Durham University

Durham Research Online

Deposited in DRO:

31 March 2016

Version of attached file:

Accepted Version

Peer-review status of attached file:

Peer-reviewed

Citation for published item:

Erickson, A. and Kiasari, A.E. and Navaridas, J. and Stewart, I.A. (2015) 'An efficient shortest path routing algorithm in the data centre network DPillar.', in Combinatorial optimization and applications: 9th International Conference, COCOA 2015, Houston, TX, USA, December 18-20, 2015, proceedings. , pp. 209-220. Lecture notes in computer science., 9486

Further information on publisher's website:

http://dx.doi.org/10.1007/978-3-319-26626-8$_1$6

Publisher's copyright statement:

The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-319-26626-8$_1$6

Additional information:

Use policy

The full-text may be used and/or reproduced, and given to third parties in any format or medium, without prior permission or charge, for personal research or study, educational, or not-for-profit purposes provided that:

- a full bibliographic reference is made to the original source
- a link is made to the metadata record in DRO
- the full-text is not changed in any way

The full-text must not be sold in any format or medium without the formal permission of the copyright holders.

Please consult the full DRO policy for further details.

Durham University Library, Stockton Road, Durham DH1 3LY, United Kingdom
Tel : +44 (0)191 334 3042 | Fax : +44 (0)191 334 2971
https://dro.dur.ac.uk

arXiv:1509.01746v1 [cs.DC] 5 Sep 2015

An efficient shortest-path routing algorithm in the data centre network DPillar

Alejandro Erickson[1], Abbas Eslami Kiasari[2], Javier Navaridas[2] and Iain A. Stewart[1]

[1] School of Engineering and Computing Sciences, Durham University, Science Labs, South Road, Durham DH1 3LE, U.K.
{alejandro.erickson,i.a.stewart}@durham.ac.uk
[2] School of Computer Science, University of Manchester, Oxford Road, Manchester M13 9PL, U.K.
{abbas.kiasari,javier.navaridas}@manchester.ac.uk

Abstract. DPillar has recently been proposed as a server-centric data centre network and is combinatorially related to the well-known wrapped butterfly network. We explain the relationship between DPillar and the wrapped butterfly network before proving a symmetry property of DPillar. We use this symmetry property to establish a single-path routing algorithm for DPillar that computes a shortest path and has time complexity $O(k \log(n))$, where k parameterizes the dimension of DPillar and n the number of ports in its switches. Moreover, our algorithm is trivial to implement, being essentially a conditional clause of numeric tests, and improves significantly upon a routing algorithm earlier employed for DPillar. A secondary and important effect of our work is that it emphasises that data centre networks are amenable to a closer combinatorial scrutiny that can significantly improve their computational efficiency and performance.
Key words: data centre networks, routing algorithms, shortest paths

1 Introduction

A *data centre network* (*DCN*) is the topology by which the servers, switches and other components of a data centre are interconnected and the choice of DCN strongly influences the data centre's practical performance (see, e.g., [13]). DCNs have traditionally been tree-like and *switch-centric*; that is, so that the servers are located at the 'leaves' of a tree-like structure that is composed entirely of switches and where the interconnection intelligence resides within the switches. Typical examples of such switch-centric DCNs are ElasticTree [9], Fat-Tree [4], VL2 [5], HyperX [3], Portland [14] and Flattened Butterfly [1]. However, it is generally acknowledged that tree-like, switch-centric DCNs have deficiencies when it comes to, for example, scalability with the core switches (at the 'roots' of the tree-like structure) quickly becoming bottlenecks.

Alternative architectures have recently emerged and *server-centric* DCNs have been proposed whereby the interconnection intelligence resides within the

servers as opposed to the switches. Now, switches only operate as dumb crossbars (and so the need for high-end switches is diminished as are the infrastructure costs). This paradigm shift means that more scalable topologies can be designed and the fact that routing resides within servers means that more effective routing algorithms can be adopted. However, packet latency can increase in server-centric DCNs, with the need to handle routing providing a computational overhead on the server. Nevertheless, server-centric data centres are now becoming commercially available. Typical examples of server-centric DCNs are DCell [7], BCube [8], FiConn [10], CamCube [2], MCube [15], DPillar [12], HCN and BCN [6] and SWKautz, SWCube and SWdBruijn [11]. An additional positive aspect of some server-centric DCNs is that not only can commodity switches be used to build the data centres but commodity servers can too: the DCNs FiConn, MCube, DPillar, HCN, BCN, SWKautz, SWCube and SWdBruijn are all such that any server only needs two NIC ports (the norm in commodity servers) in order to incorporate it into the DCN.

It is with the DCN DPillar that we are concerned here. In [12], basic properties of DPillar are demonstrated and single-path and multi-path routing algorithms are developed (along with a forwarding methodology for the latter). Our focus here is on single-path routing. The algorithm in [12] is appealing in its simplicity but for most source-destination pairs it does not produce a path of shortest length. We remedy this situation and develop a single-path routing algorithm that always outputs a shortest path and does so in linear time complexity. What is more, although the proof of correctness of our algorithm is non-trivial, the actual algorithm itself is a very simple sequence of numeric tests and consequently yields no implementation difficulties.

