, k^2\}$, so that Q is the block of H_0 to which the set of blocks B_Q in which $B_{Q,U}$ sits is associated and U is the block of T to which $B_{Q,U}$ corresponds.

In addition, each node of T can be indexed $u_{i,j}$, where $i \in \{1, 2, \ldots, \Delta\}$ and $j \in \{1, 2, \ldots, k\}$, so that D_i is the group of nodes in which $u_{i,j}$ sits and j is the index of $u_{i,j}$ in that group.

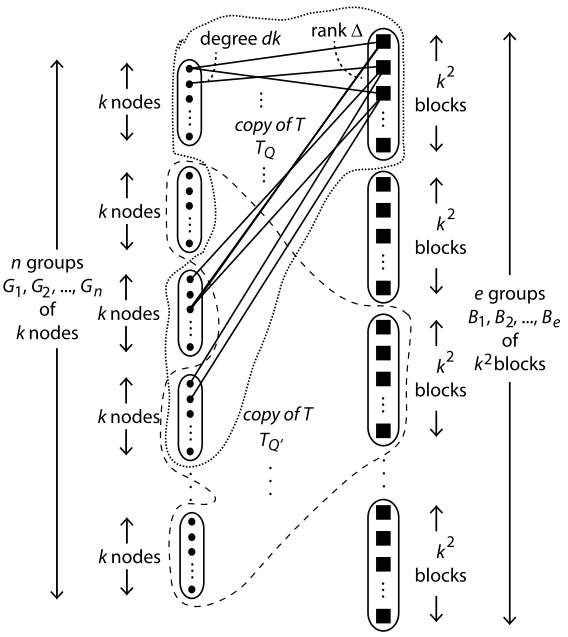

Fig. 3. Amalgamating H_0 and T to get H.

Step 3: Let H^* be the bipartite graph obtained from the bipartite graph H by reversing the roles of nodes and blocks (so, H^* is the dual bipartite graph of H). Note that the bipartite graph H^* is regular of degree Δ and uniform of rank dk.

We refer to the (dk, Δ)-bipartite graph H (resp. the (Δ, dk)-bipartite graph H^*) constructed above as having been constructed by the 2-step (resp. 3-step) method using the (d, Δ)-bipartite graph H_0 and the $[\Delta, k]$-transversal design T. Note that H (resp. H^*) has nk nodes (resp. ek^2 nodes) and ek^2 blocks (resp. nk blocks).

Our intention with our constructions is to ultimately design switch-centric DCNs with beneficial properties (as we outlined in Section 2). Whilst there are many properties we would like our DCNs to have, it is important that DCNs can integrate a large number of server-nodes so that the server-node-to-server-node distances are short and so that there is redundancy as to which (short) server-node-to-server-node routes we choose to use. In our framework of bipartite graphs, this translates as building bipartite graphs with a large number of nodes and with redundant (short) node-to-node paths. As a first step, the following result was proven in [11] (it is actually derivable from the proofs of our upcoming results) and allows us control over the length of shortest block-to-block paths in 2-step constructions (and so shortest node-to-node paths in 3-step constructions).

Theorem 2 *([11]). Suppose that the (dk, Δ)-bipartite graph H has been constructed using the 2-step method using the (d, Δ)-bipartite graph H_0 and the $[\Delta, k]$-transversal design T. If H_0 has line-diameter $\lambda \geq 4$ then H has line-diameter λ.*

Of course, if H^* is the dual bipartite graph of H in Theorem 2 then it has diameter λ. We reiterate that our notion of diameter and line-diameter differs from that in [11,10] (where the length of a block-to-block path is the number of nodes on that path; so, in [11,10] the bound $\lambda \geq 4$ in our Theorem 2 appears as $\lambda \geq 2$).

3.2. Iteration

We can iterate the 3-step construction (as was done in [10]). Note that if H_0 is a (d, Δ)-bipartite graph of line-diameter $\lambda \geq 4$, with n nodes and e blocks, then the bipartite graph H_1 resulting from the 2-step construction (using H_0 and some $[\Delta, k]$-transversal design T) is a (dk, Δ)-bipartite graph of line-diameter λ. So, repeating the 2-step construction but with H_1 replacing H_0 (we keep the same T, although we do not have to) yields a (dk^2, Δ)-bipartite graph H_2 of line-diameter λ. By iterating this construction, we can clearly obtain a (dk^i, Δ)-bipartite graph H_i of line-diameter λ. Converting H_i into H_i^* results in a bipartite graph with ek^{2i} nodes, with nk^i blocks, with diameter λ, and that is regular of degree Δ and uniform of rank dk^i.

3.3. Composition

We are now in a position to transform our bipartite graphs into switch-centric DCNs. As well as the constructions, and their associated proofs, that were presented in [10], new methods of composing bipartite graphs (built according to the

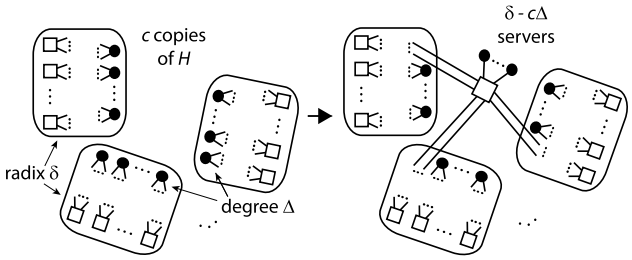

Fig. 4. Building a switch-centric DCN via Method A when $c > 1$.

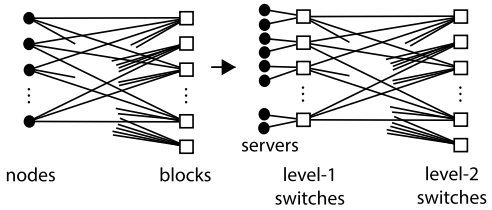

Fig. 5. Building a switch-centric DCN via Method A when $c = 1$.

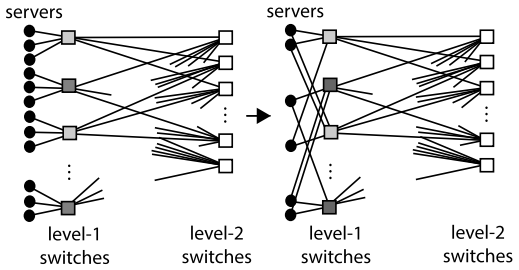

Fig. 6. Building a switch-centric DCN via Method B.

3-step construction) so as to obtain switch-centric DCNs were also derived. In [10], 4 such methods were given: Methods M_1, M_2 and M_3 are different cases of Method A, below; and Method M_4 is Method B.

In what follows, let H be a (Δ, δ)-bipartite graph where $\Delta < \delta$ and where there are n nodes and e blocks.

Method A: We take c copies of H where $\delta - c\Delta > 0$ and $c \geq 1$. For each node u of H: we remove the corresponding node in each of the c copies of H and introduce a new switch-node (common to all copies of H); we make all of the $c\Delta$ edges incident with the c original nodes incident with this new switch-node; and we attach $\rho = \delta - c\Delta$ pendant server-nodes to the new switch-node. All blocks of H are considered as switch-nodes. We follow [10] and call the new switch-nodes *level*-1 *switch-nodes*, and the original switch-nodes *level*-2 *switch-nodes*. The construction of the switch-centric DCN $N(H)$ from H via this method can be visualised as in Fig. 4, where we only show the construction for the c nodes corresponding to one node of H. Note that every switch-node of $N(H)$ has δ ports. Also, there is some choice as regards the parameter c (so that choosing different values for c yields different values for ρ). We illustrate the special case when $c = 1$ in Fig. 5, where H is a $(3, 5)$-bipartite graph. The general case when $c \geq 1$ corresponds to Method M_2 of [10]; the special case when $c = 1$ corresponds to Method M_1; and the special case when $c = \lfloor \frac{\lfloor \frac{\delta}{2} \rfloor}{\Delta} \rfloor$ corresponds to Method M_3. In this latter case, the aim is to ensure that every level-1 switch-node is adjacent to roughly the same number of level-2 switch-nodes as it is server-nodes. Note that: the number of server-nodes in $N(H)$ is $n(\delta - c\Delta)$; the number of level-1 switch-nodes is n; and the number of level-2 switch-nodes is ce.

Method B: We now work with a switch-centric DCN as constructed by Method A. Let every level-1 switch-node have ρ adjacent server-nodes. Suppose that there is an even number of level-1 switch-nodes. Partition the set of level-1 switch-nodes into pairs. For each pair of switch-nodes (S', S''): remove $\lfloor \frac{\rho}{2} \rfloor$ server-nodes that are adjacent to S' and remove $\lceil \frac{\rho}{2} \rceil$ server-nodes that are adjacent to S''; and make every server-node that is adjacent to the switch-node S' or the switch-node S'' also adjacent to the other switch-node. Note that the number of ports of any switch-node has not changed but that every server-node is now adjacent to 2 switch-nodes. The philosophy behind this construction is to better tolerate the failure of a level-1 switch-node. The construction can be visualized as in Fig. 6 where paired level-1 switch-nodes have the same shade of grey and where $\rho = 3$.

Table 1
Comparing switch-centric DCNs built with switch-nodes with 64 ports.

Network	# switch ports	Diameter	# server-nodes	# switch-nodes
Fat-Tree	64	6	65,536	5,120
H^*	64	4	54,720	6,840
$N_A^1(H^*)$	64	6	3,064,320	61,560
$N_A^2(H^*)$	64	6	437,760	102,600
$N_A^3(H^*)$	64	6	1,751,040	82,080
$N_B(H^*)$	64	6	1,532,160	61,560
$\bar{H}^*$	64	4	20,480	1,280
$N_A^1(\bar{H}^*)$	64	6	1,228,800	21,760

3.4. Some illustrations of DCNs

In [10], switch-centric DCNs constructed using the 3-step method allied with Methods A and B were favourably compared with the 3-level Fat-Tree DCN from [6] with regard to the number of server-nodes therein when the diameter and the number of ports of a switch-node are held constant. The reader is referred to [6,10] for full details as regards the topology of Fat-Tree and to Tables 2–4 in [10] for the complete comparison; however, we include a replicated table here purely for illustrative purposes. In Table 1 (which is Table 2 from [10]): the number of ports of any switch-node is forced to be 64; the diameters of the DCNs resulting from using the 3-step method, iteration and composition are forced to be (at most) 6 (like that of Fat-Tree); and the numbers of server-nodes and switch-nodes in the resulting DCNs are as given (note that the length of a server-node-to-server-node path as defined in [10] is the number of switch-nodes on it, which is one less than our notion of length which is the number of links on the path).

