Network Dilemma

A Survey of 6 Different Network Configurations
MATH 182, Dr. Manami Roy

Jackson Eshbaugh, Amy Fabara, Tyler Salmon, Billy Zou

May 3, 2024

Abstract

This study surveys six different network configurations—complete, complete bipartite, hypercubes, trees, cyclic, and wheel—to evaluate their suitability for a network of 32 nodes. Through comprehensive analysis of each configuration's cost, speed, reliabilty, and extensibility, we explore the pros and cons of each layout. Our evaluation criteria include network connectivity, vertex and edge reliability, the existence of Hamilton cycles and Euler tours, and the feasibility of network expansion. Among the configurations considered, the complete bipartite, the hypercube, and the wheel layouts emerged as the most suitable, depending on specific needs. To conclude, we provide recommendations tailored to different organizational requirements, providing a nuanced understanding of how each configuration could serve specific network infrastructure needs effectively.

1 Introduction

Networks are at the basis of some of the most simple and important things we do on our devices, daily. The internet, other intranets, and even local networks play a very significant role in computing. In fact, setting up a computer network is no simple task! Networks are complex, and every setup has its pros and cons. Here, we consider setting up a network with 32 computers in six different configurations. We begin with a discussion of graphs in general, then hypercubes and their properties. After this, we lay out our criteria with which we will judge each network setup. Then, we evaluate each using the criteria we discussed. Finally, we make a judgement call to declare the best network configurations out of the six we discuss here.

1.1 What is a Graph?

Before we begin our discussion of these different network layouts, it would behove us to define exactly what a graph is.

Definition 1.1 (graph)**.** A graph $G = (V, E)$ consists of a set of vertices (or nodes or points) and a set of edges which serve as connections between the vertices.

Each of the network configurations we explore below are a type of graph, and each have different advantages and disadvantages when implemented as network configurations. Now, before we begin our analysis, a word on hypercubes is due.

1.2 On Hypercubes

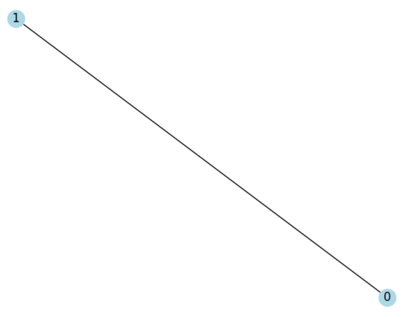
Hypercubes or n-cubes, symbolized by Q_n, where n represents the dimension of the cube, are one example of a possible network configuration. There are some interesting properties that n-cubes have that we explore here.

Firstly, we'll introduce an interesting connection between hypercubes and binary strings. The n-cube Q_n corresponds with a binary string of length n, such that each vertex represents a possible string.

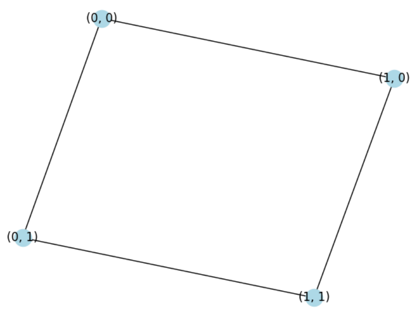
In Figure 1, notice that all possible combinations of a binary string of length n are encoded into the hypercube. This idea allows us to derive some general information about Q_n.

Theorem 1.1. *In any hypercube Q_n, the number of vertices is equal to the number of possible binary strings of length n, which is 2^n.*

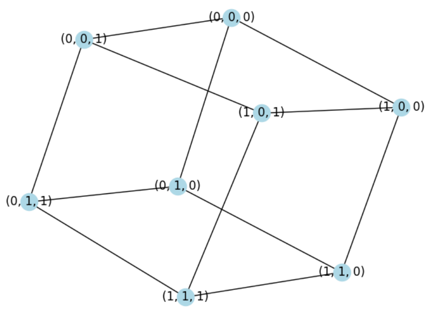
Definition 1.2 (Degree of a vertex)**.** The degree of a vertex v, denoted $deg(v)$, is a measure of the number of connections to the vertex. For example, in Figure 2, $deg(a) = 1$, but $deg(b) = 2$.