A pervasive theme within our work is that the design and performance of modern data centres can benefit significantly from additional combinatorial and mathematical analysis. Often, when new DCNs are proposed they are done so within a broader context so that the topology is considered as part of a wider and more practically-driven network environment. As such, the analysis is often empirical with a key aim being to demonstrate the practical viability of the DCN taking into account issues relating to, for example, infrastructure costs, traffic patterns, fault tolerance, network protocols and so on. Such presentations are often impressive in holistic terms but unavoidably basic in terms of combinatorial sophistication: the driver of practical viability means that there is a lessened inclination to optimize the various intrinsic components. It is once practical viability has been established that a closer combinatorial scrutiny can lead to improved performance (as we demonstrate here).

2 The DCN DPillar

We abstract the DCN DPillar as an undirected graph whose nodes represent the servers of the DCN DPillar and whose edges represent pairs of servers that are connected to the same switch. Representing a sever-switch-server connection of the DCN DPillar by one edge of the graph serves to reflect the negligible latency

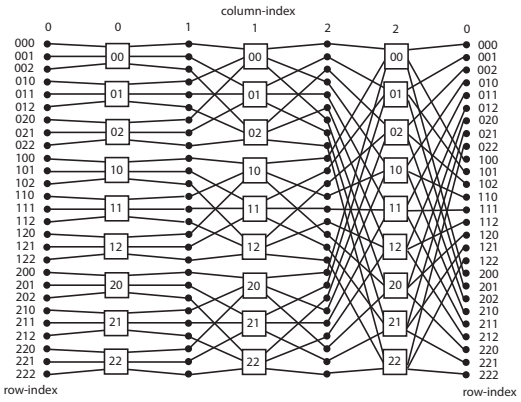

Fig. 1. Visualizing $\text{DPillar}_{6,3}$.

overheads encountered in a crossbar switch as compared to the overheads of routing through a server. We call this graph $\text{DPillar}_{n,k}$ when the DCN has *dimension* k and each switch has n-ports, where $2|n$. A node in *column* c with *row-index* $v_{k-1}v_{k-2}\cdots v_0$ is labelled $(c, v_{k-1}v_{k-2}\cdots v_0)$ with $0 \leq c < k$ and $0 \leq v_i < \frac{n}{2}$ for $0 \leq i < k$. $\text{DPillar}_{n,k}$ has 4 types of edges, called *clockwise edges (c-edges)*, *anti-clockwise edges (a-edges)*, *basic static edges (b-edges)*, and *decremented static edges (d-edges)*, which are of the following form:

$$\text{(c): } ((c, v_{k-1}\ldots v_{c+1}v_cv_{c-1}\ldots v_0), (c+1, v_{k-1}\ldots v_{c+1} * v_{c-1}\ldots v_0))$$
$$\text{(a): } ((c, v_{k-1}\ldots v_cv_{c-1}v_{c-2}\ldots v_0), (c-1, v_{k-1}\ldots v_c * v_{c-2}\ldots v_0))$$
$$\text{(b): } ((c, v_{k-1}\ldots v_{c+1}v_cv_{c-1}\ldots v_0), (c, v_{k-1}\ldots v_{c+1} * v_{c-1}\ldots v_0))$$
$$\text{(d): } ((c, v_{k-1}\ldots v_cv_{c-1}v_{c-2}\ldots v_0), (c, v_{k-1}\ldots v_c * v_{c-2}\ldots v_0)).$$

The c and b edges characterise the servers connected to the same column-c switch as the server represented by $(c, v_{k-1}v_{k-2}\cdots v_0)$, whilst the a and d edges characterise the servers connected to the same column-$(c-1)$ switch as the server represented by $(c, v_{k-1}v_{k-2}\cdots v_0)$. The DCN $\text{DPillar}_{6,3}$ can be visualized as in Fig. 1, where switches are shown and the left-most and right-most columns are actually the same columns, represented twice for clarity, with the graph $\text{DPillar}_{6,3}$ obtained by replacing each switch by a clique of edges on 6 nodes.

We shall rely on symmetry within $\text{DPillar}_{n,k}$; the term is used but not defined in [12], and the literature on DCN design tends to use 'symmetry' somewhat loosely. We omit the (straightforward) proof of the following result due to space constraints. Our notion of symmetry is node-symmetry (i.e., vertex-transitivity).

Lemma 1. *The graph* $\text{DPillar}_{n,k}$ *is node-symmetric.*

As a result of Lemma 1, the problem of routing from a node *src* to a node *dst*, is equivalent to the problem of routing from node $\phi(src)$ to the node $\phi(dst) = (0, 00\cdots0)$, where ϕ is an automorphism of $\text{DPillar}_{n,k}$ (such

automorphisms may be constructed explicitly to prove Lemma 1). Suppose a $(\phi(src), \phi(dst))$-path $p_0, p_1, \ldots, p_{\ell-1}$ is found. The desired (src, dst)-path is $\phi^{-1}(p_0), \phi^{-1}(p_1), \ldots, \phi^{-1}(p_{\ell-1})$.

3 Abstracting routing in DPillar

In this section we describe the single-path routing algorithms from [12] in order to motivate an abstraction that serves to describe a broad class of useful single-path routing algorithms.