- The bipartite graph H^* is obtained using the 3-step method starting with a $(8, 8)$-bipartite graph H_0, that has 855 nodes, 855 blocks, and diameter and line-diameter 4 (such a bipartite graph H_0 exists; see [20]), and a $[8, 8]$-transversal design T. The DCN H^* in Table 1 is the DCN obtained by simply regarding every node of the bipartite graph H^* as a server-node (note that in this DCN we require that every server-node has 8 NIC ports); the DCN $N_A^1(H^*)$ (resp. $N_A^2(H^*)$, $N_A^3(H^*)$) is obtained by employing Method A with $c = 1$ (resp. $c = 7$, $c = 4$); and the DCN $N_B(H^*)$ is obtained by employing Method B with $N_A^1(H^*)$ (note that the number of switch-nodes entry in Table 2 in [10] is incorrect).
- The bipartite graph $\bar{H}$ is obtained using the 3-step method iterated twice, starting with a $(4, 4)$-bipartite graph $\bar{H}_0$, that has 80 nodes, 80 blocks, and diameter and line-diameter 4 (such a bipartite graph $\bar{H}_0$ exists; see [20]), and a $[4, 4]$-transversal design $\bar{T}$ (actually, in [10] this transversal design is not mentioned; it does, however, exist). The DCN $\bar{H}^*$ in Table 1 is the DCN obtained by simply regarding every node of the bipartite graph $\bar{H}^*$ as a server-node (note that the number of server-nodes entry in Table 2 in [10] is incorrect, though the correct number is stated in the text); and the DCN $N_A^1(H^*)$ is obtained by employing Method A with $c = 1$ (note that the numbers of server-nodes and of switch-nodes entries in Table 2 in [10] are incorrect).

It is clear from Table 1 (and from [10]) that we can build much bigger server-centric DCNs using the 3-step method and the subsequent iterations and compositions than Fat-Tree but without increasing the diameter (which is a proxy for latency); of course, we would wish the new DCNs to have other properties that make them attractive within a data centre context. Establishing such properties was essentially the whole point of [10] and we continue with this line of research in what follows. Note that we provide additional illustrations of our constructions of switch-centric DCNs, in tandem with our upcoming results, in Section 4.3.

Before we move to our main results, let us comment on using the 2-step method as opposed to the 3-step method when building our switch-centric DCNs (the same comment was made in [10]). Note that when one uses the (iterated) 2-step method, whilst the rank of the resulting bipartite graph stays the same, the degree grows. Were we to attach server-nodes to the switch-nodes that replace the nodes of the 2-step bipartite graph H, rather than the 3-step bipartite graph H^*, the number of ports of the level-2 switch-nodes (which would be Δ) would be much less than the number of ports of the level-1 switch-nodes. Hence, it makes more sense to proceed as we have done above.

4. One-to-one path diversity

So far, we have set the scene from [10] and described a method by which we can build bipartite graphs (the 3-step method) which can then be transformed into switch-centric DCNs with many more servers than Fat-Tree whilst maintaining the diameter of Fat-Tree, i.e., 6. However, as we mentioned earlier, there are many more aspects to the design of DCNs with an important one being path diversity. In what follows, we highlight some problems with the proofs of one-to-one path diversity in [10] for bipartite graphs built using the 3-step method. We then provide not only correct proofs as regards one-to-one path diversity but we also extend and improve the analysis in [10] with new results. We end the section by applying our constructions so as to build DCNs with good one-to-one path diversity properties.

4.1. Difficulties with proofs

In order to detail the difficulties in [10], we adopt the terminology of [10]. There are slight problems with the proof of Theorem 2 in [10] (although they are easily surmountable). For example, in Subcases (1.2) and (2.2), $\{r_i, s_i, t_i\} \subseteq G_i^E$ and consequently we cannot generate the blocks R_j and S_j. Also, in Subcase (2.1), the situation where $q \in P \cap Q \setminus \{p\}$ is not considered; it could be that $r_j = s_j$, for some $j \neq i$.

An attempt was also made in [10] to extend Theorem 2 of [10]: see Theorem 3 of [10]. Assumptions concerning the connectivity of H_0 are made and the existence of additional paths in H^* to those constructed in the proof of Theorem 2 are claimed in the situation when the two blocks $B_{Q,U}$ and $B_{Q',U'}$ are such that $Q \neq Q'$ (recall our method of indexing in Section 3.1 which we adopt here). However, there are serious flaws in the proof of Theorem 3 of [10], so much so that the theorem is untrue. In short, Theorem 3 of [10] claims that if there are ω pairwise internally-disjoint paths in H_0 from Q to Q' then there are $\min\{\Delta\omega, k\omega\}$ pairwise internally-disjoint paths in H from $B_{Q,U}$ to $B_{Q',U'}$. This does not make sense: the maximum number of pairwise internally-disjoint paths in H from $B_{Q,U}$ to $B_{Q',U'}$ is Δ (as the bipartite graph H has rank Δ) and so we must have that $\min\{\Delta\omega, k\omega\} \leq \Delta$. For instance, in Example 1 of [10], the bipartite graph H_0 is the cycle of length 10 (H_0 is derived from the cycle of length 5 using its natural representation as a hypergraph; see Section 2.2), so that $d = \Delta = 2$, $n = e = 5$, and there are 2 internally-disjoint paths from any block of H_0 to any other block of H_0. A $[2, 3]$-transversal design T is used and the bipartite graph H^* built by the 3-step method has rank 6 and degree 2. However, if Theorem 3 of [10] were true then there would be 4 pairwise disjoint paths from $B_{Q,U}$ to $B_{Q',U'}$ in H^* which clearly cannot be the case.

4.2. The one-to-one scenario

We now resurrect (some of) the proofs of the main results from [10] and extend the results claimed in that paper. The following lemma proves most useful.

Lemma 3. *Let T be some $[\Delta, k]$-transversal design with groups of nodes $\{D_1, D_2, \ldots, D_\Delta\}$. Let U be some block of T. For each $i \in \{1, 2, \ldots, \Delta\}$, let $r_i \in D_i$ be the unique node of D_i that is adjacent to U. Set $R = \{r_i : i = 1, 2, \ldots, \Delta\}$. Let P be a set of distinct pairs of nodes so that: exactly one node of any pair in P is in R and no node of R is in more than one pair of P; and no pair in P is such that both nodes lie in the same group. The blocks generated by the pairs in P are all distinct and different from U.*

Proof. Suppose that $\{r_i, x\} \in P$, where $x \in D_l \setminus R$ with $l \neq i$ and where $i \in \{1, 2, \ldots, \Delta\}$. Let $U_{r_i,x}$ be the block generated by r_i and x. If $U_{r_i,x} = U$ then U is adjacent to the distinct nodes r_l and x in D_l which yields a contradiction.

Suppose that $\{r_j, y\} \in P \setminus \{\{r_i, x\}\}$, where $j \in \{1, 2, \ldots, \Delta\}$. Let $U_{r_j,y}$ be the block generated by r_j and y. Suppose that $U_{r_i,x} = U_{r_j,y}$; hence, $U_{r_i,x}$ is adjacent to both r_i and r_j with $i \neq j$. As any two nodes lying in distinct groups in T are adjacent to a unique block of T, we must have that $U_{r_i,x} = U_{r_j,y} = U$; but this yields a contradiction as above. Hence, the blocks generated by the pairs in P are all distinct and all different from U. □

We use this lemma throughout, both explicitly and implicitly.

Our main result in the one-to-one context is concerned with building as many pairwise internally-disjoint paths as we can from any block to any other block in the bipartite graph built using the 2-step method (or, equivalently, from any node to any other node in the bipartite graph built using the 3-step method). We explain the impact of the existence of these paths on the path diversity of subsequently built DCNs presently. One added and significant complication in the proof of the following result comes about when the transversal design T is a $[k+1, k]$-transversal design (so, there is the potential for $\Delta = k+1 > k$ paths).

Theorem 4. *Let $k, \Delta, d \geq 2$. Let H be built by the 2-step method from the (d, Δ)-bipartite graph H_0 using the $[\Delta, k]$-transversal design T.*

(a) *Let Q and Q' be distinct blocks of H_0 so that there are $\lambda \geq 1$ pairwise internally-disjoint paths in H_0 from Q to Q', each of length at most μ. There are $\min\{\Delta, k\}$ pairwise internally-disjoint paths from any block $B_{Q,V}$ of H to any other block $B_{Q',V'}$ of H. Furthermore, if $\lambda \geq 2$ then there are Δ pairwise internally-disjoint paths from any block $B_{Q,V}$ of H to any other block $B_{Q',V'}$ of H. All paths have length at most $\mu + 4$.*

(b) *If $B_{Q,V}$ and $B_{Q,V'}$ are distinct blocks of H then there are Δ pairwise internally-disjoint paths from $B_{Q,V}$ to $B_{Q,V'}$, each of length at most 6 and lying entirely within T_Q.*

Proof. Recall that we mentioned in Section 2.6 that necessarily $\Delta \leq k+1$.

Case (a)(i): Suppose that: $\Delta = k+1$; $\lambda \geq 2$; and the distinct nodes p_1 and p_2 are common neighbours in H_0 of Q and Q'.

We 'batch' the groups of nodes of T_Q and $T_{Q'}$ together so that in each of T_Q and $T_{Q'}$, the $k+1$ groups of nodes form 1 batch of k groups and 1 batch of 1 group as follows:

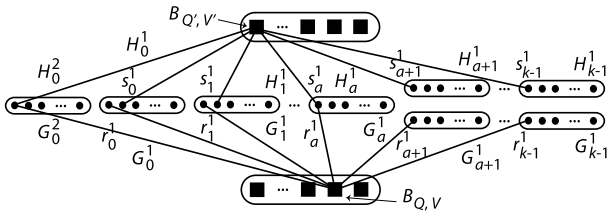

Fig. 7. The basic set-up in Case $(a)(i)$.

- for $i \in \{1,2\}$, define $G_0^i = G_{p_i} = H_0^i$
- the remaining $k-1$ groups within T_Q are $G_1^1, G_2^1, \ldots, G_{k-1}^1$ and the remaining $k-1$ groups within $T_{Q'}$ are H_1^1, $H_2^1, \ldots, H_{k-1}^1$ so that:
  - any group of the form G_j^1, where $j > 0$, is associated with some node $p \notin \{p_1, p_2\}$ of H_0 that is adjacent to both Q and Q' if, and only if, the group H_j^1 is associated with the same node p of H_0 (so, if G_j^1 and H_j^1 are associated with the same node $p \notin \{p_1, p_2\}$ of H_0 then they are the same group in H).

For each $j \in \{0, 1, \ldots, k-1\}$, let $r_j^1 \in G_j^1$ (resp. $s_j^1 \in H_j^1$) be the unique node of G_j^1 (resp. H_j^1) that is adjacent to $B_{Q,V}$ (resp. $B_{Q',V'}$) in H. Note that the pair r_j^1 and s_j^1 lie in the same group of nodes in H if, and only if, both G_j^1 and H_j^1 are associated with the same node p of H_0 and this node p is adjacent to both Q and Q' in H_0. The situation can be visualized as in Fig. 7 (where in this case Q and Q' have $a+2 \geq 2$ common neighbours in H_0 and where, for example, $r_1^1 \neq s_1^1$ but $r_a^1 = s_a^1$).

Let $G_0^1 = \{r_0^1, t_1, \ldots, t_{k-1}\}$ and $H_0^1 = \{s_0^1, w_1, \ldots, w_{k-1}\}$ so that:

- if $r_0^1 = s_0^1$ then $t_j = w_j$, for $j = 1, 2, \ldots, k-1$
- if $r_0^1 \neq s_0^1$ then $r_0^1 = w_1$, $s_0^1 = t_1$ and $t_j = w_j$, for $j = 2, 3, \ldots, k-1$.