(a) Q_1



(b) Q_2



(c) Q_3

Figure 1: N-cubes for $1 \leq n \leq 3$

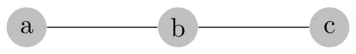


Figure 2: In the above graph, $deg(a) = deg(c) = 1$, but $deg(b) = 2$.

Theorem 1.2. *For any hypercube Q_n, the number of edges is*

$$n2^{n-1}$$

[1]

Proof. Notice that the degree of each vertex in an n-cube is equal to n. If we count each vertex's degree, we have

$$e = n \cdot 2^n.$$

However, simply using this quantity as the number of edges is incorrect—we've counted each edge twice. The correct number of edges is

$$e = \frac{n \cdot 2^n}{2} = n2^{n-1}$$

□

In terms of Hamilton cycles, we can find them in n-cubes, but there's a special kind of binary we use to derive them called Gray code.

Definition 1.3 (Gray code)**.** Gray code is a way to write binary strings (of length 2^n) in a cyclical format, such that each string only differs from the next by 1 digit. [8]

Using Gray code, we can construct a Hamilton cycle (see Definition 1.4):

1. Start with a 1-bit Gray code sequence:

$$0, 1$$

2. Until length n (the dimension of the hypercube) is reached:

   (a) Reflect the previous sequence, i.e.

$$0, 1, 1, 0$$

(b) Prepend the original sequence with 0, and the reflected sequence with 1, e.g.

$$00, 01, 11, 10$$

3. Each string in the sequence is mapped to a specific vertex, such that all vertices immediately next to one another differ by exactly one digit. The created sequence of strings is the Hamilton cycle.

We also find that Euler tours are present in hypercubes, because all of their vertices are of even degree.

Theorem 1.3. *For every even n, Q_n has an Euler tour (See Definition 1.6) because for every even n, all vertices in Q_n are of even degree. By Theorem 1.5, all even n-cubes have an Euler tour.*

Finally, another interesting thing about hypercubes is the simplicity of their diameters.

Theorem 1.4. *The diameter of a hypercube Q_n is n [3].*

In fact, if we take two points labeled using Gray code, we can find the distance between these points by simply counting the number of different digits in the binary strings—that number is the distance. Theorem 1.4 is trivially proven when we think about the possibilities of the vertex labeling. The two furthest points from each other will be labelled by n zeros and n ones. The number of different digits is, of course, n.

1.3 Design Considerations

Before we begin evaluating the possible setups, we need to list the criteria we will use to judge these configurations. Each prospective network setup will be evaluated using the following criteria:

- Cost — Network connections are expensive. We try to optimize the number of edges as best possible.
- Speed — We measure speed in network diameter, which is the furthest apart two vertices can be.
- Edge Reliability — We determine edge reliability by considering the scenario where one edge (connection) stops functioning. In this scenario, we find how many vertices are still accessible.
- Vertex Reliability — We determine vertex reliability by considering the scenario where one vertex (computer) stops functioning. In this scenario, we find how many vertices are still accessible.
- Hamilton Cycles — We check for a Hamilton cycle. This allows the client to send a message exactly once to every computer on the network.

  Definition 1.4 (Hamilton cycle)**.** A cycle that uses every vertex in a graph exactly once is called a Hamilton cycle [2].

- Euler tours — We check if the network is Eulerian or not. An Euler tour would allow a message to be sent through every connection once.

  Definition 1.5 (Euler trail)**.** An Euler trail is a trail in which every pair of adjacent vertices appear consecutively. That is, every edge is used exactly once [5].

  Definition 1.6 (Euler tour)**.** An Euler tour is a closed Euler trail [5].

  Definition 1.7 (closed trail)**.** A trail is closed if it begins and ends with the same vertex [5].

  Theorem 1.5. *A connected graph (or multigraph, with or without loops) has an Euler tour if and only if every vertex in the graph has even degree [5].*

- Extensibility — We determine what it would require to add another 32 computers to this network structure.

2 Potential Configurations

2.1 Complete Graph K_{32}

Definition 2.1 (Complete graph)**.** A complete graph (denoted as K_n, where n is the number of vertices in the graph) is a graph in which every vertex is connected to every other vertex.

Because of its structure, K_{32} is very reliable but very expensive.