Let $\mathbf{0}$ denote the node $(0, 00\cdots0)$. Consider the problem of routing from $src = (c, v_{k-1}v_{k-2}\cdots v_0)$ to $dst = \mathbf{0}$ where each step is made using an edge of type c, a, b, or d. The nodes reachable from src via c-, a-, b-, and d-edges, given in Section 2, differ from src in at most one of coordinates $c-1$ and c of the row-index (and no others), and lie in one of columns $c-1, c$, or $c+1$. Any non-zero symbols v_i of the row-index of src must be 'fixed' in one of these steps in order to reach dst, which has row index $00\cdots0$; v_i can only be fixed by visiting a column of the graph where coordinate i can be changed by one of the edges of type c, a, b, or d. Recall that edges of type c and b outgoing from src, in column c, enable us to change symbol v_c to whichever element of $\{0, 1, \ldots, n/2 - 1\}$ is desired; as such, we say that c- and b-edges *cover* the column they originate in. Edges of type a and d cover column $c-1$; as such, we say that a- and d-edges cover the anti-clockwise neighbouring column. In addition to fixing bits by covering the appropriate columns, the route may need to travel from column to column, via c- and a- edges, possibly without changing the row-index.

One of the routing algorithms detailed in [12] is to travel from src only along c-edges whilst changing a symbol v_i to 0, if necessary, at each step until dst is reached. After at most k steps, every column i with a non-zero coordinate v_i has been covered and fixed and the node $(j, 00\cdots0)$ is reached, for some j. We then continue to travel along c-edges to increase j until $\mathbf{0}$ is reached and the route is complete. The other single-path routing algorithm of [12] is the anticlockwise analogue, where only a-edges are used. It is very easy to see (by looking at some typical source-destination examples) that this routing algorithm is by no means optimal and that more often than not much shorter paths exist (an upper bound of $2k-1$ on the lengths of paths produced was stated in [12]). For example, if one chooses to route with only c-edges (in a clockwise fashion) in $\mathrm{DPillar}_{n,k}$ and the source is $\mathbf{0}$ and the destination is $(1, 10\ldots0)$ then the routing algorithm in [12] yields a path of length $k+1$ because the $(k-1)$st column must be visited in order to change the $(k-1)$st coordinate; the a-edge algorithm in [12] yields a path of length $k-1$. Neither of these algorithms are optimal (when $k > 3$), however, since the shortest path is of length 2 and can only be achieved by following a d-edge and then a c-edge to yield the path $\mathbf{0}$, $(0, 10\ldots0)$, $(1, 10\ldots0)$.

3.1 Another abstraction: the marked cycle

Observe in the above discussion that the need to cover column c and fix the cth coordinate of the row-index arises if, and only if, $v_c \neq 0$, but it does not

matter here what other value v_c may take on. Consequently, for arbitrary nodes *src* and *dst*, we instead consider a walk on a cycle on k-nodes, $G_{n,k}(src, dst)$, with a node corresponding to each column of $\text{DPillar}_{n,k}$ in which we *mark* the ith node whenever the ith column must be covered in $\text{DPillar}_{n,k}$; that is, where the coordinates of the row-indices of *src* and *dst* differ. Let src' and dst' be the columns of *src* and *dst*. We abstract a path from *src* to *dst* in $\text{DPillar}_{n,k}$ by a sequence of *moves* in $G_{n,k}(src, dst)$ starting with node src' and ending at node dst', where each move is analogous to a c-, a-, b-, or d-edge. From node c: (i) a *c-move covers* node c and moves to node $c+1$, (ii) an *a-move covers* and moves to node $c-1$, (iii) a *b-move covers* and stays at node c; and, (iv) a *d-move covers* node $c-1$ and stays at node c. We call $G_{n,k}(src, dst)$ a *marked cycle*.

It should be clear as to how moves in the marked cycle $G_{n,k}(src, dst)$ correspond to edges of type c, a, b, and d in $\text{DPillar}_{n,k}$ (and so to server-switch-server link-pairs in the DCN $\text{DPillar}_{n,k}$) with the coverage of a node in $G_{n,k}(src, dst)$ and a node of $\text{DPillar}_{n,k}$ being in direct correspondence. A *path* in $G_{n,k}(src, dst)$ is a sequence of moves leading from src' to dst' and corresponds to a path of the same length in $\text{DPillar}_{n,k}$ from node *src* to node *dst* (and *vice versa*). Consequently, in order to find a shortest (src, dst)-path in the DCN $\text{DPillar}_{n,k}$, it suffices to find a shortest (src', dst')-path in the marked cycle $G_{n,k}(src, dst)$ so that every marked node is covered by a move.

4 Routing in a marked cycle

We make some initial observations about shortest paths in a marked cycle, and then prove a structural result on shortest paths. Let *src* and *dst* be nodes of $\text{DPillar}_{n,k}$, in columns src' and dst', respectively. Henceforth, ρ is a shortest path from src' to dst' in $G_{n,k}(src, dst)$; therefore, ρ is a sequence of moves. We denote a sequence of moves by strings of the (corresponding) letters c, a, b, and d so that, for example, *ccbaaa* represents two c-moves, followed by a b-move, followed by three a-moves. In addition i repeated symbols, say, $cc\cdots c$ can be written c^i so that $ccbaaa = c^2ba^3$.

