Pegasus: The second connectivity graph for large-scale quantum annealing hardware

Nike Dattani[*]
Harvard-Smithsonian Center for Astrophysics and National Research Council of Canada

Szilard Szalay[†]
Wigner Research Centre for Physics

Nicholas Chancellor[‡]
Joint Quantum Centre, Durham University

Pegasus is a graph which offers substantially increased connectivity between the qubits of quantum annealing hardware compared to the graph Chimera. It is the first fundamental change in the connectivity graph of quantum annealers built by D-Wave since Chimera was introduced in 2009 and then used in 2011 for D-Wave's first commercial quantum annealer. In this article we describe an algorithm which defines the connectivity of Pegasus and we provide what we believe to be the best way to graphically visualize Pegasus in order to see which qubits couple to each other. As Supplemental Material, we provide a wide variety of different visualizations of Pegasus which expose different properties of the graph in different ways. We provide an open source code for generating the many depictions of Pegasus that we show.

The 128 qubits of the first commercial quantum annealer (D-Wave One, released in 2011) were connected by a graph called Chimera (first defined publicly in 2009 [1]), which is rather easy to describe: A 2D array of $K_{4,4}$ graphs, with one "side" of each $K_{4,4}$ being connected to the same corresponding side on the $K_{4,4}$ cells directly above and below it, and the other side being connected to the same corresponding side on the $K_{4,4}$ cells to the right and left of it (see Figure 1). The qubits can couple to up to 6 other qubits, since each qubit couples to 4 qubits within its $K_{4,4}$ unit cell, and to 1 qubit in a $K_{4,4}$ above it and 1 qubit in a $K_{4,4}$ to the right of it. All commercial quantum annealers built to date follow this graph connectivity, with just larger and larger numbers of $K_{4,4}$ unit cells (See Table 1).

Table I: Chimera graphs in all commercial quantum annealers to date.

	Array of $K_{4,4}$ unit cells	Total # of qubits
D-Wave One	4×4	128
D-Wave Two	8×8	512
D-Wave 2X	12×12	1152
D-Wave 2000Q	16×16	2048

In 2018, D-Wave announced the construction of a (not yet commercial) quantum annealer with a greater connectivity than Chimera offers, and a publicly available program (NetworkX) which allows users to generate Pegasus graphs of arbitrary size. However, no explicit description of the graph connectivity in Pegasus has been published yet, so we have had to apply the process of reverse engineering to determine it, and the following section describes the algorithm we have developed for generating Pegasus.

I. ALGORITHM FOR GENERATING PEGASUS

A. The vertices (qubits)

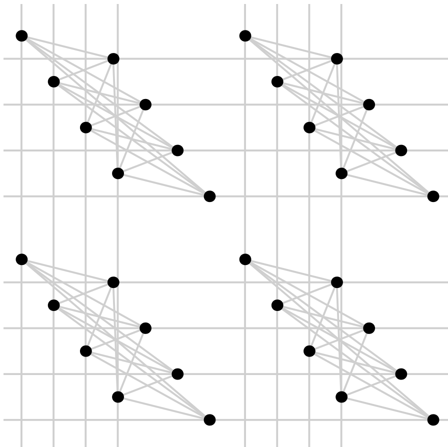
Start with Z layers of Chimera graphs, each being an $X \times Y$ array of $K_{4,4}$ unit cells (therefore we have an $X \times Y \times Z$ array of $K_{4,4}$ unit cells). The indices (x, y, z) will be used to describe the location of each unit cell along the indices corresponding to the dimension picked from (X, Y, Z). The values of X and Y have no restriction, but $Z = 3$ in Pegasus. Each $K_{4,4}$ cell has two "sides", labeled $i \in \{0, 1\}$, so that there are 4 qubits (vertices) for every combination: (x, y, z, i). We will arbitrarily label these 4 qubits using two more labels: $j \in \{0, 1\}$ and $k \in \{0, 1\}$. Therefore every qubit is associated with 6 indices: (x, y, z, i, j, k), with their ranges and descriptions given in Table II.

In all of the figures in this publication, the origin will be in the bottom-left corner, the x index will increase in the right-hand direction, the y index will increase in the upward direction, and the z index (which indicates which Chimera layer is being considered) will increase in the direction going upward and rightward simultaneously (since paper and computer screens are still two-dimensional): See Figure 2a. Then $i = 0$ will represent the left side in the classic $K_{4,4}$ depiction, or the horizontal vertices in the diamond-shaped or triangle-shaped depictions, while $i = 1$ will represent the right side in the classic and vertical vertices in the diamond and triangle.