We are now ready to generate some blocks within T_Q and $T_{Q'}$ in H. For each $j \in \{1, 2, \ldots, k-1\}$:

- let $B_{r_j^1, t_j}$ be the unique block of T_Q in H generated by the nodes $r_j^1 \in G_j^1$ and $t_j \in G_0^1$
- let $B'_{s_j^1, w_j}$ be the unique block of $T_{Q'}$ in H generated by the nodes $s_j^1 \in H_j^1$ and $w_j \in H_0^1$.

So, we have generated $k-1$ blocks in T_Q and $k-1$ blocks in $T_{Q'}$. Note that any block of T_Q is necessarily distinct from any block of $T_{Q'}$. By Lemma 3 applied twice to both T_Q and $T_{Q'}$, all blocks of $\{B_{r_j^1, t_j} : j = 1, 2, \ldots, k-1\}$ are distinct and different from $B_{Q,V}$, and all blocks of $\{B'_{s_j^1, w_j} : j = 1, 2, \ldots, k-1\}$ are distinct and different from $B_{Q',V'}$. Call these two sets of blocks our working sets of blocks.

We are now in a position to build some paths from $B_{Q,V}$ to $B_{Q',V'}$ in H. If $r_0^1 = s_0^1$ then define the paths:

- π_0^1 as $B_{Q,V}, r_0^1, B_{Q',V'}$
- π_1^1 as $B_{Q,V}, r_1^1, B_{Q',V'}$, if $r_1^1 = s_1^1$, and as $B_{Q,V}, r_1^1, B_{r_1^1,t_1}, t_1, B'_{s_1^1,w_1}, s_1^1, B_{Q',V'}$, if $r_1^1 \neq s_1^1$ (note that $t_1 = w_1$).

If $r_0^1 \neq s_0^1$ then define the paths:

- π_0^1 as $B_{Q,V}, r_0^1, B'_{s_1^1,w_1}, s_1^1, B_{Q',V'}$ (note that $w_1 = r_0^1$)
- π_1^1 as $B_{Q,V}, r_1^1, B_{r_1^1,t_1}, s_0^1, B_{Q',V'}$ (note that $t_1 = s_0^1$).

We'll now build paths from $B_{Q,V}$ to $B_{Q',V'}$ using nodes from the groups $\{G_0^1\} \cup \{G_j^1, H_j^1 : j = 2, 3, \ldots, k-1\}$. For each $j \in \{2, 3, \ldots, k-1\}$:

- if $r_j^1 \neq s_j^1$ then define the path:
  - π_j^1 as $B_{Q,V}, r_j^1, B_{r_j^1,t_j}, t_j, B'_{s_j^1,w_j}, s_j^1, B_{Q',V'}$ (note that $t_j = w_j$)
- if $r_j^1 = s_j^1$ then define the path:
  - π_j^1 as $B_{Q,V}, r_j^1, B_{Q',V'}$.

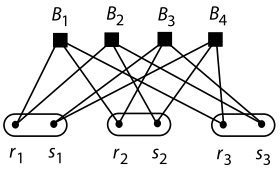

Fig. 8. The unique [3, 2]-transversal design.

Note that out of all of the k 'π-paths' constructed above, the only way that we can have that two of our paths are not internally-disjoint is when $r_0^1 \neq s_0^1$ but $r_1^1 = s_1^1$ (in which case π_0^1 and π_1^1 share the common node $r_1^1 = s_1^1$). In this case, choose $x_1 \in G_1^1 \setminus \{r_1^1\}$. Let $B_{r_0^1,x_1}$ be the block of T_Q in H generated by $r_0^1 (= w_1) \in G_0^1$ and $x_1 \in G_1^1$, and let $B'_{s_0^1,x_1}$ be the block of $T_{Q'}$ in H generated by $s_0^1 (= t_1) \in G_0^1$ and $x_1 \in G_1^1$ (in essence, we have dispensed with the blocks $B_{r_1^1,t_1}$ and $B'_{s_1^1,w_1}$ and replaced them with the blocks $B_{r_0^1,x_1}$ and $B'_{s_0^1,x_1}$ in our working sets of blocks). The conditions of Lemma 3 still hold and so the blocks in our working sets of blocks are all distinct and different from $B_{Q,V}$ and $B_{Q',V'}$. Redefine the paths:

- π_0^1 as $B_{Q,V}, r_1^1, B_{Q',V'}$
- π_1^1 as $B_{Q,V}, r_0^1, B_{r_0^1,x_1}, x_1, B'_{s_0^1,x_1}, s_0^1, B_{Q',V'}$.

The paths from the resulting set of k π-paths are now pairwise internally-disjoint and each has length at most 6.

Let r_0^2 (resp. s_0^2) be the unique node of G_0^2 (resp. H_0^2) that is adjacent to $B_{Q,V}$ (resp. $B_{Q',V'}$) in H. Suppose that $r_0^2 = s_0^2$. In this case, we build the path π_1^2 defined as $B_{Q,V}, r_0^2, B_{Q',V'}$. This path is clearly internally-disjoint from all of the k π-paths constructed above. Alternatively, suppose that $r_0^2 \neq s_0^2$. If $k \geq 3$ then there is a node $x_1^1 \in G_1^1 \setminus \{r_1^1, s_1^1\}$. Let $B_{r_0^2,x_1^1}$ be the block of T_Q within H generated by r_0^2 and x_1^1, and let $B_{s_0^2,x_1^1}$ be the block of $T_{Q'}$ within H generated by s_0^2 and x_1^1. By Lemma 3, these blocks are different from $B_{Q,V}$, $B_{Q',V'}$ and all other blocks used within the k π-paths constructed above (even when we make the amendments to our working sets of blocks as detailed in the preceding paragraph). Define the path π_0^2 as $B_{Q,V}, r_0^2, B_{r_0^2,x_1^1}, x_1^1, B_{s_0^2,x_1^1}, s_0^2, B_{Q',V'}$. This path has length 6 and is clearly internally-disjoint from all of the k π-paths constructed above.

On the other hand, suppose that $k = 2$; so, $\Delta = 3$. In particular, a [3, 2]-transversal design exists. We deal with this case from scratch.

Lemma 5. *There is exactly one* [3, 2]*-transversal design up to isomorphism and this is the transversal design depicted in Fig. 8.*

Proof. In some [3, 2]-transversal design, let the set of blocks be $\{B_1, B_2, B_3, B_4\}$ and let the group of nodes G_i be $\{r_i, s_i\}$, for $i = 1, 2, 3$. W.l.o.g., we must have the set of edges

$$\{(r_1, B_1), (r_1, B_2), (s_1, B_3), (s_1, B_4), (r_2, B_1), (r_2, B_3), (s_2, B_2), (s_2, B_4)\}.$$

W.l.o.g., the node B_4 is adjacent to r_3, and r_3 is adjacent to one other block. The only possible block that r_3 can be adjacent to is B_1 (as otherwise we would have two nodes in different groups adjacent to 2 distinct blocks). □

Name the blocks and nodes of T_Q as in Fig. 8. W.l.o.g. suppose that $B_{Q,V} = B_1$ (it is easy to see that there is an automorphism of T_Q mapping any block to any other block). There are two cases to consider: when Q and Q' have 3 common neighbours in H_0; and when they have only 2 common neighbours in H_0. However, before we deal with these cases, choose any 3 distinct nodes in T_Q. A tedious case-by-case analysis yields that no matter which 3 nodes are chosen, there are 3 pairwise internally-disjoint paths from B_1 to the 3 nodes within T_Q. For example, suppose that the 3 chosen nodes are r_1, s_2 and s_3. The 3 paths are: B_1, r_1; B_1, r_3, B_4, s_2; and B_1, r_2, B_3, s_3. It turns out that if the 3 chosen nodes are in 3 different groups then the length of any path is at most 5, whereas if the 3 chosen nodes are in 2 different groups then the length of any path is at most 3.

Suppose that Q and Q' have 3 common neighbours in H_0. Choose the 3 nodes in T_Q as the neighbours of $B_{Q',V'}$ in $T_{Q'}$. Consequently, from above, we clearly obtain 3 pairwise internally-disjoint paths from $B_{Q,V}$ to $B_{Q',V'}$ as required. Moreover, each path has length at most 6.

Suppose that Q and Q' have only 2 common neighbours in H_0 where the groups corresponding to these neighbours are $\{r_i, s_i\}$ and $\{r_j, s_j\}$, with $i, j \in \{1, 2, 3\}$, $i \neq j$. Choose 3 nodes in $T_{Q'}$ as the two neighbours of $B_{Q',V'}$ in $\{r_i, s_i\}$ and $\{r_j, s_j\}$, call them x_i and x_j, plus one other node, call it x, say, from one of these groups, with the remaining unchosen node from these two groups being denoted by y. By above, there are 3 pairwise internally-disjoint paths, π'_1, π'_2 and π'_3, in $T_{Q'}$ from $B_{Q',V'}$ to x_i, x_j and x, and we may assume that these paths do not involve y; for if one does then it must be the path to x, in which case we simply choose y as our third chosen node, above, instead of x (it is not difficult to see that the path to x has length at most 3). By above, there are also 3 pairwise internally-disjoint paths, π_1, π_2 and π_3, in T_Q from B_1 to x_i, x_j

and x, each of length at most 3; moreover, these paths do not share any nodes with the paths π'_1, π'_2 and π'_3 apart from the end-nodes. Consequently, we clearly obtain 3 pairwise internally-disjoint paths from $B_{Q,V}$ to $B_{Q',V'}$ as required. Moreover, each path has length at most 6.

Case $(a)(ii)$: Suppose that: $\Delta = k+1$; $\lambda \geq 2$; and there is exactly one common neighbour in H_0 of Q and Q', namely the node p_1.

As $\lambda \geq 2$, there is a path in H_0 of the form $Q, q_1, Q_1, q_2, Q_2, \ldots, Q_{m-1}, q_m, Q'$ where $m \leq \frac{\mu}{2}$ and where p_1 does not appear on this path (note that $q_1 \neq q_m$). We 'batch' our groups similarly to as we did before:

- define $G^1_0 = G_{p_1} = H^1_0$, $G^2_0 = G_{q_1}$ and $H^2_0 = G_{q_m}$
- the remaining $k-1$ groups within T_Q are $G^1_1, G^1_2, \ldots, G^1_{k-1}$ and the remaining $k-1$ groups in $T_{Q'}$ are $H^1_1, H^1_2, \ldots, H^1_{k-1}$.

Note that we necessarily have that the groups G^1_j and H^1_j are distinct, for $j \in \{1, 2, \ldots, k-1\}$ (as are the groups G^2_0 and H^2_0).

For each $j \in \{0, 1, \ldots, k-1\}$, let $r^1_j \in G^1_j$ (resp. $s^1_j \in H^1_j$) be the unique node of G^1_j (resp. H^1_j) that is adjacent to $B_{Q,V}$ (resp. $B_{Q',V'}$) in H. Also, let $r^2_0 \in G^2_0$ (resp. $s^2_0 \in H^2_0$) be the unique node of G^2_0 (resp. H^2_0) that is adjacent to $B_{Q,V}$ (resp. $B_{Q',V'}$) in H.