We can often rule out certain consecutive pairs of moves in ρ. For example, if we have a subsequence bd then this has the same effect as db, and so we may suppose that a subsequence db within ρ is forbidden. We can achieve much more by arguing on the optimality (as regards length) of ρ; for example, suppose the subsequence of moves ca occurs in ρ. We can replace ca by a b-move so as to obtain a shorter path with identical coverage to ρ. This contradicts the optimality of ρ, so we may assume that ca does not occur in ρ. Similarly, ac can be replaced by a d-move, so we may assume ac does not occur in ρ. Continuing in this manner, we obtain two tables of move-pair replacements, given below, whose (i, j)th entries represent the following: the move in the i-th position of the first column followed by the move in the j-th position of the first row must be replaced by the move given in the the (i, j)th entry of the table.

	b c
a	a d
b	b c

	a d
c	b c
d	a d

We describe the structure of paths resulting from similar arguments.

Lemma 2. *If ρ has length at least 3 then it must be of one of two forms:*

$$d^{\epsilon}c^{i_1}ba^{j_1}dc^{i_2}\dots c^{i_m}b^{\delta} \quad \text{or} \quad d^{\epsilon}c^{i_1}ba^{j_1}dc^{i_2}\dots a^{j_m}d^{\delta}, \tag{1}$$

for some $m \geq 1$, where $i_1, i_2, \dots, i_m, j_1, j_2, \dots, j_m > 1$ and where $\epsilon, \delta \in \{0, 1\}$;

$$b^{\epsilon}a^{i_1}dc^{j_1}ba^{i_2}\dots a^{i_m}d^{\delta} \quad \text{or} \quad b^{\epsilon}a^{i_1}dc^{j_1}ba^{i_2}\dots c^{j_m}b^{\delta}, \tag{2}$$

for some $m \geq 1$, where $i_1, i_2, \dots, i_m, j_1, j_2, \dots, j_m > 1$ and where $\epsilon, \delta \in \{0, 1\}$.

Proof. Omitted due to space constraints (it is a simple case analysis).

4.1 A shortest path has at most two turns

If we have a c-move followed by a b-move followed by an a-move in ρ then we say that an *anti-clockwise turn*, or simply an *a-turn*, occurs at the b-move; similarly, if we have an a-move followed by a d-move followed by a c-move then we say that a *clockwise turn*, or simply a *c-turn*, occurs at the d-move. Note that if we have an a-turn in ρ then the node at which this turn occurs, i.e., the node that is covered by the d-move, must be marked in $G_{n,k}(src, dst)$ as otherwise we could delete the corresponding d-move from ρ and still have a sequence from src' to dst' covering all the marked nodes, which would yield a contradiction. Similarly, if we have a c-turn then the node at which this c-turn occurs, i.e., the node that is covered by the b-move, must be marked. We will use these observations later; but now we prove that any shortest path ρ must contain at most 2 turns.

Suppose that ρ is a shortest path and has at least 3 turns.
Case (a): Suppose that ρ is of Form (1) and has a prefix ρ' of the form $c^iba^jdc^lba$, where $i, j, l \geq 1$. By this we mean that ρ begins with i c-moves followed by a b-move followed by j a-moves followed by a d-move followed by l c-moves followed by a b-move followed by an a-move.

If $j < i$ then we can replace the prefix c^iba^jdc in ρ' with c^iba^{j-1} and still obtain the same coverage; this contradicts that ρ is a shortest path (note that we have actually only assumed so far that ρ has 2 turns). If $j = i$ then we can replace the prefix c^iba^idc in ρ' with dc^iba^{i-1} so as to obtain a contradiction (we have still actually only assumed that ρ has 2 turns). Hence, we must have that $j > i$. Suppose that $j \geq l > j - i$. We can replace the prefix $c^iba^jdc^l$ in ρ' with $a^{j-i}dc^jba^{j-l}$ so as to obtain a contradiction (we have still actually only assumed that ρ has 2 turns). Hence, $j > i$ and either $l \leq j - i$ or $l > j$.

Suppose that $l > j$. We can replace the prefix $c^iba^jdc^l$ in ρ' with $a^{j-i}dc^l$ so as to obtain a contradiction (we have still actually only assumed that ρ has 2 turns). Hence, we must have that $j > i$ and $l \leq j - i$. However, if we replace ρ' with $c^iba^jdc^{l-1}$ then we obtain a contradiction (here we do use the fact that ρ

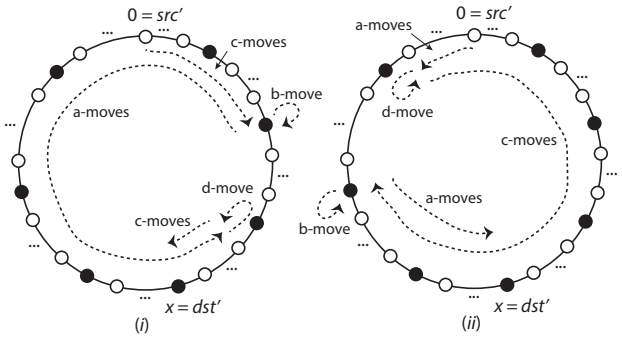

Fig. 2. Visualizing paths with 2 turns.

has at least 3 turns). So, ρ has at most 2 turns and if it has 2 turns then ρ is of the form $c^iba^jdc^l$ where $j > i$ and $l \leq j - i$.