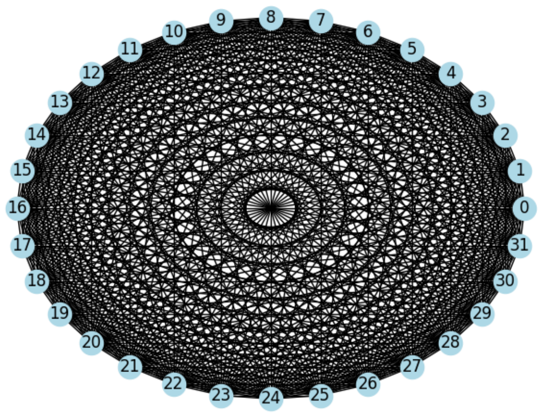


Figure 3: The complete graph K_{32}

2.1.1 Network Connections

Speaking of cost, we calculate the number of edges in complete graphs using a combination.

Theorem 2.1. *The number of edges in a complete graph K_n is*

$$\binom{n}{2}$$

[4].

By Theorem 2.1, the graph $K_{n=32}$ has

$$\binom{32}{2} = 496$$

edges.

2.1.2 Diameter

With all those edges, every point is connected to every other point, so, the graph K_{32} has a diameter of 1.

2.1.3 Edge Reliability

Suppose one connection were to be broken. Even so, since every computer is connected to every other computer, all computers would be completely connected.

2.1.4 Vertex Reliability

Suppose one computer were to go down for maintenance. Still, all of the other 31 computers would remain completely connected to the network, because they are all connected to each other directly.

2.1.5 Hamilton Cycles

Theorem 2.2. *A complete graph K_n always has at least one Hamilton cycle [6].*

Since all complete graphs have at least one Hamilton cycle, the graph we're considering has a Hamilton cycle.

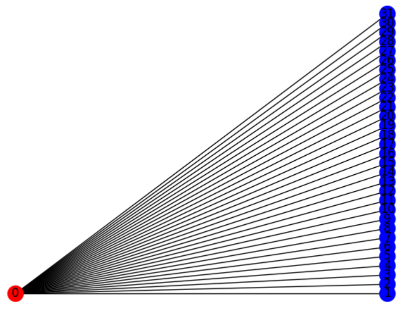
2.1.6 Euler Tours

Consider Theorem 1.5. For a complete graph K_n, an important corollary arises.

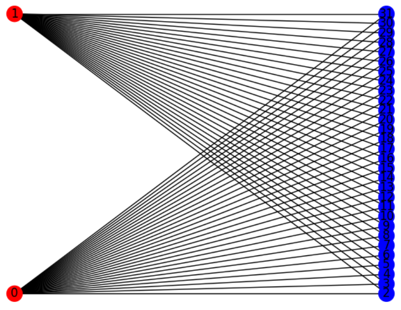
Corollary 2.2.1. *Let K_n be a complete graph with $n \geq 3$ vertices. Then, if n is even, K_n does not have an Euler tour, but if n is odd, K_n does have an Euler tour.*

Proof. Consider the graph K_n. Note that in a complete graph, each vertex has degree $n - 1$, since each vertex is connected to every other vertex except itself. Proceed by cases.

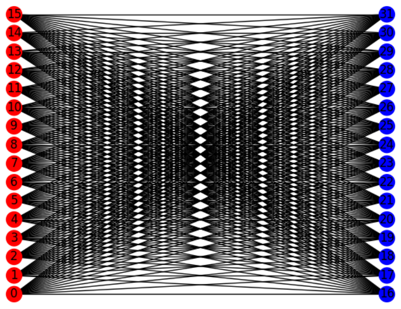
Case 1: n is even. Since the degree of each vertex in K_n is $n - 1$ and n is even, the degree of each vertex in K_n is odd. By Theorem 1.5, K_n does not have an Euler tour.



(a) $K_{31,1}$



(b) $K_{30,2}$



(c) $K_{16,16}$

Figure 4: Different bipartite configurations for $m + n = 32$

Case 2: n is odd. Since the degree of each vertex in K_n is $n - 1$ and n is odd, the degree of each vertex in K_n is even. By Theorem 1.5, K_n has an Euler tour.

Hence, for $n \geq 3$, K_n has an Euler tour only when n is even. □

Applying this corollary, we find that K_{32} does not have an Euler tour.

