A COMPUTER-AIDED EXAMINATION OF SOME CLASSES OF HYPERCUBE-LIKE SUPER FAULT-TOLERANT HAMILTONIAN NETWORKS

EDDIE CHENG, DANIEL STEFFY AND AARON ZENG

ABSTRACT. An r-regular graph $G = (V, E)$ is *k-fault-tolerant Hamiltonian* if $G - F$ is Hamiltonian for every set $F \subseteq V \cup E$ of faulty elements where $|F| \leq k$. Similarly, G is *k-fault-tolerant Hamiltonian connected* if $G - F$ is Hamiltonian connected for every set $F \subseteq V \cup E$ of faulty elements with $|F| \leq k$. If an r-regular graph is both $(r-2)$-fault-tolerant Hamiltonian and $(r-3)$-fault-tolerant Hamiltonian connected, we say that it is *super fault-tolerant Hamiltonian.* In this paper, we discuss different classes $\mathcal{G}_r$ of graphs which are recursively defined by $\mathcal{G}_r = \cup_{i,j}(G_i \oplus G_j)$ for all $G_i, G_j \in \mathcal{G}_{r-1}$; $G_0 \oplus G_1$ refers to the set of graphs obtained by joining each vertex in G_0 to exactly one in G_1 and vice versa. It was proven by Chen, Tsai, Hsu, and Tan in 2004 that if G_0 and G_1 are super fault-tolerant Hamiltonian r-regular graphs where $r \geq 5$, then every graph G in $G_0 \oplus G_1$ is an $(r+1)$-regular graph that is also super fault-tolerant Hamiltonian. Using a computer program, we demonstrate the super fault-tolerant Hamiltonicity of several classes of graphs.

Keywords: Hamiltonian, Hamiltonian-connected, classes of recursively defined graphs.

1. Introduction

Interconnection networks are important in parallel computing. For example, the IBM Blue Gene Project uses a 3-dimensional torus to connect 65,536 nodes. The first major class is the classical n-cubes. The n-cube is defined as having the vertex set of binary strings of length n. Two vertices are adjacent if their strings differ in exactly 1 bit. However, they can also be defined recursively. Interconnection networks can be represented by graphs where vertices are computer processors and edges are links between processors. In this paper, we use standard graph theory terminologies and notations. Since our interests are interconnection networks and they are usually regular, we will only consider regular graphs in this paper. A graph is *Hamiltonian* if it has a cycle containing every vertex, and a graph is

Aaron Zeng started this project at the 2013 Oakland University Summer Mathematics Institute.

Hamiltonian-connected if for every pair of distinct vertices u and v, there is a path between u and v containing every vertex. We note that if a graph has at least three vertices, then a Hamiltonian-connected graph is Hamiltonian. The properties of Hamiltonicity and Hamiltonian connectedness are important topics in the study of interconnection networks and have been studied extensively. Since processors and links between them may fail in a computer system, many researchers focus on the properties of interconnection networks when vertices and/or edges are deleted (representing processors and/or links failure). In terms of Hamiltonicity, we consider the following: An r-regular graph $G = (V, E)$ is *k-fault-tolerant Hamiltonian* if $G - F$ is Hamiltonian for every set $F \subseteq V \cup E$ of faulty elements where $|F| \leq k$. Of course, the objective is to find the best (maximum) k. This is an $\mathcal{N}P$-hard problem. There is an obvious bound, that is , $k \leq r - 2$. An r-regular graph is *maximally fault-tolerant Hamiltonian* if it is $(r-2)$-fault-tolerant Hamiltonian. Similarly, G is is *k-fault-tolerant Hamiltonian-connected* if $G - F$ is Hamiltonian-connected for every set $F \subseteq V \cup E$ of faulty elements where $|F| \leq k$. As before, we want the best k and that this is an $\mathcal{N}P$-hard problem. In this case, the obvious bound is $k \leq r - 3$, and G is *maximally fault-tolerant Hamiltonian-connected* if it is $(r-2)$-fault-tolerant Hamiltonian-connected. Moreover G is *super fault-tolerant Hamiltonian* if it is both maximally fault-tolerant Hamiltonian and maximally fault-tolerant Hamiltonian-connected. In this paper, we provide several classes of super fault-tolerant Hamiltonian graphs based on a known result and via a computer package.