We can say more if ρ has 2 turns. Suppose that $j \geq k - 1$. The b-move can be deleted from ρ' and we obtain a contradiction. Hence, if ρ has 2 turns then ρ is of the form $c^iba^jdc^l$ where $k - 1 > j > i \geq 1$ and $1 \leq l \leq j - i$. We can visualize ρ as in Fig. 2(i). The marked cycle $G_{n,k}(src, dst)$ is shown as a cycle where a black node denotes a node of B; that is, a node that needs to be covered by some path in $G_{n,k}(src, dst)$ (with $0 = src' \neq dst' = x$ in this illustration). The path ρ is depicted as a dotted line partitioned into composite moves.

Case (b): Suppose that ρ is of Form (1) and has a prefix ρ' of the form $dc^iba^jdc^lba$, where $i, j, l \geq 1$. If $j \leq i$ then we can replace the prefix dc^iba^jdc in ρ' with dc^iba^jc so as to obtain a contradiction, and if $j > i$ then we can delete the first d-move from ρ to obtain a contradiction. Hence, if ρ starts with a d-move then it has at most 1 turn.

Case (c): Suppose that ρ is of Form (2) and has a prefix ρ' of the form $a^idc^jba^ldc$, where $i, j, l \geq 1$. If $j < i$ then we can replace the prefix a^idc^jba in ρ' with a^idc^{j-1} so as to obtain a contradiction. If $i = j$ then we can replace the prefix a^idc^iba in ρ' with ba^idc^{i-1} so as to obtain a contradiction. Hence, $j > i$.

Suppose that $j \geq l > j - i$. We can replace the prefix $a^idc^jba^l$ in ρ with $c^{j-i}ba^jdc^{j-l}$ so as to obtain a contradiction. Suppose that $l > j$. We can delete the first occurrence of a d-move in ρ so as to obtain a contradiction. Hence, $l \leq j - i$. Note that if ρ has 2 turns then ρ is of the form $a^idc^jba^l$ where $j > i$ and $l \leq j - i$. Alternatively, suppose that ρ has at least 3 turns. We can replace the prefix $a^idc^jba^ldc$ in ρ with $a^idc^jbc^{l-1}$ so as to obtain a contradiction. Hence, ρ has at most 2 turns.

We can say more if ρ has 2 turns. Suppose that $j \geq k - 1$. The d-move can be deleted from ρ' and we obtain a contradiction. Hence, if ρ has 2 turns then ρ is of the form $a^idc^jbd^l$ where $k - 1 > j > i \geq 1$ and $1 \leq l \leq j - i$. We can visualize ρ as in Fig. 2(ii).

Case (d): Suppose that ρ is of Form (2) and has a prefix ρ' of the form $ba^idc^jba^ldc$, where $i, j, l \geq 1$. If $j \leq i$ then we can replace the prefix ba^idc^jba with ba^idc^ja

so as to obtain a contradiction, and if $j > i$ then we can delete the first b-move from ρ to obtain a contradiction. Hence, if ρ starts with a b-move then it has at most 1 turn.

So, we have proven the following lemma.

Lemma 3. *If ρ is a shortest path (from src' to dst') in $G_{n,k}(src, dst)$ then ρ has at most 2 turns, and if ρ has 2 turns then it must be of the form $c^i ba^j dc^l$ or $a^i dc^j ba^l$, where $k-1 > j > i \geq 1$ and $1 \leq l \leq j-i$.*

With reference to Fig. 2, the numerical constraints in Lemma 3 mean that there is no interaction or overlap involving the 2 turns in ρ.

5 An optimal routing algorithm for DPillar

We now develop an optimal single-path routing algorithm for DPillar. We do this by finding a small set Π of paths (from src' to dst') in $G_{n,k}(src, dst)$ so that at least one of these paths is a shortest path. By Lemma 1, we may assume that $src = \mathbf{0}$ and $dst = (x, v_{k-1}v_{k-2}\ldots v_0)$, and by Lemma 2, we may assume that any shortest path has at most 2 turns.

5.1 Building our set of paths when $x \neq 0$

We first suppose that $0 \neq x$. Let $B = \{i : 0 \leq i \leq k-1, v_i \neq 0\}$ (that is, the bit-positions that need to be 'fixed'). Suppose that $B \setminus \{0, x\} = \{i_l : 1 \leq l \leq r\} \cup \{j_l : 1 \leq l \leq s\}$ so that we have $0 < j_s < j_{s-1} < \ldots < j_1 < C < i_1 < i_2 < \ldots < i_r < k$ (we might have that either r or s is 0 when the corresponding set is empty). If $r \geq 2$ then define $\delta_l = i_{l+1} - i_l$, for $l = 1, 2, \ldots, r-1$, with $\delta = \max\{\delta_l : l = 1, 2, \ldots, r-1\}$; and if $s \geq 2$ then define $\epsilon_l = j_l - j_{l+1}$, for $l = 1, 2, \ldots, s-1$, with $\epsilon = \max\{\epsilon_l : l = 1, 2, \ldots, s-1\}$. Also: define $\Delta_0 = 1$ (resp. 0), if $0 \in B$ (resp. $0 \notin B$); and $\Delta_x = 1$ (resp. 0), if $x \in B$ (resp. $x \notin B$). We can visualize the resulting marked cycle $G_{n,k}(0, x)$ as in Fig. 3(i). Note that in this particular illustration $0 \notin B$ and $x \in B$; so, $\Delta_0 = 0$ and $\Delta_x = 1$. Of course, what we are looking for is a sequence of (a-, b-, c- and d-)moves that will take us from 0 to x in $G_{n,k}(0, x)$ so that all nodes of B have been covered.