2.1.7 Extensibility

To add another 32 nodes to the network, we simply construct K_{64}. By Theorem 2.1, the number of edges in the complete graph K_{64} is

$$\binom{64}{2} = 2,016.$$

So, this would require an extra $2,016 - 496 = 1,520$ connections.
In a word, K_{32} is ... expensive. Yet, it has great reliability. Perhaps another layout will be more cost-optimized and share in similar reliability statistics.

2.2 Complete Bipartite Graphs $K_{m,n}$ where $m + n = 32$

Recall from Definition 1.1 that a graph G has a set of vertices V. If we consider the complete graph K_n's set of vertices V, we can group the vertices into two subsets, $A \subset V$ and $B \subset V$, where $A \cup B = V$. This is the basic idea behind the complete bipartite graph.

Definition 2.2 (Complete bipartite graph)**.** A complete bipartite graph is a complete graph that is partitioned into two sets of vertices. Each vertex from each set must is connected to each vertex from the other set. A complete bipartite graph is denoted $K_{m,n}$, where m and n are the sizes of the two sets of vertices.

For our cases, we'll consider $m + n = 32$. There are three different bipartite graphs we'll look at in this section: $K_{31,1}$, $K_{30,2}$, and $K16, 16$. $K_{31,1}$ and $K_{30,2}$ are master-slave network setups. $K_{31,1}$ has one master, and $K_{30,2}$ has two masters (one acts as a backup).

2.2.1 Network Connections

Theorem 2.3. *For a bipartite graph $K_{m,n}$, the number of edges in K is*

$$m \cdot n$$

since each vertex in each subset will need to be connected to every vertex in the other subset.

Using the above theorem, $K_{31,1}$ has 31 edges, $K_{30,2}$ has 60 edges, and $K_{16,16}$ has 256 edges.

2.2.2 Diameter

Because of the properties of complete bipartite graphs, the diameter of any complete bipartite graph is 2. Hence, $K_{31,1}$, $K_{30,2}$, $K_{16,16}$ all have a diameter of 2.

2.2.3 Edge Reliability

If an edge were to be removed from $K_{31,1}$, the only computer knocked offline would be the one connected to the downed edge in the set of 32. This results in all 31 other machines remaining online. As for $K_{30,2}$, the master has a backup. This means that in the event that connection to the master is knocked offline, the secondary master will take over, allowing connection to all 32 computers on the network to persist. This may result in slightly less efficient speeds than when the network is fully operational. In $K_{16,16}$, similarly, if one connection were to go offline, all computers can remain connected to the network. Any computers that relied on the dead connection can simply re-route through another computer in the opposite set.

2.2.4 Vertex Reliability

If a computer was to malfunction or go offline in a network configured like $K_{31,1}$, there are two possibilities:

- If any of the slave machines go offline, only that computer will be affected.
- If the master goes offline, the whole network is effectively offline.

Because there are two outcomes, we use worst-case analysis. When doing such analysis, only the worst possible outcome is considered, and all decisions are made with these metrics in mind. So for $K_{31,1}$ the worst case computer outage is a whole network outage. When considering $K_{30,2}$, we loose this vulnerability. If the master goes offline, only it is knocked offline. This means that any computer in a $K_{30,2}$ configuration that goes offline will be the only machine affected. A $K_{16,16}$ layout shares this same property—if a machine goes down, other machines can access any machine on the network besides the one which fell offline.

2.2.5 Hamilton Cycles

Theorem 2.4. *A bipartite graph has a Hamilton cycle if and only if* $m = n \ \forall\ m, n \geq 2$.

Proof. Because this is a bi-conditional, we need to prove both directions of the statement.
$\Longrightarrow$ First, suppose $K_{m,n}$ has a Hamilton cycle. This means that $m = n$ because traversal of the graph must occur between partitions. Additionally, a cycle implies that the same edge is never used twice. Thus the trivial case of $K_{1,1}$ does not have a Hamilton cycle.
So, we have $m = n \ \forall\ m, n \geq 2$
Next, we focus on the other side of the conditional.
$\Longrightarrow$ Consider the graph $K_{m,n}$, where $m = n$ and $m, n \geq 2$. If we label each of its vertices based on which set it comes from (denoted here as either M or N), we have

$$M_1, N_1, M_2, N_2, M_3, N_3, ..., M_n, N_n, M_1$$

which is a Hamilton cycle.
Hence, a bipartite graph has a Hamilton cycle $\iff m = n \ \forall\ m, n \geq 2$. □

Using Theorem 2.4, $K_{31,1}$ and $K_{30,2}$ do not have Hamilton cycles (since $m \neq n$), but $K_{16,16}$ does have such a cycle.