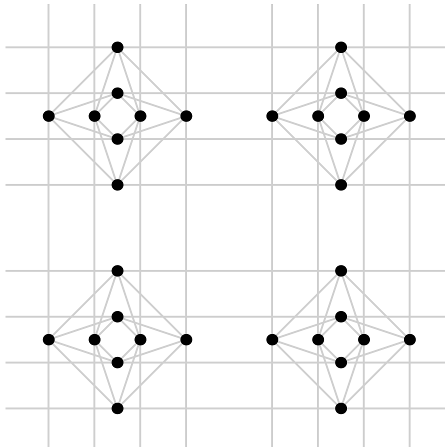
[*] n.dattani@cfa.harvard.edu
[†] szalay.szilard@wigner.mta.hu
[‡] nicholas.chancellor@durham.ac.uk

Figure 1: Three different depictions of the Chimera graph, with open edges to show that the pattern repeats.

(a) Tilted classic



(b) Diamond



(c) Upper triangle

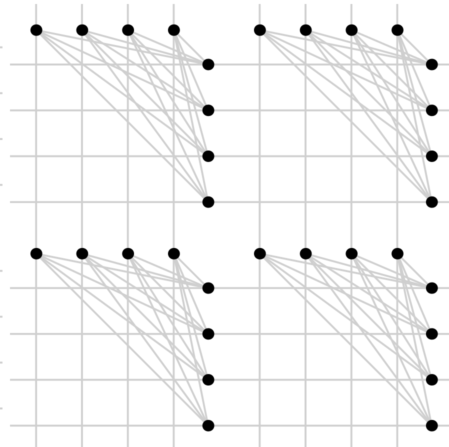
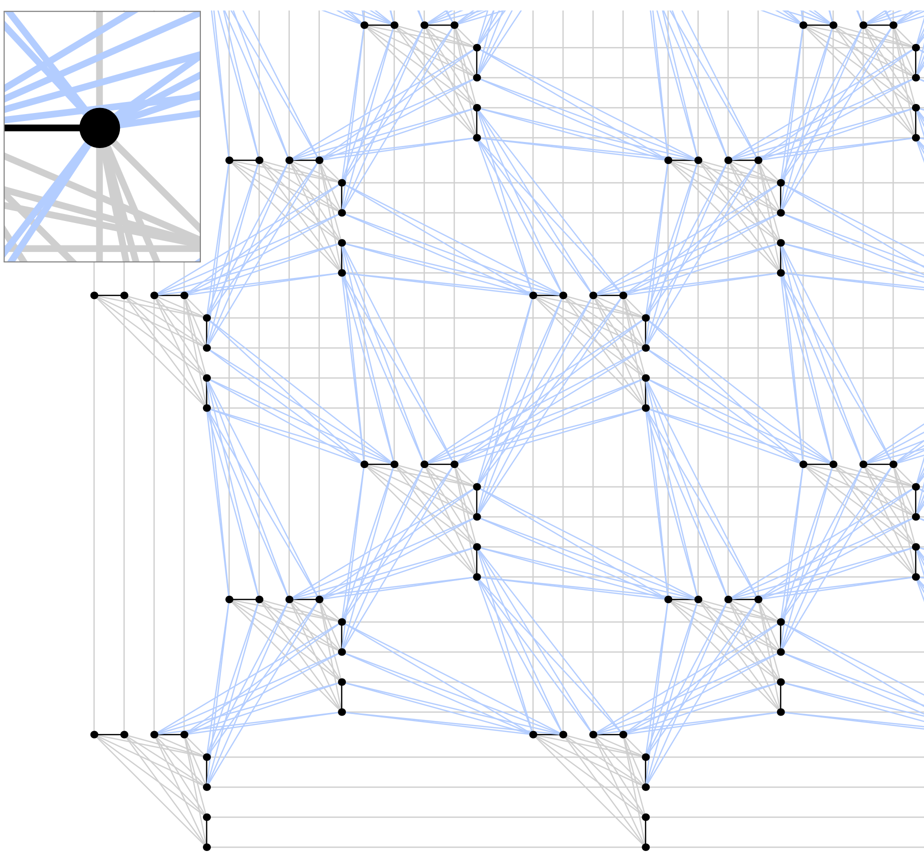
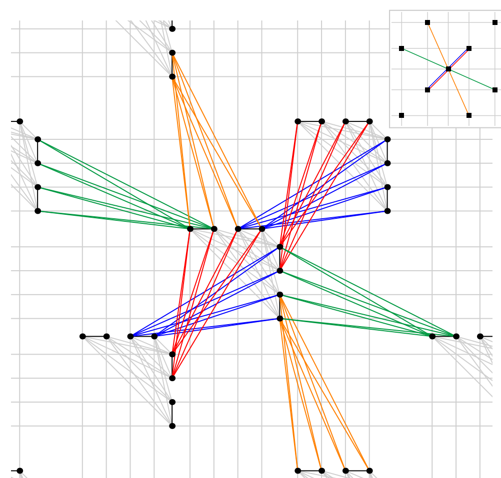