In what follows, we examine different scenarios involving the number of marked nodes, r, and also the number of marked nodes, s. Each scenario for r contributes certain paths to Π as does each scenario for s. Note that perhaps the most obvious paths to consider as potential members of Π are the paths c^{k+x} and a^{2k-x} which have lengths $k+x$ and $2k-x$, respectively. So, we begin by setting $\Pi = \{c^{k+x}, a^{2k-x}\}$.

From Lemma 3, any shortest path ρ from 0 to x having 2 turns requires that $r \geq 2$ or $s \geq 2$ and that both nodes at which these turns occur are different from 0 and x and lie on the anti-clockwise path from 0 to x or on the clockwise path from 0 to x, accordingly. Also, the node at which any turn occurs on a shortest path ρ is necessarily a marked node (irrespective of the number of turns in ρ).

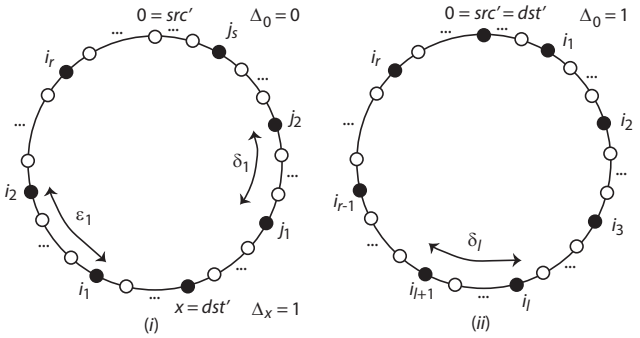

Fig. 3. Visualizing our notation.

Case (a): Suppose that $r = 0$. In this scenario, we contribute either the path $c^x b$ to Π, if $x \in B$, or the path c^x to Π, if $x \notin B$; either way, the length of the path contributed is $x + \Delta_x$.

Case (b): Suppose that $s = 0$. In this scenario, we contribute either the path ba^{k-x} to Π, if $0 \in B$, or the path a^{k-x} to Π, if $0 \notin B$; either way, the length of the path contributed is $k - x + \Delta_0$.

Case (c): Suppose that $r = 1$. In this scenario, we contribute 2 paths to Π. If $x \in B$ then we contribute the path $a^{k-i_1-1}dc^{k-i_1-1+x}b$ to Π, or if $x \notin B$ then we contribute the path $a^{k-i_1-1}dc^{k-i_1-1+x}$ to Π; either way, the length of the resulting path is $2k - 2i_1 + x - 1 + \Delta_x$. We also contribute the path $c^{i_1}ba^{i_1-x}$ to Π of length $2i_1 - x + 1$. There is potentially another path when $i_1 = x + 1$ and $x \in B$, namely $a^{k-x-1}dc^{k-1}$, but the length of this path is $2k - x - 1$ which is greater than $2k - x - 3 + \Delta_x$ which is $2k - 2i_1 + x - 1 + \Delta_x$ evaluated with $i_1 = x + 1$.

Case (d): Suppose that $s = 1$. In this scenario, we contribute 2 paths to Π. If $0 \in B$ then we contribute the path $ba^{k-j_1-1}dc^{x-j_1-1}$ to Π, or if $0 \notin B$ then we contribute the path $a^{k-j_1-1}dc^{x-j_1-1}$ to Π; either way, the length of the resulting path is $k - 2j_1 + x - 1 + \Delta_0$. We also contribute the path $c^{j_1}ba^{k+j_1-x}$ to Π of length $k + 2j_1 - x + 1$. There is potentially another path when $j_1 = 1$ and $0 \in B$, namely $a^{k-1}dc^{x-1}$, but the length of this path is $k + x - 1$ which is greater than $k + x - 3 + \Delta_0$ which is $k - 2j_1 + x - 1 + \Delta_0$ evaluated with $j_1 = 1$.

Case (e): Suppose that $r \geq 2$. In this scenario, we contribute $r+1$ paths to Π. For each $l \in \{1, 2, \ldots, r-1\}$, we contribute the path $a^{k-i_{l+1}-1}dc^{k-i_{l+1}-1+i_l}ba^{i_l-x}$ to Π of length $2k-2\delta_l-x$. If $x \in B$ then we contribute the path $a^{k-i_1-1}dc^{k-i_1-1+x}b$ to Π, or if $x \notin B$ then we contribute the path $a^{k-i_1-1}dc^{k-i_1-1+x}$ to Π; either way, the length of the path is $2k - 2i_1 + x - 1 + \Delta_x$. We also contribute the path $c^{i_r}ba^{i_r-x}$ to Π of length $2i_r - x + 1$. (These last 2 paths mirror those constructed in Case (c).)