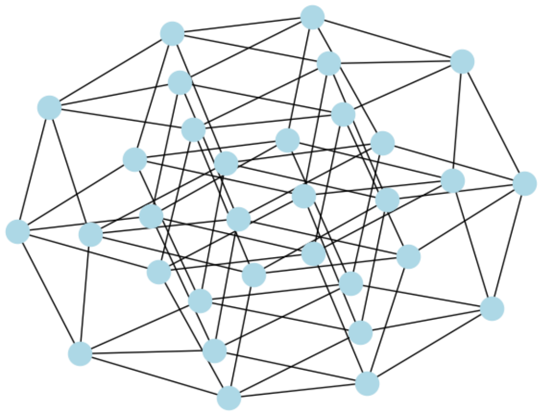


Figure 5: The hypercube Q_5

2.2.6 Euler Tours

We apply Theorem 1.5 to determine the existence of Euler tours.
For $K_{31,1}$, clearly the degree of every vertex that isn't the master is 1. This immediately shows that it does not have an Euler tour. As for $K_{30,2}$, every vertex is of even degree. All the slave vertices have a degree of 2—one connection to each master. The masters are connected to 30 separate computers (but not each other), so they have degree of 30 each. Thus, $K_{30,2}$ has an Euler tour. Finally, $K_{16,16}$ clearly has an Euler tour because the degree of each vertex is 16.

2.2.7 Extensibility

To double the network in a $K_{31,1}$ configuration, only one additional connection is needed per computer—from the new slave to the master, totaling 32 extra connections. In the case of the $K_{30,2}$ network, a connection needs to be made to both master computers from each new slave, totalling 64 additional connections. Finally, for a $K_{16,16}$ layout, we have

$$(16 + 16) \cdot (16 + 16) = 1,024$$

which is $1,024 - 256 = 768$ new connections. So, a complete bipartite layout presents a healthy competitor to K_{32} in $K_{30,2}$, although the other two are either quite unreliable ($K_{31,1}$) or quite expensive ($K_{16,16}$).

2.3 5-Dimensional Hypercube Q_5

We discussed hypercubes at length in Section 1.2, but we now consider Q_5 specifically.

2.3.1 Network Connections

Applying Theorem 1.2, Q_5 has $5 \cdot 2^4 = 5 \cdot 16 = 80$ edges.

2.3.2 Diameter

Applying Theorem 1.4, Q_5 has a diameter of 5.

2.3.3 Edge Reliability

If a connection is down in this configuration, messages could still traverse the whole network since one node is connected to other five adjacent nodes. The messages would be rerouted to other edges when one of the edges is down, reaching their destination (even if slower than without a broken connection).

2.3.4 Vertex Reliability

If a computer falls offline from a Q_5 network configuration, it will be the only machine impacted, because there are multiple paths to the same destination that don't utilize all the same nodes.

2.3.5 Hamilton Cycles

Using Gray code (see Definition 1.3), we construct a string to traverse through the hypercube Q_5:
$n = 1$

$$0, 1$$

$n = 2$

$$00, 01, 11, 10$$

$n = 3$

$$000, 001, 011, 010, 101, 111, 110, 100$$

$n = 4$

$$0000, 0001, 0011, 0010, 0101, 0111, 0110, 0100, 1001, 1011, 1111, 1101, 1010, 1110, 1100, 1000$$

$n = 5$

$$00000, 00001, 00011, 00010, 00101, 00111, 00110, 00100, 01001, 01011, 01111, 01101, 01010, 01110, 01100, 01000,$$

$$10001, 10011, 10111, 10101, 11011, 11111, 11101, 11001, 10010, 10110, 11110, 11010, 10100, 11100, 11000, 10000$$

Since each binary string only differs by one digit, when mapped onto an n-cube where all vertices immediately next to one another differ by exactly one digit this list of binary strings represents a Hamilton cycle through the hypercube Q_5.

2.3.6 Euler Tours

According to Theorem 1.3, Q_n has an Euler tour when n is even. Since $n = 5$, Q_n doesn't have an Euler tour.