Figure 2: The Pegasus graph.

(a) A patch of size $(X, Y, Z) = (2, 2, 3)$ cropped out of a Pegasus graph. Grey edges are part of the three layers of Chimera graphs, while black and blue edges form the remainder of the Pegasus graph.



(b) Groups of $K_{2,4}$ edges connecting cells belonging to different Chimera layers, colored according to Eqs. (12)-(19).



(c) "Compressed" version of the Pegasus graph, with each cell represented by one vertex and each $K_{2,4}$ represented by one edge.

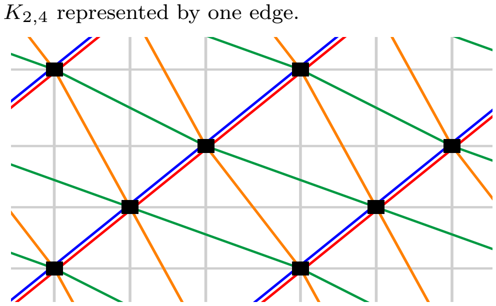


Table II: Indices used to describe each qubit (vertex) in Pegasus.

Index	Range	Description
x	0 to $X-1$	Row within a Chimera layer
y	0 to $Y-1$	Column within a Chimera layer
z	0 to 2	Chimera layer
i	$0,1$	Bi-partition within $K_{4,4}$
j	$0,1$	First index within each part of $K_{4,4}$
k	$0,1$	Second index within each part of $K_{4,4}$

B. The edges (couplings) in Chimera

1. *Edges forming each $K_{4,4}$ cell*

The $K_{4,4}$ cells are given by:

$$(x,y,z,0,j,k) \longleftrightarrow (x,y,z,1,j',k'). \tag{1}$$

This means for each $K_{4,4}$ cell, all vertices (j,k) for partition $i=0$ are coupled to all vertices (j',k') of partition $i=1$. ***For this entire publication, j' can be equal to or different from j (and likewise for $k'$and k).***

2. *Edges connecting different $K_{4,4}$ cells*

The horizontal lines between $K_{4,4}$ cells in Figure 1 can be described by adding 1 to x while keeping all other variables constant and setting $i=1$:

$$(x,y,z,1,j,k) \longleftrightarrow (x+1,y,z,1,j,k), \tag{2}$$

and the vertical lines can be described by adding 1 to y while keeping all other variables constant and setting $i=0$:

$$(x,y,z,0,j,k) \longleftrightarrow (x,y+1,z,0,j,k). \tag{3}$$

For each z, Equations (1)-(3) define edges connecting vertices labeled by x,y,i, j and k. This completes the definition of a Chimera graph. In all figures, these Chimera edges are grey.

C. The *new* edges (couplings) in Pegasus

1. *New edges added to each $K_{4,4}$ cell:*

Pegasus first adds connections within each $K_{4,4}$ cell, given by simply coupling each vertex labeled as $k=0$ to its $k=1$ counterpart with all other variables unchanged:

$$(x,y,z,i,j,0) \longleftrightarrow (x,y,z,i,j,1). \tag{4}$$

These edges are drawn black in the figures.

2. *Edges connecting different $K_{4,4}$ cells:*

The rest of the new connections in Pegasus come from connecting the $K_{4,4}$ cells between different layers (different z) of Chimera graphs. The qubits of a $K_{4,4}$ cell located at coordinates (x,y,z) will be connected to 6 different $K_{4,4}$ cells on the other Chimera layers, with 64 edges in the form of 8 different $K_{2,4}$ graphs: 1 $K_{2,4}$ graph (8 edges) for each of 4 different connecting $K_{4,4}$ cells, and 2 $K_{2,4}$ graphs (16 edges) for the other 2 connecting $K_{4,4}$ cells.