Case (f): Suppose that $s \geq 2$. In this scenario, we contribute $s + 1$ paths to Π. For each $l \in \{1, 2, \ldots, s-1\}$, we contribute the path $c^{j_{l+1}}ba^{j_{l+1}+k-j_l-1}da^{x-j_l-1}$

to Π of length $k-2\epsilon_l+x$. If $0 \in B$ then we contribute the path $ba^{k-j_s-1}dc^{x-j_s-1}$ to Π, or if $0 \notin B$ then we contribute the path $a^{k-j_s-1}dc^{x-j_s-1}$ to Π; either way, the length of the path is $k-2j_s+x-1+\Delta_0$. We also contribute the path $c^{j_1}ba^{j_1+k-x}$ to Π of length $k+2j_1-x+1$. (These last 2 paths mirror those constructed in Case (c).)

Thus, our set Π of potential shortest paths contains $r+s+2$ paths (from which at least one is a shortest path).

5.2 Building our set of paths when $x = 0$

Now we suppose that $x = 0$. We proceed as we did above and build a set Π of potential shortest paths. Let $B = \{i : 0 \leq i \leq k-1, v_i \neq 0\}$. Suppose that $B \setminus \{0\} = \{i_l : 1 \leq l \leq r\}$ so that we have $0 < i_1 < i_2 < \ldots < i_r < k$ (we might have that r is 0 when the corresponding set is empty). If $r \geq 2$ then define $\delta_l = i_{l+1} - i_l$, for $l = 1, 2, \ldots, r-1$, with $\delta = \max\{\delta_l : l = 1, 2, \ldots, r-1\}$. We define $\Delta_0 = 1$, if $0 \in B$, and $\Delta_0 = 0$, if $0 \notin B$. We can visualize the resulting marked cycle $G_{n,k}(0,0)$ as in Fig. 3(ii). Again, the most obvious path to consider is c^k (or a^k) which has length k. We begin by setting $\Pi = \{c^k\}$.

Case(a): Suppose that $r = 0$. In this scenario, we contribute the path b of length 1 (note that in this case the node 0 is necessarily marked as we originally assumed that we started with distinct source and destination servers in the DCN DPillar$_{n,k}$).

Case(b): Suppose that $r = 1$. If $i_1 = k-1$ then we contribute the path bd, if $0 \in B$, and the path d, if $0 \notin B$; either way, the path has length $1+\Delta_0$. If $1 = i_1 \neq k-1$ then we contribute the path cba of length 3. If $1 \neq i_1 \neq k-1$ then we contribute 2 paths. The first of these paths is the path $ba^{k-i_1-1}dc^{k-i_1-1}$, if $0 \in B$, and the path $a^{k-i_1-1}dc^{k-i_1-1}$, if $0 \notin B$; either way, this path has length $2k-2i_1-1+\Delta_0$. The second of these paths is the path $c^{i_1}ba^{i_1}$ of length $2i_1+1$.

Case(c): Suppose that $r \geq 2$. In this scenario, we contribute $r+1$ paths to Π. For each $l \in \{1, 2, \ldots, r-1\}$, we contribute the path $a^{k-i_{l+1}-1}dc^{k-i_{l+1}-1+i_l}ba^{i_l}$ to Π of length $2k-2\delta_l$. If $0 \in B$ then we contribute the path $ba^{k-i_1-1}dc^{k-i_1-1}$ to Π, or if $0 \notin B$ then we contribute the path $a^{k-i_1-1}dc^{k-i_1-1}$ to Π; either way, this path has length $2k-2i_1-1+\Delta_0$. We also contribute the path $c^{i_r}ba^{i_r}$ to Π of length $2i_r+1$. (These last 2 paths mirror those constructed in Case (b).)

Thus, our set Π of potential shortest paths contains at most $r+1$ paths (from which at least one is a shortest path).

5.3 Our algorithm

We now use our set Π of potential shortest paths so as to find a shortest path or the length of a shortest path. Our algorithm for $G_{n,k}(0,x)$ is as follows.

```
calculate B
if 0 ≠ x then
   L = min{k + x, 2k − x}
   calculate r, s, δ, ϵ, Δ_0 and Δ_x
```

```
   if r = 0 then L = min{L, x + Δ_x}
   if s = 0 then L = min{L, k − x + Δ_0}
   if r = 1 then L = min{L, 2k − 2i_1 + x − 1 + Δ_x, 2i_1 − x + 1}
   if s = 1 then L = min{L, k − 2j_1 + x − 1 + Δ_0, k + 2j_1 − x + 1}
   if r ≥ 2 then
      calculate δ                    % we need only consider the maximal δ_l
      L = min{L, 2k − 2δ − x, 2k − 2i_1 + x − 1 + Δ_x, 2i_r − x + 1}
   if s ≥ 2 then
      calculate ϵ                    % we need only consider the maximal ϵ_l
      L = min{L, k − 2ϵ + x, k − 2j_s + x − 1 + Δ_0, k + 2j_1 − x + 1}
else
   calculate r and δ
   if r = 0 then L = 1
   if r = 1 then
      if i_1 = k − 1 then L = 1 + Δ_0
      if 1 = i_1 ≠ k − 1 then L = 3
      if 1 ≠ i_1 ≠ k − 1 then L = min{2k − 2i_1 − 1 + Δ_0, 2i_1 + 1}
   if r ≥ 2 then L = min{k, 2k − 2δ, 2k − 2i_1 − 1 + Δ_0, 2i_r + 1}
output L
```

If we wish to output a shortest path then all we do is apply the above algorithm but remember which shortest path corresponds to the final value of L and output this shortest path (note that there may be more than one shortest path; exactly which path one obtains depends upon how one implements checking the paths of Π). The time complexity of both algorithms is clearly $O(k \log(n))$ (that is, linear in the length of the input).