The class of *bijective connection networks (BC networks)* is defined recursively as follows: Let $\mathcal{H}_1 = \{K_2\}$ and for $i \geq 2$, let $\mathcal{H}_i$ be the set of all graphs that can be constructed by taking two (possibly the same) elements $H_1 = (V_1, E_1)$ and $H_2 = (V_2, E_2)$ from $\mathcal{H}_{i-1}$ (if we take the same element, we will assume they are two different copies and so $V_1 \cap V_2$ remains empty) with a bijection $f : V_1 \longrightarrow V_2$ to form the graph $H = (V_1 \cup V_2, E_1 \cup E_2 \cup M)$ where $M = \{(v, f(v)) : v \in V_1\}$. This class of networks include a number of networks such as hypercubes, crossed cubes and twisted cubes. We note that BC networks are triangle-free, that is, they do not contain a K_3 as a subgraph. The class of *matching composition networks (MC networks or simply MCN's)* is defined as follows: Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two graphs with $|V_1| = |V_2|$, $f : V_1 \longrightarrow V_2$ be a bijection and $M = \{(v, f(v)) : v \in V_1\}$; then construct the graph $G = (V_1 \cup V_2, E_1 \cup E_2 \cup M)$. We remark that this is called matching composition network as M is a perfect matching in G. Often G is denoted by $G(G_1, G_2, M)$. Moreover, we let $G_1 \oplus G_2$ denote the set of graphs of the form $G(G_1, G_2, M)$ for all possible M. Of course, one can replace $\mathcal{H}_1$ by any set of graphs with the same number of vertices and the same regularity. Let $\mathcal{R}$ be a set of graphs with such properties. Then *$\mathcal{R}$-composition* *graphs* can be defined recursively as follows: Let $\mathcal{R}_1 = \mathcal{R}$ and for $i \geq 2$, let $\mathcal{R}_i = \cup_{H_1,H_2 \in \mathcal{R}_{i-1}} (H_1 \oplus H_2)$.

2. A known result and motivation

The following result of Chen, Tsai, Hsu and Tan motivated us for the problem in this short note.

Theorem 2.1 ([1])**.** *Let $R \geq 5$. Let G_0 and G_1 be r-regular super fault-tolerant Hamiltonian graphs with the same number of vertices. Then every graph in $G_0 \oplus G_1$ is $(r+1)$-regular and super fault-tolerant Hamiltonian.*

From Theorem 2.1, we can see that $\mathcal{R}$-composition graphs are natural classes of graphs to consider. This motivated us to constructed classes of $\mathcal{R}$-composition graphs with appropriate $\mathcal{R}$'s. Given that we want the graphs to be Hamiltonian-connected, the graphs in $\mathcal{R}$ cannot be bipartite. In the next section, we present several such classes of graphs.

3. Classes of super fault-tolerant Hamiltonian graphs

The most basic class can be constructed using $\mathcal{R} = \{K_6\}$. Clearly K_6 is 5-regular and super fault-tolerant Hamiltonian. Hence we can apply Theorem 2.1 immediately to conclude that $\{K_6\}$-composition graphs are super fault-tolerant Hamiltonian. In fact, this applies to any K_n where $n \geq 6$. More interesting examples can be obtained by starting with K_5 or K_4. Let's first consider K_5. It is 4-regular and super fault-tolerant Hamiltonian. However, we need the graph to be at least 5-regular to use Theorem 2.1. Nevertheless we can compute $K_5 \oplus K_5$, which contains exactly one graph, namely, $K_2 \square K_5$, that is, the Cartesian product of K_2 and K_5. Clearly $K_2 \square K_5$ is 5-regular and our computer program verifies that $K_2 \square K_5$ is super fault-tolerant Hamiltonian. Thus we may apply Theorem 2.1 to conclude that $\{K_5\}$-composition graphs are super fault-tolerant Hamiltonian.