We construct the paths π^1_j, for $j = 0, 1, \ldots, k-1$, exactly as we did in Case $(a)(i)$. In addition, define the paths η^2_0 as $B_{Q,V}, r^2_0$ and ν^2_0 as $B_{Q',V'}, s^2_0$. If we can find a path in H from r^2_0 to s^2_0 so that no node or block of this path, apart from the nodes and blocks of η^2_0 and ν^2_0, lies in T_Q or $T_{Q'}$ then we are done. In H: there is a path of length 2 lying entirely within T_{Q_1} so that the source is r^2_0 and the destination is some node $y_2 \in G_{q_2}$; there is a path of length 2 lying entirely within T_{Q_2} so that the source is $y_2 \in G_{q_2}$ and the destination is some node $y_3 \in G_{q_3}$; …; there is a path of length 2 lying entirely within $T_{Q_{m-1}}$ so that the source is $y_{m-1} \in G_{q_{m-1}}$ and the destination is $s^2_0 \in H^2_0$. We clearly have a required path of length at most μ. So, we have constructed $\Delta = k+1$ pairwise internally-disjoint paths from $B_{Q,V}$ to $B_{Q',V'}$ so that k of these paths have length at most 6 and the remaining path has length at most μ.

Case $(a)(iii)$: Suppose that: $\Delta = k+1$; $\lambda \geq 2$; and there are no common neighbours in H_0 of Q and Q'.

As $\lambda \geq 2$, there are paths in H_0 of the form $Q, q_1, Q_1, q_2, Q_2, \ldots, Q_{a-1}, q_a, Q'$ and $Q, p_1, P_1, p_2, P_2, \ldots, P_{b-1}, p_b, Q'$ where $a, b \leq \frac{\mu}{2}$ and where these paths are internally-disjoint.

We 'batch' our groups similarly to as we did before:

- define $G^1_0 = G_{q_1}$ and $H^1_0 = G_{q_a}$
- choose $k-1$ groups within T_Q (different from G^1_0) as $G^1_1, G^1_2, \ldots, G^1_{k-1}$ and choose $k-1$ groups in $T_{Q'}$ (different from H^1_0) as $H^1_1, H^1_2, \ldots, H^1_{k-1}$.

Note that we necessarily have that the groups G^1_j and H^1_j are distinct, for $j \in \{0, 1, \ldots, k-1\}$.

For each $j \in \{0, 1, \ldots, k-1\}$, let $r^1_j \in G^1_j$ (resp. $s^1_j \in H^1_j$) be the unique node of G^1_j (resp. H^1_j) that is adjacent to $B_{Q,V}$ (resp. $B_{Q',V'}$) in H. Let $G^1_0 = \{r^1_0, t_1, \ldots, t_{k-1}\}$ and $H^1_0 = \{s^1_0, w_1, \ldots, w_{k-1}\}$. For each $j \in \{1, 2, \ldots, k-1\}$, let $B_{r^1_j, t_j}$ be the block of T_Q generated by $r^1_j \in G^1_j$ and $t_j \in G^1_0$, and let $B'_{s^1_j, w_j}$ be the block of $T_{Q'}$ generated by $s^1_j \in H^1_j$ and $w_j \in H^1_0$. By Lemma 3 applied twice to both T_Q and $T_{Q'}$, all blocks of $\{B_{r^1_j, t_j} : j = 1, 2, \ldots, k-1\}$ are distinct and different from $B_{Q,V}$, and all blocks of $\{B'_{s^1_j, w_j} : j = 1, 2, \ldots, k-1\}$ are distinct and different from $B_{Q',V'}$.

For $j \in \{1, 2, \ldots, k-1\}$, let η^1_j be the path $B_{Q,V}, r^1_j, B_{r^1_j, t_j}, t_j$ and let ν^1_j be the path $B_{Q',V'}, s^1_j, B'_{s^1_j, w_j}, w_j$. Define the path η^1_0 as $B_{Q,V}, r^1_0$ and the path ν^1_0 as $B_{Q',V'}, s^1_0$. In H: there are k paths of length 2 from the nodes $r^1_0, t_1, \ldots, t_{k-1}$ of G_{q_1} to distinct nodes $y^2_0, y^2_1, \ldots, y^2_{k-1}$ of G_{q_2}, respectively, so that all blocks on these paths lie in T_{Q_1} and are distinct; there are k paths of length 2 from the nodes $y^2_0, y^2_1, \ldots, y^2_{k-1}$ of G_{q_2} to distinct nodes $y^3_0, y^3_1, \ldots, y^3_{k-1}$ of G_{q_3}, respectively, so that all blocks on these paths lie in T_{Q_2} and are distinct; …; and there are k paths of length 2 from $y^{a-1}_0, y^{a-1}_1, \ldots, y^{a-1}_{k-1}$ to the nodes $s^1_0, w_2, \ldots, w_{k-1}$ of G_{q_a}, respectively, so that all blocks on these paths lie in $T_{Q_{m-1}}$ and are distinct. We can clearly piece all of the paths together to obtain k pairwise internally-disjoint paths from $B_{Q,V}$ to $B_{Q',V'}$ so that each path has length at most $\mu + 4$.

We can build another path from $B_{Q,V}$ to $B_{Q',V'}$ that is internally-disjoint from the k paths just constructed by proceeding exactly as we did above or in Case $(a)(ii)$, corresponding to the alternative path from Q to Q' in H_0. This path has length at most μ.

Case $(a)(iv)$: Suppose that $\lambda = 1$ or $\Delta \leq k$.

By choosing the appropriate construction from the cases above, depending upon whether there is a common neighbour of Q and Q' in H_0, we can clearly construct $\min\{\Delta, k\}$ pairwise internally-disjoint paths from $B_{Q,V}$ to $B_{Q',V'}$ so that: if

there is a common neighbour of Q and Q' in H_0, all paths have length at most 6; and if there is no common neighbour of Q and Q' in H, all paths have length at most $\mu + 4$.

Case (b): Consider the case when our two blocks are $B_{Q,V}$ and $B_{Q,V'}$. Suppose that the block Q of H_0 is adjacent to the nodes $p_1, p_2, \ldots, p_\Delta$. For each $i \in \{1, 2, \ldots, \Delta\}$, let $r_i \in G_{p_i}$ be adjacent to $B_{Q,V}$ in H and let $s_i \in G_{p_i}$ be adjacent to $B_{Q,V'}$ in H. W.l.o.g. suppose that $r_i \neq s_i$, for $i = 1, 2, \ldots, b$, and that $r_i = s_i$, for $i = b+1, b+2, \ldots, \Delta$.

Suppose that $b \geq 2$. For each $i \in \{1, 2, \ldots, b-1\}$, let $B_{r_i,s_{i+1}}$ be the block of T_Q that is generated by r_i and s_{i+1}, and let B_{r_b,s_1} be the block of T_Q that is generated by r_b and s_1. By Lemma 3, all blocks $B_{r_1,s_2}, B_{r_2,s_3}, \ldots, B_{r_{b-1},s_b}, B_{r_b,s_1}$ are distinct and different from $B_{Q,V}$ and $B_{Q,V'}$. Hence: if π_i is the path $B_{Q,V}, r_i, B_{r_i,s_{i+1}}, s_{i+1}, B_{Q,V'}$, for $i \in \{1, 2, \ldots, b-1\}$; if π_b is the path $B_{Q,V}, r_b, B_{r_b,s_1}, s_1, B_{Q,V'}$; and if π_i is the path $B_{Q,V}, r_i, B_{Q,V'}$, for $i \in \{b+1, b+2, \ldots, \Delta\}$, then paths in the resulting set are pairwise internally-disjoint, with each path having length at most 4.

If $b = 0$ then the above construction trivially yields Δ paths of length 2 from $B_{Q,V}$ to $B_{Q,V'}$. Suppose that $b = 1$. Choose $x_2 \in G_{p_2} \setminus \{r_2\}$ and let B_{r_1,x_2} (resp. B_{s_1,x_2}) be the block of T_Q generated by r_1 and x_2 (resp. s_1 and x_2). Clearly, B_{r_1,x_2}, B_{s_1,x_2}, $B_{Q,V}$ and $B_{Q,V'}$ are all distinct. So, if π_1 is the path $B_{Q,V}, r_1, B_{r_1,x_2}, x_2, B_{s_1,x_2}, s_1, B_{Q,V'}$ and π_i is the path $B_{Q,V}, r_i, B_{Q,V'}$, for $i \in \{2, 3, \ldots, \Delta\}$, then we obtain Δ pairwise internally-disjoint paths, with all paths having length 2 except one which has length 6. □

Theorem 4 is clearly optimal in the sense that the maximal number of pairwise internally-disjoint paths is always constructed (this follows from a simple application of Menger's Theorem). Also, irrespective of the erroneous proofs in [10], Theorem 4(b) extends the claimed results in [10] by deriving Δ pairwise internally-disjoint paths from any block $B_{Q,V}$ in H to any block $B_{Q,V'}$ (this scenario was not dealt with in [10]). Note also that the chance to obtain more than $\min\{\Delta, k\}$ pairwise internally-disjoint paths comes about when we force $\Delta = k + 1$ and choose a $[k+1, k]$-transversal design (if one exists).

Of course, Theorem 4 yields path diversity in any DCN constructed using the 3-step method with Methods A and B. Suppose that Method A has been used to construct a DCN where the number of server-nodes adjacent to some level-1 switch-node is at most the number of level-2 switch-nodes adjacent to the level-1 switch-node. If all level-1 switch-nodes are non-blocking then we can simultaneously facilitate data transfers from all the server-nodes adjacent to some level-1 switch-node to all the server-nodes adjacent to any other level-1 switch-node (in fact, we need only that the source and destination level-1 switch-nodes are non-blocking; all other level-1 switch-nodes can be blocking).

4.3. Applying our construction

In this section, we apply Theorem 4 and provide some concrete illustrations of how we can obtain switch-centric DCNs that have the same diameter as Fat-Tree yet have more server-nodes and significant one-to-one path diversity.

The primary difficulty in the proof of Theorem 4 is in dealing with when the $[\Delta, k]$-transversal design is such that $\Delta = k + 1$ (recall, $k, \Delta, d \geq 2$). However, dealing with this difficulty is worth it as having the capability to use $[k+1, k]$-transversal designs when applying the construction means that we obtain more flexibility as to the number of switch ports necessarily required in the resulting DCNs, as we illustrate now. In what follows, we limit ourselves (on the grounds of practicality) to switch-nodes with at most 128 ports. If we were only to use $[\Delta, k]$-transversal designs where $(\Delta, k) \in \{(3,3), (4,4), (5,5), (7,7), (8,8), (9,9), (11,11)\}$ (note that each of these $[\Delta, k]$-transversal designs exists; see Section 2.6) in the (one-iteration) 3-step method then (assuming that we use bipartite graphs H_0 that have the same number of nodes as blocks; that is, for which $d = \Delta$) we need level-2 switch-nodes with ($\Delta k =$) 9, 16, 25, 36, 49, 64, 81, 100 or 121 ports (cf. the parameter values given immediately after the definition of the 3-step construction in Section 3.1). If we allow $[\Delta, k]$-transversal designs where $(\Delta, k) \in \{(3,2), (4,3), (5,4), (6,5), (8,7), (9,8), (10,9)\}$ (again, note that each of these $[\Delta, k]$-transversal designs exists; see Section 2.6) then we have added flexibility in that we can also build DCNs with level-2 switch-nodes with 6, 12, 20, 30, 56, 72 or 90 ports; of course, to ensure that we obtain full path diversity, we need that H_0 has at least 2 internally-disjoint paths joining any two distinct blocks.