2.3.7 Extensibility

To keep the same structure while expanding the network, a second Q_5 unit is constructed and this and the original are connected to each other. Another Q_5 is 80 connections and it takes 32 to connect the two networks together. So, it takes a total of $80 + 32 = 112$ connections to double this network's capacity. Hypercubes make very efficient and reliable networks, but the cost to implement one may not always be worth it, depending on use cases.

2.4 Tree Configuration

Definition 2.3 (Tree)**.** A tree is a connected, simple graph where each node is connected to only one parent node; the only node without a parent is the root node (the node on the layer with only itself). Leaf nodes are nodes with no children. There is only one path to each node in a tree layout.

We consider 2- (binary), 3- (ternary), and 4-degree tree configurations below.

2.4.1 Network Connections

Theorem 2.5. *A tree with n vertices has $n - 1$ edges [7].*

Applying this theorem, all three trees we consider will have $32 - 1 = 31$ edges.

2.4.2 Diameter

The diameter of a tree is equal to the height of that tree (since the height of the tree gives the number of steps it takes to get from the root node to the furthest leaf node.

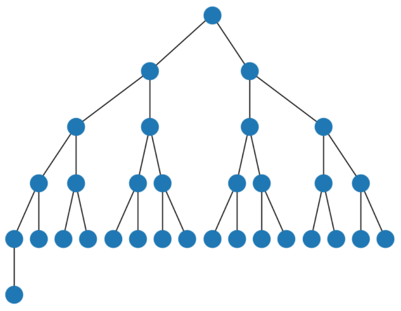
Theorem 2.6. *The height h of a d-ary tree (tree of degree d) is $h = \lfloor log_d(n) \rceil$.*

Proof. Each level of a d-ary tree contributes d^k nodes (where k starts at 0 and ends at h). This implies

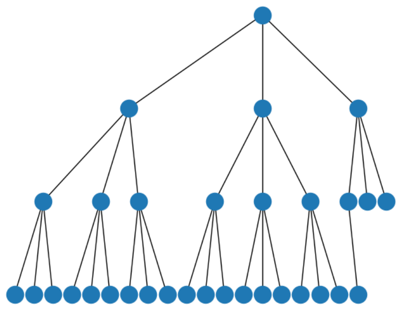
$$n = \sum_{k=0}^{h} d^k = 1 + d + d^2 + ... + d^h.$$

Using the closed form of a geometric sum, we have

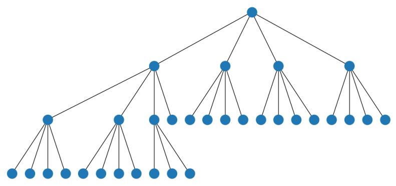
$$n = \frac{d^{h+1} - 1}{d - 1}$$



(a) Binary (2-degree) tree



(b) Ternary (3-degree) tree



(c) 4-degree tree

Figure 6: Different tree configurations with $n = 32$

$$\implies d^{h+1} = n(d-1) + 1$$

Now, taking the logarithm of both sides, we have

$$h + 1 = log_d(n(d-1)+1) \implies h = log_d(n(d-1)+1) - 1$$

We can simplify the logarithm as follows

$$h \approx log_d(n)$$

Since the height must be an integer, we round the right side to the nearest integer.

$$h = \lfloor log_d(n) \rceil$$

Hence, the height h of a d-ary tree (tree of degree d) is $h = \lfloor log_d(n) \rceil$. □

Applying Theorem 2.6, we have $h = log_2(32) = 5$ for a binary tree, $h = \lfloor log_3(32) \rceil = 3$ for a ternary tree, $h = \lfloor log_4(32) \rceil = 3$ for a 4-degree tree.

2.4.3 Edge Reliability

For any tree, the worst case of an edge failure is when an edge connected to the root fails. This partitions the tree into two: a root-connected unit and a separate cluster. Those machines we count as "online" in this instance are the root-connected ones. For a binary tree, notice only 16 nodes (half of the original number) would be accessible after such an event. As for a ternary tree, the worst case will be one of the edges connected to the larger number of nodes. This would leave 19 nodes accessible. Finally, a 4-degree tree will have only 16 nodes operational after the removal of its most vital edge.