It should be clear (using Lemma 2) that the different considerations for r and s exhaust all possibilities and that consequently the set of paths Π considered by the above algorithm is such as to contain a shortest path. Hence, our algorithm outputs the length of a shortest path from some source node to some destination node in $\mathrm{DPillar}_{n,k}$. The validity of our algorithm was verified with a breadth-first search and we also found that it yields a 20–30% improvement in the average length of a path over the algorithms given in [12], for various small parameters n and k; for example, $\mathrm{DPillar}_{16,5}$, which has 163840 server-nodes, has an average shortest path length of 4.77, but employing the clockwise algorithm from [12] yields an average path length of 6.86.

6 Conclusions

In this paper we have developed an optimal and efficient single-path routing algorithm for the DCN DPillar and have shown that DPillar is node-symmetric. We feel that there are other areas where efficiency gains might be made; in particular, we intend to focus on multi-path routing in forthcoming research.

Acknowledgement This work has been funded by the Engineering and Physical Sciences Research Council (EPSRC) through grants EP/K015680/1 and EP/K015699/1.

References

1. D. Abts, M.R. Marty, P.M. Wells, P. Klausler and H. Liu, "Energy Proportional Datacenter Networks", *Proc. of 37th Ann. Int. Symp. on Comput. Arch.*, 2010, pp. 338–347.
2. H. Abu-Libdeh, P. Costa, A. Rowstron, G. OShea and A. Donnelly, "Symbiotic Routing in Future Data Centers", *SIGCOMM Comput. Comm. Rev.*, vol. 40, no. 4, 2010, pp. 51–62.
3. J.H. Ahn, N. Binkert, A. Davis, M. McLaren and R.S. Schreiber, "HyperX: Topology, Routing, and Packaging of Efficient Large-scale Networks", *Proc. of Conf. on High Perf. Comput. Networking, Storage and Analysis*, 2009, article 41.
4. M. Al-Fares, A. Loukissas and A. Vahdat, "A Scalable, Commodity Data Center Network Architecture", *SIGCOMM Comput. Comm. Rev.*, vol. 38, no. 4, 2008, pp. 63–74.
5. A. Greenberg, J.R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D.A. Maltz, P. Patel and S. Sengupta, "VL2: A Scalable and Flexible Data Center Network", *SIGCOMM Comput. Comm. Rev.*, vol. 39, no. 4, 2009, pp. 51–62.
6. D. Guo, T. Chen, D. Li, M. Li, Y. Liu and G. Chen, "Expandible and Cost-effective Network Structures for Data Centers using Dual-port Servers", *IEEE Trans. Comput.*, vol. 62, no. 7, 2013, pp. 1303–1317.
7. C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang and S. Lu, "DCell: A Scalable and Fault-tolerant Network Structure for Data Centers", *SIGCOMM Comput. Comm. Rev.*, vol. 38, no. 4, 2008, pp. 75–86.
8. C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang and S. Lu, "BCube: A High Performance, Server-centric Network Architecture for Modular Data Centers", *SIGCOMM Comput. Comm. Rev.*, vol. 39, no. 4, 2009, pp. 63–74.
9. B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee and N. McKeown, "ElasticTree: Saving Energy in Data Center Networks", *Proc. 7th USENIX Conf. on Networked Systems Design and Implementation*", 2006, pp. 249–264.
10. D. Li, C. Guo, H.Wu, K. Tan, Y. Zhang and S. Lu, "FiConn: Using Backup Port for Server Interconnection in Data Centers", *Proc. of INFOCOM*, 2009, pp. 2276–2285.
11. D. Li and J. Wu, "On Data Center Network Architectures for Interconnecting Dual-port Servers", *IEEE Trans. Comput.*, 2015, to appear.
12. Y. Liao, J. Yin, D. Yin and L. Gao, "DPillar: Dual-port Server Interconnection Network for Large Scale Data Centers", *Comput. Networks*, vol. 56, no. 8, 2012, pp. 2132–2147.
13. Y. Liu, J.K. Muppala, M. Veeraraghavan, D. Lin and J. Katz, *Data Centre Networks: Topologies, Architectures and Fault-Tolerance Characteristics*, Springer, 2013.
14. R.N. Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya and A. Vahdat, "Portland: A Scalable Fault-tolerant Layer 2 Data Center Network Fabric", *SIGCOMM Comput. Comm. Rev.*, vol. 39, no. 4, 2009, pp. 39–50.
15. C. Wang, C. Wang, Y. Yuan and Y. Wei, "MCube: A High Performance and Fault-tolerant Network Architecture for Data Centers", *Proc. Int. Conf. on Comput. Design and App.*, vol. 5, 2010, pp. V5-423–V5-427.