As regards finding large, regular, uniform bipartite graphs of line-diameter 4 and so that there are at least 2 internally-disjoint paths joining any two distinct blocks, this is not as straightforward as it is if we drop the second stipulation. There is an extensive literature as regards the construction of regular, uniform bipartite graphs of a given degree and where the degree is equal to the rank (see, for example, [20]) but in so far as we are aware, the construction of such graphs with any added stipulations (relating to connectivity, for example) has not been considered. Nevertheless, there are simple constructions that enable us to apply Theorem 4 to the full, as we now illustrate.

From [20], there is a regular, uniform bipartite graph of degree and rank 7 with 173 nodes and 173 blocks, and which has graph-theoretic diameter 4. Enumerate the nodes as $n_1, n_2, \ldots, n_{173}$ and the blocks as $b_1, b_2, \ldots, b_{173}$. Take two disjoint copies of this graph and add 346 edges joining n_i in one graph to b_i in the other graph; moreover, the nodes (resp. blocks) of the new bipartite graph are exactly the nodes (resp. blocks) of the original disjoint copies. The resulting graph is a regular, uniform bipartite graph of degree and rank 8 with graph-theoretic diameter at most 5; furthermore, there are clearly at least 2 internally-disjoint paths joining any pair of distinct blocks or any pair of distinct nodes where these paths have length at most 6. Take this bipartite graph as H_0. The construction of H_0 can be visualized as in Fig. 9, where we have also illustrated, using dotted lines, a pair of internally-disjoint paths between a pair of blocks.

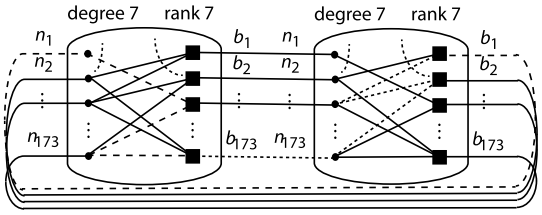

Fig. 9. Our regular, uniform bipartite graph construction.

Apply the 3-step method to H_0 using a $[8,7]$-transversal design. In the notation of Section 3.1, we have: the number of nodes, n, in H_0 equal to 346; the number of blocks, e, in H_0 equal to 346; $\Delta = d = 8$; and $k = 7$. This results in a bipartite graph H with $(ek^2 =)$ 16,954 nodes, $(nk =)$ 2,442 blocks, degree $(\Delta =)$ 8, and rank $(dk =)$ 56. Now we apply Method A with $c = 4$ to H; so, with reference to the parameters of H in the construction of Method A (cf. Section 3.3), we now have $n = 16{,}954$, $e = 2{,}442$, $\Delta = 8$, and $\delta = 56$. Consequently, we obtain a DCN of diameter 6 and with $(n(\delta - c\Delta) =)$ 406,896 server-nodes, $(n =)$ 16,954 level-1 switch-nodes, $(ce =)$ 9,768 level-2 switch-nodes, and so that all switch-nodes have $(\delta =)$ 56 ports. By Theorem 4, there are paths from the 24 server-nodes adjacent to the same level-1 switch-node X to the 24 server-nodes adjacent to another level-1 switch-node Y so that the only switch-nodes that lie on more than one of these paths are X and Y and so that the length of each of these paths is at most 10. In addition, we have spare capacity at the level-1 switch-nodes X and Y as 8 links to level-2 switch-nodes are not used.

Alternatively (for an increase in the number of server-nodes incorporated and in path diversity but so that more ports are required on switch-nodes), apply the 3-step method with H_0 using a $[8,8]$-transversal design. This results in a bipartite graph H with 22,144 nodes, 2,768 blocks, degree 8, and rank 64. Now apply Method A with $c = 4$ and we obtain a DCN of diameter 6 and with 708,608 server-nodes, 22,144 level-1 switch-nodes, 11,072 level-2 switch-nodes, and so that all switch-nodes have 64 ports. By Theorem 4, there are paths from the 32 server-nodes adjacent to the same level-1 switch-node X to the 32 server-nodes adjacent to another level-1 switch-node Y so that the only switch-nodes that lie on more than one of these paths are X and Y and so that the length of each of these paths is at most 10. The actual construction chosen will be dominated by the available hardware; that is, numbers of server-nodes and switch-nodes and the radix of switch-nodes.

Undertaking more iterations of the 2-step construction before building our DCNs yields that if we use $[\Delta, k]$-transversal designs where $(\Delta, k) \in \{(3,3), (4,4), (5,5), (7,7), (8,8), (9,9), (11,11)\}$ then we need level-2 switch-nodes with $(dk^2 =)$ 27, 64, 81 or 125 ports (cf. the parameter values pertaining to the iterated construction as specified in Section 3.2); and if we use $[\Delta, k]$-transversal designs where $(\Delta, k) \in \{(3,2), (4,3), (5,4), (6,5), (8,7), (9,8), (10,9)\}$ then we need level-2 switch-nodes with an alternative range of port numbers. As an illustration, iterating the 2-step method by mixing the use of $[3,3]$- and $[3,2]$-transversal designs, we can build DCNs where the level-2 switch-nodes need 6, 9, 12, 18, 24, 27, 36, 48, 54, 72, 81, 96 and 108 ports; for example, by iterating the 2-step construction twice with the $[3,2]$-transversal, we obtain a bipartite graph of degree 3 and rank 12, and a further application of the construction with the $[3,3]$-transversal yields a bipartite graph of degree 3 and rank 36. What is more, note that irrespective of the value of λ in relation to H_0 in Theorem 4, one application of the 2-step construction yields a bipartite graph with at least 2 internally-disjoint paths from any block to any other block; consequently, by Theorem 4, any bipartite graph formed by additional applications of the 2-step construction will necessarily have maximal path diversity.

It has already been established in [10] that the 2-step and 3-step methodologies are viable when it comes to building switch-centric DCNs that can host more server-nodes than a Fat-Tree and retain an acceptable level of (one-to-one) path diversity whilst maintaining a diameter of 6; we further cement this viability in this paper. An important point to note is that we need not choose our bipartite graph H_0 to be as large as we can; as we have shown, smaller bipartite graphs might yield DCNs with a sufficiently large number of server-nodes and optimal one-to-one path diversity.

5. One-to-many path diversity

We now work towards building Δ pairwise edge-disjoint paths from any block in some bipartite graph H built using the 2-step method to the blocks of any given multi-set of Δ blocks (so, there might possibly be repeated blocks; here, H and Δ are as in the statement of Theorem 4 but where $\Delta \leq k$). Henceforth, when we write 'set' we often mean 'multi-set'. We begin by working only within some transversal design.

Theorem 6. *Let T be any $[\Delta, k]$-transversal design where $k, \Delta \geq 2$ and where $\Delta \leq k$. Let U be any block and let $t_1, t_2, \ldots, t_\Delta$ be any Δ nodes or blocks, called target-nodes or target-blocks, as appropriate, where there may be repetitions. For each $i = 1, 2, \ldots, \Delta$, there is a path π_i from U to t_i of length at most 7 so that these paths are pairwise edge-disjoint.*

Proof. For each group of nodes D_j within T, where $j \in \{1, 2, \ldots, \Delta\}$, let r_j be the (unique) node of D_j adjacent to the block U; we call the nodes $r_1, r_2, \ldots, r_\Delta$ *root-nodes*. Consider some group D_j. There may be target-nodes that are identical

to the root-node r_j; call these target-nodes *rooted*, with the remaining target-nodes in D_j called *non-rooted*. Call the number of rooted target-nodes in D_j the *multiplicity* of the root-node r_j.

There are two essential cases: (*a*) we have Δ target-nodes and no target-blocks; (*b*) we have at least 1 target-block.

Case (*a*): Suppose that we have Δ target-nodes and no target-blocks.

We rank the groups of T as $D_{n_1}, D_{n_2}, \ldots, D_{n_\Delta}$ in decreasing order of the number of occurrences of non-rooted target nodes within the group, with ties broken according to decreasing multiplicity of the root-nodes (and then arbitrarily). We attempt to match the non-rooted target-nodes in D_{n_1} with the root-nodes $r_{n_2}, r_{n_3}, \ldots, r_{n_\Delta}$ in this order but only if the root-node has multiplicity 0 (that is, we skip over root-nodes of non-zero multiplicity; note that any skipped root-node is identical to at least 1 target-node). If we are successful then we attempt to match the non-rooted target-nodes in D_{n_2} by continuing down our list of root-nodes (again, skipping over root-nodes of non-zero multiplicity). If we are successful then we attempt to match the non-rooted target-nodes in D_{n_3}, and so on. There are three possibilities.

(*i*) We successfully match every non-rooted target-node without running out of root-nodes (of multiplicity 0). This happens when r_{n_1} has non-zero multiplicity or when there is a root-node with multiplicity at least 2.
(*ii*) We successfully match all but one of the non-rooted target-nodes and the final non-rooted target-node does not lie in D_{n_1}, in which case we match this target-node with r_{n_1}. This happens when r_{n_1} has multiplicity 0, every root-node has multiplicity at most 1, and there is a non-rooted target-node that does not lie in D_{n_1}.
(*iii*) We have one non-rooted target-node of D_{n_1} remaining to be matched and also the root-node r_{n_1} unmatched. This happens when all of the non-rooted target-nodes lie in D_{n_1} and r_{n_1} has multiplicity 0.

Consider Sub-case (*ii*). We extend our matching so that every root-node of multiplicity 1 is matched with the unique target-node that is identical to it. We have a complete matching of root-nodes to target-nodes so that no target-node is matched with the root-node in its own group unless the target-node is (the unique target-node) identical to the root-node. For every pair (r, t) where r is a root-node matched with a target-node t and so that r and t do not lie in the same group, let $U_{r,t}$ be the block generated by r and t. Call the resulting set of blocks the U-blocks. By Lemma 3, all of the U-blocks are distinct and different from U. If $U_{r,t}$ is a U-block then define the path π_r as $U, r, U_{r,t}, t$; and if the target-node t is identical to the root-node r then define the path π_r as U, t. The resulting Δ paths are pairwise internally-disjoint.

Consider Sub-case (*iii*). We have an almost complete matching of root-nodes to target-nodes so that no target-node is matched with the root-node in its own group, except that some target-node t' of D_{n_1} is not matched and nor is the root-node r_{n_1}. As we did above, we generate a set of U-blocks, one for each matched-pair. Again, these U-blocks are all distinct and different from U, and by proceeding as above we obtain $\Delta - 1$ pairwise internally-disjoint paths from U to target-nodes.

Consider t' and r_{n_1}. As $\Delta \geq 2$, there is some node x in the group D_{n_2} that is neither a root-node nor a target-node. Let $U'_{r_{n_1},x}$ (resp. $U'_{x,t'}$) be the block generated by r_{n_1} and x (resp. x and t'). By Lemma 3, $U'_{r_{n_1},x}$ is different from U and every U-block; also, $U'_{x,t'}$ is different from U and $U'_{r_{n_1},x}$. However, it could be that $U'_{x,t'}$ is identical to some U-block (for this to happen we would need that t' is identical to some other target-node). If t is the target-node of D_{n_1} matched with r_{n_2} then $U'_{x,t'}$ is different from $U_{r_{n_2},t}$. Hence, there are at most $\Delta - 2$ U-blocks with which $U'_{x,t'}$ might be identical. As we have at least $\Delta - 1$ choices for x in D_{n_2} (recall, $\Delta \leq k$), we can always choose x so that $U'_{x,t'}$ is different from every U-block. Define the path $\pi_{r_{n_1}}$ as $U, r_{n_1}, U'_{r_{n_1},x}, x, U'_{x,t'}, t'$. The resulting Δ paths from U to the target-nodes are pairwise internally-disjoint.