Figure 2b shows these 64 edges, and the 8 groups of 8-edge $K_{2,4}$ graphs connecting the central cell to 6 others are shown with 4 different colors. This is because we have found a set of 4 convenient rules that can be repeated to generate the entire Pegasus graph, and it is found that 4 of the $K_{2,4}$ graphs can be generated by applying these 4 rules to the central cell, and the other 4 of the $K_{2,4}$ graphs can be generated by applying these 4 rules to other $K_{4,4}$ cells (or by applying the 4 rules to the central cell again, but in reverse). We will now describe the rules.

First, all edges are between vertices of one $K_{4,4}$ part i and its complementary part $\bar{i}$ in a different $K_{4,4}$ (so $i=0$ vertices are coupled to $i=1$ vertices of a $K_{4,4}$ in a different layer). In fact all connections will be of the form $(i,j,k) \longleftrightarrow (\bar{i},j',k')$ where j' and k' can be any value in $\{0,1\}$. ***All edges can actually be described using just a one-line rule:***

$$\begin{gathered}(x,y,z,i,j,k)\\ \longleftrightarrow\\ \left(x-j\bar{i}+\delta_{z2},y-ji+\delta_{z2},(z+1) \bmod 3,\bar{i},j',k'\right).\end{gathered} \tag{5}$$

For the layers labeled $z=0$ or 1 (meaning that $\delta_{z2}=0$), we have:

$$(x,y,z,i,j,k) \longleftrightarrow (x-j\bar{i},y-ji,z+1,\bar{i},j',k'). \tag{6}$$

Substituting $j=0$ into Eq. (6) tells us that all $j=0$ vertices of a $K_{4,4}$ cell are connected to all vertices of the opposite part $\bar{i}$ in the $K_{4,4}$ cell with the same (x,y), but next layer $z+1$:

$$(x,y,z) \longleftrightarrow (x,y,z+1), \text{ for } z\in\{0,1\}, j=0. \tag{7}$$

Substituting $j=1$ into Eq. (6) tells us that all $j=1$ vertices are also connected to all j' and k' vertices in the opposite part $\bar{i}$ in a $K_{4,4}$ cell in the next layer $z+1$, but with x and y coordinates shifted by 1 in the following way:

$$(x,y,z) \longleftrightarrow (x-\bar{i},y-i,z+1) \text{ for } z\in\{0,1\}, j=1. \tag{8}$$

For the $z=2$ layer ($\delta_{z2}=1$), we have from Eq. (5):

$$(x,y,2,i,j,k) \longleftrightarrow (x-j\bar{i}+1,y-ji+1,0,\bar{i},j',k'). \tag{9}$$

Substituting $j=0$ into Eq. (9) tells us that all $j=0$ vertices of a $K_{4,4}$ cell are connected to the $K_{4,4}$ cells in the $z=0$ layer, but with x and y coordinates *both* shifted by 1:

$$(x, y, 2) \longleftrightarrow (x+1, y+1, 0), \text{ for}, j = 0. \tag{10}$$

Substituting $j = 1$ into Eq. (9) tells us that all $j = 1$ vertices are connected in the following way:

$$(x, y, 2) \longleftrightarrow (x+i, y+\bar{i}, 0), \text{ for}, j = 1. \tag{11}$$

We now explicitly write down the 4 rules which lead to the grouping scheme depicted in Figure 2b (these rules are different depending on whether $z \in \{0, 1\}$ or $z = 2$, so there are actually 8 of them):

$$(x, y, z, 0, 0, k) \leftrightarrow (x, y, z+1, 1, j', k'), \quad z \in \{0, 1\} \tag{12}$$
$$(x, y, z, 1, 0, k) \leftrightarrow (x, y, z+1, 0, j', k'), \quad z \in \{0, 1\} \tag{13}$$
$$(x, y, z, 0, 1, k) \leftrightarrow (x-1, y, z+1, 1, j', k'), \quad z \in \{0, 1\} \tag{14}$$
$$(x, y, z, 1, 1, k) \leftrightarrow (x, y-1, z+1, 0, j', k'), \quad z \in \{0, 1\} \tag{15}$$

$$(x, y, 2, 0, 0, k) \leftrightarrow (x+1, y+1, 0, 1, j', k'), \tag{16}$$
$$(x, y, 2, 1, 0, k) \leftrightarrow (x+1, y+1, 0, 0, j', k'), \tag{17}$$
$$(x, y, 2, 0, 1, k) \leftrightarrow (x, y+1, 0, 0, j', k'), \tag{18}$$
$$(x, y, 2, 1, 1, k) \leftrightarrow (x+1, y, 0, 1, j', k'). \tag{19}$$