2.4.4 Vertex Reliability

When using worst-case analysis for a computer failure in these tree configurations, we always assume the root fails—no computer could impact the network more than the root if it were to fail. Notice that the network is partitioned when the root fails. Each section is able to communicate freely internally, but any traffic outbound from these clusters cannot achieve its destination because the root is offline. To measure the impact of a (root) computer failing, we count the number of nodes in the smallest cluster that would be created by this failure. The binary tree layout would see the network separated into two partitions, the smaller of which has 15 computers. The ternary tree layout would see the network separated into three partitions, the smallest of which has 5 computers. The 4-degree tree layout would see the network separated into four partitions, the smallest of which has 5 computers.

2.4.5 Hamilton Cycles

Trees do not have Hamilton cycles since tree graphs don't have any cycles; there is no way to return to the root node of a tree while touching every node in the tree without re-using edges of the tree.

2.4.6 Euler Tours

Similarly, by Theorem 1.5, trees do not have Euler tours because any leaves of a tree are of degree 1.

2.4.7 Extensibility

To expand will always require 32 more edges, because of the definition of a tree graph—it takes only one extra connection per node, always.
Although relatively cheap, tree graphs do not enjoy either a Hamilton cycle or an Euler tour, which could be a major downside when considering possible network configurations.

2.5 Cyclic Configuration C_{32}

Definition 2.4 (Cyclic graph)**.** A cyclic graph (denoted C_n) is a graph where every vertex has a degree of 2. They are connected in a cycle, as the term suggests.

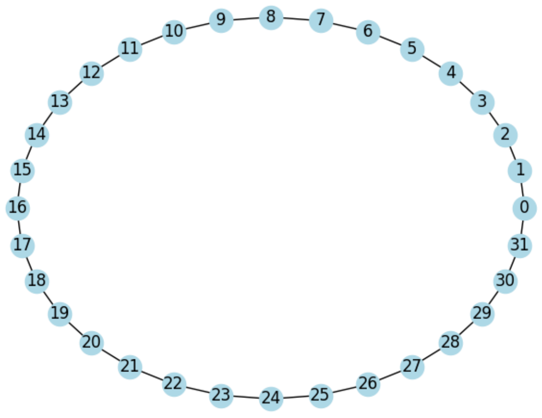


Figure 7: The graph C_{32}

2.5.1 Network Connections

Theorem 2.7. *The number of edges in a cyclic graph C_n is equal to n, because each node needs to link to the next node, and the last node links back to the first node.*

Using Theorem 2.7, the graph C_{32} has 32 edges.

2.5.2 Diameter

Theorem 2.8. *Due to the nature of circles, any length further than halfway around a circle can be traversed quicker in the opposite direction. Hence, the diameter of a cyclic graph C_n is equal to $\frac{n}{2}$.*

Applying Theorem 2.8, C_{32} has a diameter of $\frac{32}{2} = 16$.

2.5.3 Edge Reliability

Consider the case of a non-functional connection in a cyclic network. In this case, all computers are still accessible because the opposite direction can be traversed, and all n nodes are accessible. For C_{32}, all 32 nodes are accessible.

2.5.4 Vertex Reliability

In the case of a computer failure, other computers are still accessible because the opposite direction of the network can be traversed if need be.

2.5.5 Hamilton Cycles

Theorem 2.9. *Any cyclic graph C_n will have a Hamilton cycle because all of its vertices are degree 2 [2].*

This can be seen from the definition of a Hamilton cycle (see Definition 1.4)—a cycle using every vertex can be easily formed by most any cyclic graph. Hence, C_{32} has a Hamilton cycle.

2.5.6 Euler Tours

Every cyclic graph has only vertices of degree 2. Hence, by Theorem 1.5, our configuration C_{32} has an Euler tour.

2.5.7 Extensibility

To expand another 32 machines, 2 connections would need to be broken, then the next 32 machines could be inserted. This gives $n = 64$ for an upgraded network C_{64}, which requires $64 - 32 = 32$ new connections to be formed.
The reliability of the cyclic graph is excellent, and its cost is great too. It's just its speed that makes it less of a contender among the options. Perhaps we can change the layout of the cyclic graph to lower its diameter, while only minimally impacting its cost.

2.6 Wheel Configuration W_{32}

The answer to the previous "perhaps" statement is the wheel graph (W_n).