We now start with K_4, It is 4-regular and super fault-tolerant Hamiltonian. However, we need the graph to be at least 5-regular to use Theorem 2.1. Nevertheless we can compute $K_5 \oplus K_5$, which contains exactly one graph, namely, $K_2 \square K_4$. We apply $\oplus$ one more time and consider $(K_2 \square K_4) \oplus (K_2 \square K_4)$. It turns out that there are 38 non-isomporphic graphs in this set. Our computer program verifies that $K_2 \square K_4$ and each of the 38 graphs in $(K_2 \square K_4) \oplus (K_2 \square K_4)$ is super fault-tolerant Hamiltonian. Thus we may apply Theorem 2.1 to conclude that $\{K_4\}$-composition graphs are super fault-tolerant Hamiltonian. We summarize our discussion in the next result.

Theorem 3.1. *Let $n \geq 4$. Then $\{K_n\}$-composition graphs are super fault-tolerant Hamiltonian.*

We tested another class of graph with the starting graph $K_6 - \mathbf{pm}$, that is K_6 with a perfect matching deleted. It is a 4-regular graph. We computed $(K_6 - \mathbf{pm}) \oplus (K_6 - \mathbf{pm})$, which consists of three graphs and our computer programs verifies that all these graphs are super fault-tolerant Hamiltonian. Therefore we have the following result.

Theorem 3.2. *Let $n \geq 4$. Then $\{K_6 - \mathbf{pm}\}$-composition graphs are super fault-tolerant Hamiltonian.*

As we have indicated, computer programs were used in the computation. These programs are written in Python [3] and make use of library functions in the `NetworkX` package [2], which provides algorithms for graph theoretical computation.

One of the issues with generating these graphs by computer is the possibility of isomorphism. For example, $(K_2 \Box K_4) \oplus (K_2 \Box K_4)$ contains 38 non-isomorphic graphs. However, naïvely generated, the set may contain $8! = 40320$ graphs with many duplicates. Therefore, it is of utmost importance to be able to eliminate isomorphs so that the next iteration can be computed in a reasonable amount of time. The `NetworkX` library provides the method `is_isomorphic()` to test whether two graphs are isomorphic to one another. This lets us formulate an algorithm for finding the non-isomorphic graphs in a set of generated graphs. As they are generated, we keep a list of graphs that have already been seen. For each new graph, we test it against each of the graphs in the list to see whether or not it is isomorphic to a graph we have already seen. If so, we ignore the new graph and generate the next one, but if not, we add the new graph to the list and continue.

Let us consider one of the 38 graphs in $(K_2 \Box K_4) \oplus (K_2 \Box K_4)$. It is 5-regular on 16 vertices. We need to delete 3 vertices in all possible way to test Hamiltonicity in the resulting graph and we need to delete 2 vertices in all possible way to test Hamiltonian connectedness in the resulting graph. There are many graphs to test. Trying to eliminate duplicates takes too long. For graph 23 in the list, it took 27706.80 seconds to eliminate duplicates and 25.72 seconds to test whether these graphs are Hamiltonian and Hamiltonian connected. That's over 7 hours. However, If we do not eliminate duplicates and just test every possibility, it took only a mere 74.37 seconds in total. As a side note, this graph has $23,600$ possible "faulty" graphs when generated naïvely, which was reduced to $11,778$ non-isomorphic "faulty" graphs.

Computation time was recorded for the test of fault-tolerant Hamiltonicity and the test of fault-tolerant Hamiltonian connectedness. The time taken is shown below, in seconds:

Class of graphs	Fault-tol. Ham.	Fault-tol. Ham. con.
$(K_2 \square K_4) \oplus (K_2 \square K_4)$	2547.58	6179.08
$K_5 \oplus K_5$	14.45	1.55
$(K_6 - \mathbf{pm}) \oplus (K_6 - \mathbf{pm})$	770.12	25.12

4. Conclusion

In this short note, we provided several classes of natural super fault-tolerant Hamiltonian graphs. As a second objective, we would advocate the package `NetworkX` due to its ease of use.