Consider Sub-case (*i*). We can extend our matching so that every root-node of non-zero multiplicity is matched with one target-node that is identical to it. Hence, we have a partial matching of root-nodes to target-nodes so that no target-node is matched with the root-node in its own group unless the target-node is identical to the root-node. As we did above, we generate a set of U-blocks, one for each matched-pair where the root-node in the pair is different from its matched target-node. Again, these U-blocks are all distinct and different from U, and we obtain pairwise internally-disjoint paths from U to all of the target-nodes involved. We also obtain paths of length 1 from U to every target-node that is identical to a root-node and has been matched with it. If there are no root-nodes of multiplicity greater than 1 then the resulting Δ paths are pairwise internally-disjoint and we are done. So, suppose that we have paths $\pi_1, \pi_2, \ldots, \pi_{\Delta-b}$ that are pairwise internally-disjoint and that there are $a \geq 1$ root-nodes of multiplicity at least 2 with b unmatched root-nodes (so, b is the number of target-nodes remaining to be dealt with; of course, $b \geq a$). Note that any group in which some hitherto unmatched root-node lies, apart from D_{n_1} if r_{n_1} is still unmatched (that is, has multiplicity 0), contains no target-nodes (because of the order in which we initially match target-nodes to root-nodes) and the groups containing unmatched root-nodes are either $D_{n_{\Delta-b+1}}, D_{n_{\Delta-b+2}}, \ldots, D_{n_\Delta}$, if r_{n_1} is matched, or $D_{n_1}, D_{n_{\Delta-b+2}}, D_{n_{\Delta-b+3}}, \ldots, D_{n_\Delta}$, if r_{n_1} is unmatched.

Suppose that $b = 1$; hence, there is exactly one root-node r_c, where $c \leq \Delta - 1$, of multiplicity greater than 1 and this multiplicity is 2. W.l.o.g. let the solitary target-node remaining to be dealt with be t_2 (which is identical to both r_c and some other target-node t_1), with the solitary root-node remaining to be dealt with being either r_{n_Δ} or r_{n_1}, as appropriate. If $\Delta = 2$ then we must have $\{r_{n_1}, x_{n_1}\} \subseteq D_{n_1}$ and $\{r_{n_2}, x_{n_2}\} \subseteq D_{n_2}$ with $x_{n_1} \neq r_{n_1}$ and $x_{n_2} \neq r_{n_2}$ so that the two target nodes t_1 and t_2 are both equal to r_{n_1} (note that in this case we define no U-blocks). Let $U'_{x_{n_2},r_{n_1}}$ (resp. $U'_{r_{n_2},x_{n_1}}$, $U'_{x_{n_1},x_{n_2}}$) be the block generated by x_{n_2} and r_{n_1} (resp. r_{n_2} and x_{n_1}, x_{n_1} and x_{n_2}). The blocks U, $U'_{x_{n_2},r_{n_1}}$, $U'_{r_{n_2},x_{n_1}}$ and $U'_{x_{n_1},x_{n_2}}$ are all distinct. Define

the path π_2 as $U, r_{n_2}, U'_{r_{n_2},x_{n_1}}, x_{n_1}, U'_{x_{n_1},x_{n_2}}, x_{n_2}, U'_{x_{n_2},r_{n_1}}, t_2$ and the path π_1 as U, t_1; the two paths are internally-disjoint and we are done.

Alternatively, suppose that $b = 1$ and $\Delta \geq 3$ (and so $k \geq 3$). If $c = n_{\Delta-1}$ then there is a non-rooted target-node in each D_j, for $j \in \{n_1, n_2, \ldots, n_{\Delta-2}\}$, with the unmatched root-node being r_{n_1}. Choose $x \in D_\Delta \setminus \{r_{n_\Delta}\}$. Otherwise, if $c \neq n_{\Delta-1}$ then $D_{n_{\Delta-1}}$ contains at most 1 target-node, which, if it exists, is rooted, with the unmatched root-node being either r_{n_1} or r_{n_Δ}. Choose $x \in D_{\Delta-1} \setminus \{r_{n_{\Delta-1}}\}$. Whichever is the case, let r be the unmatched root-node (and so $r \in \{r_{n_1}, r_{n_\Delta}\}$). Let U'_{x,t_2} (resp. $U'_{r,x}$) be the block generated by x and t_2 (resp. r and x). By Lemma 3, the U-blocks, U, U'_{x,t_2} and $U'_{r,x}$ are all distinct. Define the path π_Δ as $U, r, U'_{r,x}, x, U'_{x,t_2}, t_2$ so as to obtain Δ pairwise internally-disjoint paths from U to the target-nodes; hence, we are done.

Now suppose that $b \geq 2$ (note that $b \leq \Delta - 1 \leq k - 1$). As stated above, the root-nodes remaining to be dealt with are either $r_{n_{\Delta-b+1}}, r_{n_{\Delta-b+2}}, \ldots, r_{n_\Delta}$ or $r_{n_1}, r_{n_{\Delta-b+2}}, r_{n_{\Delta-b+3}}, \ldots, r_{n_\Delta}$. Suppose that the root-nodes remaining to be dealt with are $r_{n_{\Delta-b+1}}, r_{n_{\Delta-b+2}}, \ldots, r_{n_\Delta}$ and the target-nodes remaining to be dealt with are $t_1, t_2, \ldots, t_b$ (of course, every such target-node is identical to an already matched root-node). For each $i \in \{\Delta - b + 1, \Delta - b + 2, \ldots, \Delta\}$, let $D_{n_i} = \{r_{n_i}, x_2^{n_i}, x_3^{n_i}, \ldots, x_k^{n_i}\}$ and choose $x'_{n_i} \in D_{n_i} \setminus \{r_{n_i}\}$ (from our earlier remark, there are no target-nodes in D_{n_i}). For each $i \in \{\Delta - b + 1, \Delta - b + 2, \ldots, \Delta - 1\}$, let $U'_{r_{n_i},x'_{n_{i+1}}}$ be the block generated by r_{n_i} and $x'_{n_{i+1}}$, and let $U'_{r_{n_\Delta},x'_{n_{\Delta-b+1}}}$ be the block generated by r_{n_Δ} and $x'_{n_{\Delta-b+1}}$; call these blocks the U'-blocks. By Lemma 3, the U'-blocks are distinct and each U'-block is different from every U-block and U. For each $i \in \{\Delta - b + 1, \Delta - b + 2, \ldots, \Delta\}$, let $\bar{U}_{x'_{n_i},t_i}$ be the block generated by x'_{n_i} and t_i; call these blocks the $\bar{U}$-blocks. By Lemma 3, each $\bar{U}$-block is different from U, every U-block and from every U'-block (note that any t_i is a root-node and so not adjacent to any U-block or U'-block). However, it is possible that $\bar{U}_{x'_{n_i},t_i} = \bar{U}_{x'_{n_j},t_j}$, for $i \neq j$ (for this to happen we would need that $t_i = t_j$, as otherwise we would have two root-nodes adjacent to both U and another block). Note that for each $i \in \{\Delta - b + 1, \Delta - b + 2, \ldots, \Delta\}$: we have $k - 1$ possible choices within D_{n_i} for x'_{n_i}; and for $j_1, j_2 \in \{2, 3, \ldots, k\}$, where $j_1 \neq j_2$, the block $\bar{U}_{x_{j_1}^{n_i},t_i}$, generated by $x_{j_1}^{n_i}$ and t_i, is different from the block $\bar{U}_{x_{j_2}^{n_i},t_i}$, generated by $x_{j_2}^{n_i}$ and t_i.

Choose $x'_{n_{\Delta-b+1}} = x_2^{n_{\Delta-b+1}}$ and $x'_{n_{\Delta-b+2}} = x_2^{n_{\Delta-b+2}}$. Suppose we have that $\bar{U}_{x'_{n_{\Delta-b+2}},t_2} = \bar{U}_{x'_{n_{\Delta-b+1}},t_1}$; if so then re-choose $x'_{n_{\Delta-b+2}} = x_3^{n_{\Delta-b+2}}$. Necessarily, $\bar{U}_{x'_{n_{\Delta-b+2}},t_2} \neq \bar{U}_{x'_{n_{\Delta-b+1}},t_1}$. Choose $x'_{n_{\Delta-b+3}} = x_2^{n_{\Delta-b+3}}$. Suppose that $\bar{U}_{x'_{n_{\Delta-b+3}},t_3} = \bar{U}_{x'_{n_{\Delta-b+1}},t_1}$; if so then re-choose $x'_{n_{\Delta-b+3}} = x_3^{n_{\Delta-b+3}}$. Suppose that $\bar{U}_{x'_{n_{\Delta-b+3}},t_3} = \bar{U}_{x'_{n_{\Delta-b+2}},t_2}$; if so then re-choose $x'_{n_{\Delta-b+3}} = x_4^{n_{\Delta-b+3}}$. Necessarily, $\bar{U}_{x'_{n_{\Delta-b+1}},t_1}, \bar{U}_{x'_{n_{\Delta-b+2}},t_2}, \bar{U}_{x'_{n_{\Delta-b+3}},t_3}$ are distinct. Proceed in this way until $x'_{n_{\Delta-b+2}}, x'_{n_{\Delta-b+3}}, \ldots, x'_{n_\Delta}$ have been chosen. Note that as $b \leq \Delta - 1 \leq k - 1$, the above procedure can always be completed. What results is the set of distinct blocks $\{\bar{U}_{x'_{n_{\Delta-b+i}},t_i} : i = 1, 2, \ldots, b\}$. For each $i = 1, 2, \ldots, b - 1$, define the path $\pi_{\Delta-b+i}$ as $U, r_{n_{\Delta-b+i}}, U'_{r_{n_{\Delta-b+i}},x'_{n_{\Delta-b+i+1}}}, x'_{n_{\Delta-b+i+1}}, \bar{U}_{x'_{\Delta-b+i+1},t_i}, t_i, t_i$, and define the path π_Δ as $U, r_{n_\Delta}, U'_{r_{n_\Delta},x'_{n_{\Delta-b+1}}}, x'_{n_{\Delta-b+1}}, \bar{U}_{x'_{n_{\Delta-b+1}},t_b}, t_b, t_b$. The resulting Δ paths $\pi_1, \pi_2, \ldots, \pi_\Delta$ from U to the target-nodes are pairwise internally-disjoint.

Alternatively, suppose that the root-nodes remaining to be dealt with are $r_{n_1}, r_{n_{\Delta-b+2}}, r_{n_{\Delta-b+3}}, \ldots, r_{n_\Delta}$. We proceed exactly as above except that we start from a node $x'_{n_1} \in D_{n_1} \setminus \{r_{n_1}\}$ that is different from any target-node (such a node exists). We obtain our pairwise internally-disjoint paths as before.