II. COMPARISON TO CHIMERA

A. Degree of the vertices

If we look for example at the cell at position $(x, yz) = (1, 1, 1)$, we can find that vertices have a degree of ***fifteen:*** The 6 grey edges that would regularly be in Chimera (4 to form the $K_{4,4}$ cell and 2 to connect to $K_{4,4}$ cells above/left and below/right), then there is 1 Pegasus edge added *within* one $K_{4,4}$ cell according to Eq. (4), then we have 8 more Pegasus edges for connecting $K_{4,4}$ cells of different Chimera layers z, from Eqs. (12)-(15) (or from Eqs. (16)-(19) if we were on a $z = 2$ cell). Therefore the degree (which is 15) has increased by a factor of 2.5 when compared to the degree of Chimera (which is 6).

In Fig. 2a, the cells at the boundary, such as at position $(x, y, z) = (0, 0, 0)$, do not show a degree of 15, just as the cells of Chimera at the very edge would not show a degree of 6.

B. Non-planarity

We note that certain binary optimization problems forming planar graphs can be solved on a classical computer with a number of operations that scales polynomially with the number of binary variables, with the blossom algorithm [2]. Therefore it is important that the qubits of a quantum annealer are connected by a non-planar graph. The $K_{4,4}$ cells of Chimera are already sufficient to make all commercial D-Wave annealers non-planar. However, if each $K_{4,4}$ cell of a Chimera's physical qubits were to encode just one logical qubit (in for example, an extreme case of minor embedding), then Chimera would be planar. While all added edges in Pegasus that connect different Chimera layers are of the form $K_{2,4}$, which itself is planar; these $K_{2,4}$ edges connect cells of different planes of chimeras in a non-planar way, such that even if each cell were to represent one logical qubit, *these logical qubits would still form a non-planar graph in Pegasus.* This should expand the number of binary optimization problems that cannot with any known algorithm be solved on a classical computer with a number of operations polynomially scaling with the number of binary variables, that can potentially be embedded onto a D-Wave annealer.

C. Embedding

We have written an entire paper on the minor-embedding of quadratization gadgets onto Chimera and Pegasus. One highlight of that work is the fact that *all* quadratization gadgets for single cubic terms which require one auxiliary qubit, can be embedded onto Pegasus with no further auxiliary qubits because Pegasus contains K_4,which means that all three logical qubits and the auxiliary qubit can be connected in any way, without any minor-embedding. We refer the reader to that paper for more thorough details about the advantage of Pegasus over Chimera for the minor-embedding of quadratization gadgets.

III. OPEN SOURCE CODE FOR GENERATION OF PEGASUS FIGURES

All figures in this publication and its Supplemental Material can be generated (in vector graphic form) using our open source and customizable code which should be cited as in Ref. [3]. The user can choose which type of Pegasus graph; the dimensions X and Y; the edge colors and widths, the vertex colors and widths; among other things (see the manual to Ref. [3] for details).

ACKNOWLEDGMENTS

We gratefully thank Kelly Boothby of D-Wave for her time spent in verifying the correctness of our algorithm. NC was funded by EPSRC (Project: EP/S00114X); SS by NRDIF (Project: NKFIH-K120569 and 2017-1.2.1-NKP-2017-00001) and by the Hungarian Academy of Sciences for a János Bolyai Research Scholarship within the "Lendület" Program.

REFERENCES

[1] H. Neven, V. S. Denchev, M. Drew-Brook, J. Zhang, W. G. Macready, and G. Rose, *NIPS 2009 Demonstration: Binary Classification using Hardware Implementation of Quantum Annealing*, Tech. Rep. (2009).

[2] J. Edmonds, Canadian Journal of Mathematics **17**, 449 (1965); *Journal of Research of the National Bureau of Standards-B.*, Tech. Rep. 2 (1965).

[3] S. Szalay, N. Dattani, and N. Chancellor, (2018), https://github.com/HPQC-LABS/PegasusDraw, DOI: 10.5281/zenodo1953876.

Figure 3: Two more ways to visualize Pegasus, where the $K_{4,4}$ cells are simply drawn using the “diamond” and “tilted classic” depictions instead of the “upper-triangle” one.