References

[1] Y.-C. Chen, C.-H. Tsai, L.-H. Hsu, and J. J. Tan. On some super fault-tolerant hamiltonian graphs. *Applied Mathematics and Computation*, 148:729–741, 2004.

[2] A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference (SciPy2008)*, pages 11–15, Pasadena, CA USA, Aug. 2008.

[3] H. P. Langtangen. *A Primer on Scientific Programming with Python*. Springer, Dordrecht, The Netherlands, July 2009.

Eddie Cheng, Department of Mathematics and Statistics, Oakland University, Rochester, MI 48309, USA. Email: echeng@oakland.edu.

Daniel Steffy, Department of Mathematics and Statistics, Oakland University, Rochester, MI 48309, USA. Email: steffy@oakland.edu

Aaron Zeng, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA. Email: zeng.aaron.l@gmail.com

FIGURE 1. The 38 non-isomorphic graphs in $(K_2 \square K_4) \oplus (K_2 \square K_4)$

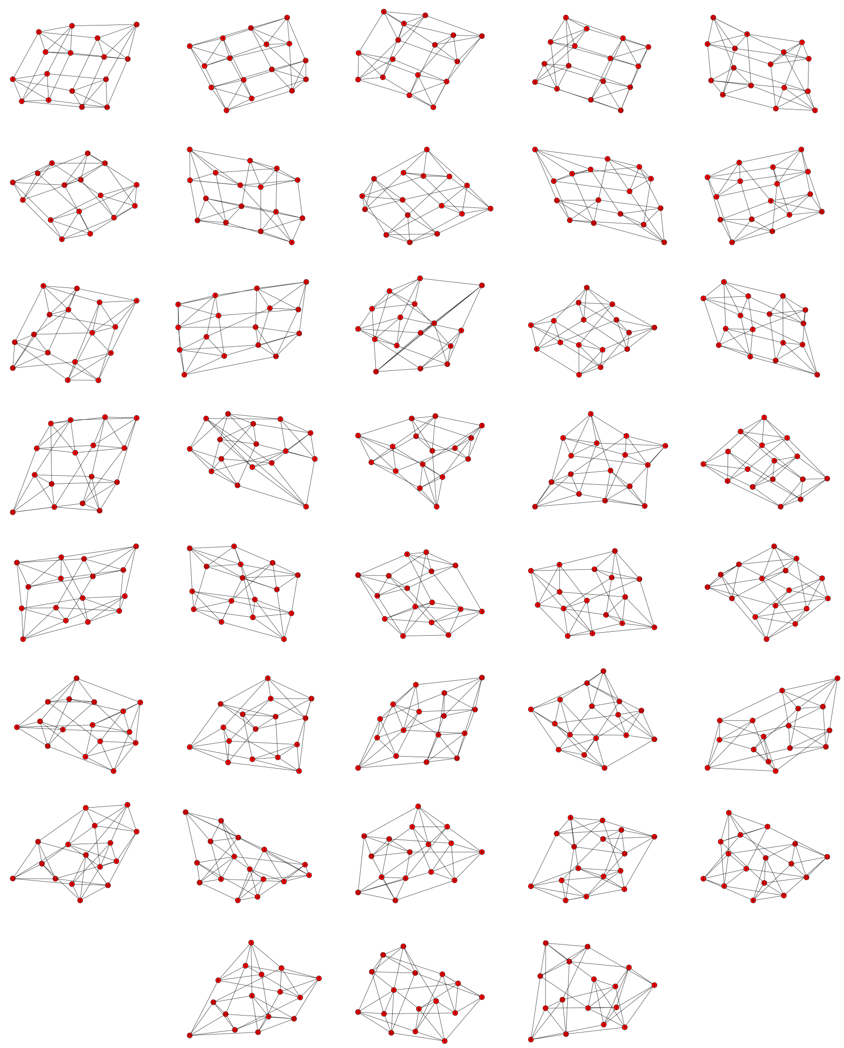

FIGURE 2. The three non-isomorphic graphs in $(K_6 - \mathbf{pm}) \oplus (K_6 - \mathbf{pm})$