Case (b): Suppose that there is at least 1 target-block.

W.l.o.g. we may assume that the a target-nodes $t_1, t_2, \ldots, t_a$, where $0 \leq a \leq \Delta - 1$, lie within the groups $D_1, D_2, \ldots, D_a$ and that the target-blocks are $U_1, U_2 \ldots, U_{\Delta-a}$. Suppose that some target-block U_i is adjacent to some root-node r_j of some group D_j, where $i \in \{1, 2, \ldots, \Delta - a\}$ and $j \in \{a + 1, a + 2, \ldots, \Delta\}$. Remove the target-block U_i (temporarily) from our set of targets and include the new target-node r_j. Iterate this process. Hence, w.l.o.g. we may assume that: our target-nodes are the original target-nodes $t_1, t_2, \ldots, t_a$ along with the new target-nodes $r_{a+1}, r_{a+2}, \ldots, r_{a+b}$, where each new target-node r_{a+i} is adjacent to the now removed old target-block U_i; and our target-blocks are $U_{b+1}, U_{b+2}, \ldots, U_{\Delta-a}$ with none of these target-blocks adjacent to any root-node in the groups $D_{a+b+1}, D_{a+b+2} \ldots, D_\Delta$. For each $i \in \{1, 2, \ldots, \Delta - a\}$: let the node $x_{a+b+i} \in D_{a+b+i} \setminus \{r_{a+b+i}\}$ be adjacent to U_{b+i}; and (temporarily) remove the target-block U_{b+i} and include the new target-node x_{a+b+i}.

Apply the construction in Case (a) to our new set of Δ target-nodes. We obtain Δ paths, one from U to each of our target-nodes so that these paths are internally-disjoint. Consider some new target-node r_{a+i}, where $i \in \{1, 2, \ldots, b\}$. By the construction of our paths, the path corresponding to this new target-node is U, r_{a+i} and r_{a+i} does not appear on any other path (there is no repetition of r_{a+i} in our set of target-nodes). Extend the path U, r_{a+i} to the path U, r_{a+i}, U_i, for $i = 1, 2, \ldots, b$. Consider some new target-node x_{a+b+i}, where $i \in \{1, 2, \ldots, \Delta - a\}$. Suppose that the edge (U_{b+i}, x_{a+b+i}) appears on some path. By the construction of our paths, the only way that this can happen is if this edge is the last edge on the path from U to x_{a+b+i}. If this is the case then truncate this path at U_{b+i}. Alternatively, if the edge (U_{b+i}, x_{a+b+i}) does not appear on some path then we extend the path from U to x_{a+b+i} by the addition of the edge to U_{b+i}. Consequently, we obtain a set of paths from U to each of our original target-nodes and target-blocks so that these paths are pairwise edge-disjoint. Note that: target-nodes only appear as destinations and apart from possibly target-nodes, no node appears on more than one path; and no block appears on more than one path except possibly for some target-blocks (which might appear as internal nodes on paths). The result follows. □

Note that the construction in Theorem 6 is weaker than those in the previous section as we obtain only that paths are pairwise edge-disjoint rather than pairwise internally-disjoint. However, we do obtain the following result as an immediate corollary of the construction in Theorem 6.

Corollary 7. *Let T be any $[\Delta, k]$-transversal design where $k, \Delta \geq 2$ and where $\Delta \leq k$. Let U be any block and let $t_1, t_2, \ldots, t_\Delta$ be any Δ nodes, called target-nodes, where there may be repetitions. For each $i = 1, 2, \ldots, \Delta$, there is a path π_i from U to t_i of length at most 7, so that the paths $\pi_1, \pi_2, \ldots, \pi_\Delta$ are pairwise internally-disjoint.*

We now build some many-to-many edge-disjoint paths within some transversal design.

Theorem 8. *Let T be any $[\Delta, k]$-transversal design where $\Delta \leq k$ and $k, \Delta \geq 2$. Let $a + b = \Delta_0 \leq \Delta$ where $a, b \geq 0$. Suppose that we are given a nodes, the target-nodes, and b blocks, the target-blocks, so that there might be repetitions amongst the target-nodes and target-blocks. Suppose that D_0 is a group of nodes that contains no target-nodes. There exists a set S of Δ_0 distinct nodes of D_0 such that there are Δ_0 pairwise internally-disjoint paths, each of length at most 3, the sources of which are the nodes of S and the destinations of which are all the target-nodes and target-blocks.*

Proof. Suppose that $b \geq 1$ (we'll deal with the case when $b = 0$ later) and suppose that the distinct target-blocks are $U_1, U_2, \ldots, U_c$, so that the target-blocks $U_{c+1}, U_{c+2}, \ldots, U_b$ all lie in $\{U_i : i = 1, 2, \ldots, c\}$. Furthermore, suppose that for each $i \in \{1, 2, \ldots, c\}$, the target-block U_i is repeated n_i times in the set of target-blocks. So, $b = \sum_{i=1}^{c} n_i$. We define that $U_i \equiv U_j$, for $i, j \in \{1, 2, \ldots, c\}$ if, and only if, U_i and U_j are adjacent to the same node of D_0. Let $U_1, U_2, \ldots, U_d$ (where $d \geq 1$) be representatives from the resulting equivalence classes (so, $d \leq c$) and let x_1^i be the node of D_0 adjacent to U_i, for $i = 1, 2, \ldots, d$. Thus, we immediately obtain d paths $\pi_1^1, \pi_1^2, \ldots, \pi_1^d$ of length 1 from distinct nodes of D_0 to the target-blocks $U_1, U_2, \ldots, U_d$.

For ease of notation, we rename some of the groups of nodes of T as $\{D_0\} \cup \{D_j^i : i = 1, 2, \ldots, d;\ j = 2, 3, \ldots, n_i\} \cup \{D_j^i : i = d+1, d+2, \ldots, c;\ j = 1, 2, \ldots, n_i\}$ so that no target-node lies in any of these groups (note that the number of such groups is $(\sum_{i=1}^{c} n_i) - d + 1 \leq b = \Delta_0 - a$ and so this is possible). For each $i \in \{1, 2, \ldots, d\}$ and each $j \in \{2, 3, \ldots, n_i\}$, choose $x_j^i \in D_0 \setminus \{x_1^1, x_1^2, \ldots, x_1^d\}$, and for each $i \in \{d+1, d+2, \ldots, c\}$ and each $j \in \{1, 2, \ldots, n_i\}$, choose $x_j^i \in D_0 \setminus \{x_1^1, x_1^2, \ldots, x_1^d\}$, so that all chosen nodes are distinct. For each $i \in \{1, 2, \ldots, d\}$ and each $j \in \{2, 3, \ldots, n_i\}$, let $r_j^i \in D_j^i$ be the unique node adjacent to U_i, and for each $i \in \{d+1, d+2, \ldots, c\}$ and each $j \in \{1, 2, \ldots, n_i\}$, let $r_j^i \in D_j^i$ be the unique node adjacent to U_i.

For each $i \in \{1, 2, \ldots, d\}$ and each $j \in \{2, 3, \ldots, n_i\}$, let U_j^i be the block generated by x_j^i and r_j^i, and for each $i \in \{d+1, d+2, \ldots, c\}$ and each $j \in \{1, 2, \ldots, n_i\}$, let U_j^i be the block generated by x_j^i and r_j^i. Call the resulting blocks generated the U-blocks. In particular, as every U-block is adjacent to a different node of D_0, all U-blocks are distinct. Moreover, as no target-block is adjacent to the same node of D_0 that any U-block is adjacent to, every U-block is different from every target-block. For each $i \in \{1, 2, \ldots, d\}$ and each $j \in \{2, 3, \ldots, n_i\}$, define the path π_j^i as x_j^i, U_j^i, r_j^i, U_i, and for each $i \in \{d+1, d+2, \ldots, c\}$ and each $j \in \{1, 2, \ldots, n_i\}$, define the path π_j^i as x_j^i, U_j^i, r_j^i, U_i. The paths from the set $\{\pi_j^i : i = 1, 2, \ldots, c;\ j = 1, 2, \ldots, n_i\}$ are clearly internally-disjoint.

Write $n_0 = a$ and suppose that the target-nodes are $t_1, t_2, \ldots, t_{n_0}$. Let $x_1^0, x_2^0, \ldots, x_{n_0}^0$ be distinct nodes of $D_0 \setminus \{x_j^i : i = 1, 2, \ldots, c;\ j = 1, 2, \ldots, n_i\}$. For each $j \in \{1, 2, \ldots, n_0\}$, let U_j^0 be the block generated by x_j^0 and t_j. As above, all such blocks are distinct and different from any block generated so far. For each $j \in \{1, 2, \ldots, n_0\}$, define the path π_j^0 as x_j^0, U_j^0, t_j. The resulting set of paths $\{\pi_j^i : i = 0, 1, \ldots, c;\ j = 1, 2, \ldots, n_i\}$ is as required.

Alternatively, if $b = 0$ then we dispense with the above construction of paths to target-blocks and proceed identically as regards the target-nodes. The result clearly follows. □

We are now in a position to use Theorems 6 and 8 to obtain the main result of this section.

Theorem 9. *Let $k, \Delta, d \geq 2$ so that $\Delta \leq k$. Let H be built by the 2-step method applied to the connected (d, Δ)-bipartite graph H_0 using the $[\Delta, k]$-transversal design T. Let B be some block of H and let $B_1, B_2, \ldots, B_\Delta$ be blocks of H that are not necessarily distinct but different from B. There exist paths from B to $B_1, B_2, \ldots, B_\Delta$ so that no edge of H appears in more than one of these paths.*

Proof. Let $Q_1, Q_2, \ldots, Q_q$ be the exact distinct blocks of H_0 such that $\cup_{i=1}^{q} T_{Q_i}$ contains the blocks $B_1, B_2, \ldots, B_\Delta$ within H (in particular, $q \leq \Delta$), and let Q_0 be the block of H_0 such that T_{Q_0} contains the block B within H. Let Z be a tree within H_0 that is rooted at Q_0 and is such that: every block of $\{Q_i : i = 1, 2, \ldots, q\}$ appears in Z; and all leaves of Z are blocks within $\{Q_i : i = 1, 2, \ldots, q\}$. We use the tree Z as a skeleton so as to build our required paths in H.

Call the blocks $B_1, B_2, \ldots, B_\Delta$ the *H-target-blocks*. Label every node p (resp. block Q) in Z with a non-negative integer $\mu(p)$ (resp. $\mu(Q)$) detailing the number of H-target-blocks that are associated with a block of Z that is a descendant of p

(resp. a descendent of Q or with Q itself). So, for example, the root Q_0 is such that $\mu(Q_0) = \Delta$ and any leaf (block) Q of Z is such that $\mu(Q)$ is the number of H-target-blocks within T_Q.

Suppose that p is some node of Z whose children are all leaves (and so blocks). Suppose that w.l.o.g. these children are $Q_1, Q_2, \ldots, Q_r$. For each $i \in \{1, 2, \ldots, r\}$, by Theorem 8, there exists a set S_i of $\mu(Q_i)$ nodes of the group of nodes of H associated with the node p of H_0 so that there are $\mu(Q_i)$ pairwise internally-disjoint paths from the nodes of S_i to the H-target-blocks associated with Q_i where each of these paths has length at most 3 (note that the edges of these paths lie in T_{Q_i}; of course, the edges of T_{Q_i} are disjoint from the edges of T_{Q_j}, for any $i \neq j$). Consequently, we obtain a multi-set $S_p = \cup_{i=1}^{r} S_i$ of $\mu(p)$ nodes in the group of nodes in H associated with the node p of H_0 so that there are $\mu(p)$ paths in $\cup_{i=1}^{r} T_{Q_i}$ from the nodes of S_p to the H-target-blocks associated with the blocks $Q_1, Q_2, \ldots, Q_r$. These $\mu(p)$ paths are pairwise internally-disjoint but they might have common sources.

Suppose that Q is some non-root block of Z whose children are w.l.o.g. $p_1, p_2, \ldots, p_r$ and so that the following holds:

- associated with each child p_i is a multi-set S_i of $\mu(p_i)$ nodes in the group of nodes of H associated with the node p_i of H_0
- for each child p_i, there are $\mu(p_i)$ paths from the nodes of S_i to the H-target-blocks associated with blocks that are descendants of p_i in T so that all of these paths have length at most l
- no edge of H appears in more than one of the $\sum_{i=1}^{r} \mu(p_i)$ paths that are associated with some child of Q.

Let p_0 be the node of Z that is the parent of Q. By Theorem 8, there is a set S_0 of $\mu(p_0)$ nodes in the group of nodes of H associated with p_0 together with $\mu(p_0)$ paths from the nodes of S_0 to the nodes of $\cup_{i=1}^{r} S_i$ in union with the H-target-blocks associated with Q where the paths are pairwise internally-disjoint and each path has length at most 3. Hence, by concatenating the paths involved, we have $\mu(p_0) = \mu(Q)$ paths from the nodes of S_0 to the H-target-blocks associated with all descendant blocks of p_0 in Z where no edge of H appears in more than one of these paths and the length of any of these paths is at most $l + 3$.

Finally, suppose that the children of Q_0 in Z are w.l.o.g. $p_1, p_2, \ldots, p_r$ and are such that the following holds:

- associated with each child p_i is a multi-set S_i of $\mu(p_i)$ nodes in the group of nodes of H associated with the node p_i of H_0
- for each child p_i, there are $\mu(p_i)$ paths from the nodes of S_i to the H-target-blocks associated with blocks that are descendants of p_i in T so that all of these paths have length at most l
- no edge of H appears in more than one of the $\sum_{i=1}^{r} \mu(p_i)$ paths that are associated with some child of Q_0.

By Theorem 6, we obtain paths from B to the nodes of $\cup_{i=1}^{r} S_i$ in union with the multi-set of blocks associated with Q_0 so that no edge of H appears in more than one of these paths and all paths have length at most 7. Consequently, by concatenating paths, we obtain Δ paths from B to $B_1, B_2, \ldots, B_\Delta$ so that no edge of H appears in more than one of these paths and the paths have length at most $l + 7$. The result follows by induction. Moreover, it is easy to see that if the depth of Z is h then the length of the longest of these paths is at most $3\frac{h}{2} + 7$. □

We have two remarks as regards Theorem 9: first, note the additional bound of $3\frac{h}{2} + 7$ on the lengths of the paths derived in the proof of Theorem 9 in terms of the height h of the tree Z; and, second, this theorem is weaker than Theorem 4 in that in Theorem 9 the paths constructed are pairwise edge-disjoint rather than pairwise internally-disjoint as they are in Theorem 4.

Of course, armed with the constructions of switch-centric DCNs from Section 3.3, and analogous to our constructions in Section 4.3, it should be clear how we can obtain pairwise edge-disjoint paths joining all the server-nodes adjacent to some level-1 switch-node in some appropriately constructed DCN to any identically-sized set of distinct server-nodes (irrespective of whether they are adjacent to different level-1 switch-nodes), so long as the number of server-nodes adjacent to some level-1 switch-node is no more than the number of level-2 switch-nodes adjacent to it. (Note also that whereas when we applied Theorem 4 in Section 4.3, complications arose because of the need to have λ at least 2, there are no such complications when it comes to applying the theorems in this section.)

6. Conclusion

In this paper, we have shown how combinatorial design theory can be used to build switch-centric DCNs of diameter at most 6 and with many more server-nodes than the Fat-Tree DCN but so that there is still considerable one-to-one and one-to-many path diversity. We regard the more general demonstration that combinatorial mathematics can enhance the design of modern-day computational infrastructures such as data centres as one of the primary contributions of this paper. Whilst we have demonstrated that combinatorial mathematics has the potential to add to and improve the design of DCNs, the DCNs obtained by our constructions need to be studied in much greater detail with regard to the numerous

other properties that a switch-centric DCN (and, indeed, any other type of DCN) has to have in order to make it viable as a practical DCN (as we hinted earlier, there is no one DCN, or even DCN paradigm, that will satisfy every property simultaneously). For example: although we bound the diameter of our DCNs, we need to derive (optimal) routing algorithms (within bipartite graphs built using the 2-step method) so as to meet these bounds; and (as was noted in [10]) it would be beneficial if the bisection width of the DCNs constructed in this paper could be ascertained (bisection width is often used as a proxy for throughput in DCNs). However, the list of desirable properties, alluded to in the Introduction, that we would wish for our DCNs is lengthy and will result in numerous new strands of research, both theoretical and more practical.

Our results also throw up some immediate directions for further research and we mention four such directions now.

It would be interesting to discover more mechanisms for converting bipartite graphs constructed using the 2-step method into DCNs than those developed in [10] and detailed in Section 3.3. We envisage that such a study would go hand-in-hand with research into building DCNs which possess yet more beneficial properties as regards their efficacy as DCNs (as highlighted above).

As we mention in Section 4.3, our constructions have drawn attention to a hitherto unstudied problem in combinatorics namely the construction of regular, uniform bipartite graphs with additional properties such as having at least 2 internally-disjoint paths joining any two blocks. It would be interesting to study problems such as this in a solely mathematical context.

Our results have hinted that the study of transversal designs as bipartite graphs and in a graph-theoretic context is worth pursuing. For example, if one looks at Theorem 4 then there are Δ pairwise internally-disjoint paths, each of length at most 6, joining any two distinct blocks in some transversal design T; and if one looks at Theorem 6 then, if $\Delta \leq k$, there are Δ pairwise edge-disjoint paths, each of length at most 6, joining any given source block with any given multi-set of Δ target blocks in some transversal design T. Such results might be of independent interest within some appropriate network context. Within a DCN N built using the 2-step method, there are many 'copies' of the bipartite graph corresponding to the chosen transversal design. These copies and the above constructions might be utilized where there is traffic localization, e.g., in a virtualization context where many guest DCNs are embedded within the DCN N.

Finally, as mentioned above, we need to derive routing algorithms within the DCNs constructed in this paper so as to make use of the available path diversity. The combinatorial first step to this is the 'control' of the 2-step and 3-step constructions, especially as one iterates the construction and then applies the constructions using Methods A and B. We need to be able to devise a combinatorial naming scheme, in terms of the constituent base bipartite graph H_0 and the transversal design T, so that we can use the resulting algebraic description as the framework for subsequent routing algorithms.

Acknowledgments

The author would like to thank the anonymous reviewers whose comments helped to improve this paper.

References

[1] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, N. McKeown, ElasticTree: saving energy in data center networks, in: Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation, NSDI'10, USENIX Association, Berkeley, CA, USA, 2010, pp. 17:1–17:16.

[2] A. Greenberg, J.R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D.A. Maltz, P. Patel, S. Sengupta, VL2: a scalable and flexible data center network, Comput. Commun. Rev. 39 (2009) 51–62.

[3] J.H. Ahn, N. Binkert, A. Davis, M. McLaren, R.S. Schreiber, HyperX: topology, routing, and packaging of efficient large-scale networks, in: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis, SC'09, ACM, New York, NY, USA, 2009, pp. 41:1–41:11.

[4] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, A. Vahdat, Portland: a scalable fault-tolerant layer 2 data center network fabric, Comput. Commun. Rev. 39 (2009) 39–50.

[5] D. Abts, M.R. Marty, P.M. Wells, P. Klausler, H. Liu, Energy proportional datacenter networks, in: Proceedings of the 37th Annual International Symposium on Computer Architecture, ISCA'10, ACM, New York, NY, USA, 2010, pp. 338–347.

[6] M. Al-Fares, A. Loukissas, A. Vahdat, A scalable, commodity data center network architecture, Comput. Commun. Rev. 38 (2008) 63–74.

[7] Y. Liu, J.K. Muppala, M. Veeraraghavan, D. Lin, M. Hamdi, Data Centre Networks: Topologies, Architectures and Fault-Tolerance Characteristics, Springer Briefs in Computer Science, Springer, 2013.

[8] K. Wu, J. Xiao, L.M. Ni, Rethinking the architecture design of data center networks, Frontiers of Computer Science 6 (2012) 596–603.

[9] T. Chen, X. Gao, G. Chen, The features, hardware, and architectures of data center networks: a survey, J. Parallel Distrib. Comput. 96 (2016) 45–74.

[10] G. Qu, Z. Fang, J. Zhang, S.-Q. Zheng, Switch-centric data center network structures based on hypergraphs and combinatorial block designs, IEEE Trans. Parallel Distrib. Syst. 26 (2015) 1154–1164.

[11] J. Bermond, J. Bond, S. Djelloul, Dense bus networks of diameter 2, in: D.F. Hsu, A.L. Rosenberg, D. Sotteau (Eds.), DIMACS Workshop on Interconnection Networks and Mapping and Scheduling Parallel Computations, in: DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 21, American Mathematical Society, 1995, pp. 9–18.

[12] C. Colbourn, J. Dinitz, D. Stinson, Applications of combinatorial designs to communications, cryptography and networking, in: J. Lamb, D. Preece (Eds.), Surveys in Combinatorics, 1999, in: London Mathematical Society Lecture Notes Series, vol. 267, Cambridge University Press, 1999, pp. 37–100.

[13] J. Luo, Y. Guo, S. Fu, K. Li, W. He, Virtual resource allocation based on link interference in Cayley wireless data centers, IEEE Trans. Comput. 64 (2015) 3016–3021.

[14] C. Camarero, E. Vallejo, R. Beivide, Topological characterization of hamming and dragonfly networks and its implications on routing, ACM Trans. Archit. Code Optim. 11 (2014) 39.

[15] D. Coudert, G. Ducoffe, Data center interconnection networks are not hyperbolic, Theor. Comput. Sci. 639 (2016) 72–90.
[16] X. Liu, N. Iftikhar, X. Xie, Survey of real-time processing systems for big data, in: Proceedings of the 18th International Database Engineering and Applications Symposium, IDEAS'14, ACM, New York, NY, USA, 2014, pp. 356–361.
[17] R. Diestel, Graph Theory, Graduate Texts in Mathematics, vol. 173, Springer, 2010.
[18] J. Dean, S. Ghemawat, MapReduce: simplified data processing on large clusters, Commun. ACM 51 (2008) 107–113.
[19] D.R. Stinson, Combinatorial Designs: Constructions and Analysis, Springer, 2004.
[20] M. Miller, J. Sirán, Moore graphs and beyond: a survey of the degree/diameter problem, Electron. J. Comb. 20 (2005), Dynamic Survey DS14.