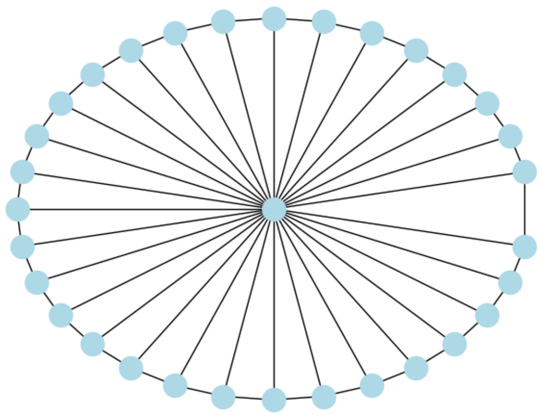


Figure 8: The wheel graph W_{32}

Definition 2.5. A wheel graph, denoted W_n is a cyclic graph with one point in the center that is connected to all other points on the graph. That is, $n - 1$ points form the cyclic part, and the remaining point is connected to all others.

When processing this configuration, we can consider a cyclic configuration C_31 and simply add the central node after this.

2.6.1 Network Connections

According to Theorem 2.7, C_31 has 31 connections. The central node is connected to every other node which yields another 31 connections, totalling 62.

Theorem 2.10. *A wheel graph W_n has $2n - 2$ edges.*

Proof. From Theorem 2.7, we have that a cyclic graph C_n has n edges. Notice that a wheel graph W_n consists of a cyclic graph C_{n-1} and an extra central point, which is connected to every other point. So, from the cyclic graph we have $n - 1$ edges, and we have an additional $n - 1$ edges from the central point (which is connected to every point on the graph besides itself).

$$2(n - 1) = 2n - 2$$

□

2.6.2 Diameter

According to Theorem 2.8 the diameter of any cyclic graph is $\frac{n}{2}$. However, the central node in the wheel configuration cuts this diameter to 2, acting as a bypass out of the cycle and directly to the target node.

2.6.3 Edge Reliability

Notice that in the event of an edge failure, no matter which edge fails, all 32 computers remain connected to the network, though the diameter of the network might slightly increase.

2.6.4 Vertex Reliability

Again, removing the most connected node (the central computer) only definitively disconnects the central computer itself. In the worst case scenario of computer failure, a wheel network becomes a cyclic network.

2.6.5 Hamilton Cycles

It's trivial to see a Hamilton cycle in a wheel configuration: simply traverse the cycle, stopping before returning to the first node. Instead, jump to the central node, then back to the first node.

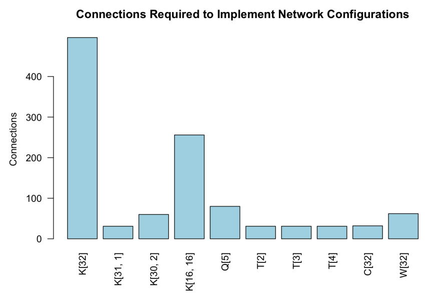
2.6.6 Euler Tours

By Theorem 1.5, there is not an Euler tour in W_{32}, because the degree of each edge node is 3.

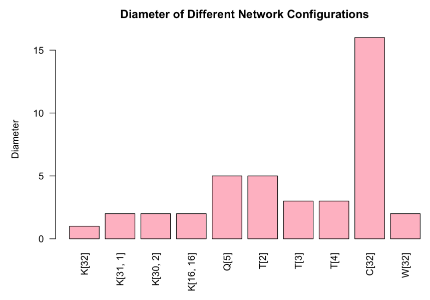
2.6.7 Extensibility

Using Theorem 2.10, W_{64} has $2 \cdot 64 - 2 = 126$ edges. Comparing this with W_{32}, W_{64} has $126 - 62 = 64$ extra edges.
All in all, the wheel is a great option for a network. It takes the best parts of a cyclic graph and innovates on them further, lowering diameter while keeping a great reliability.

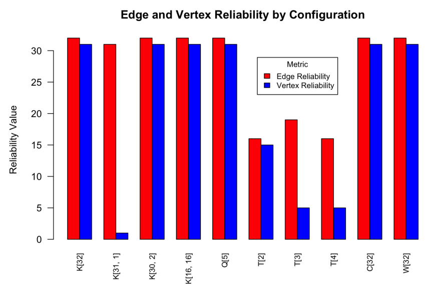
3 Conclusion



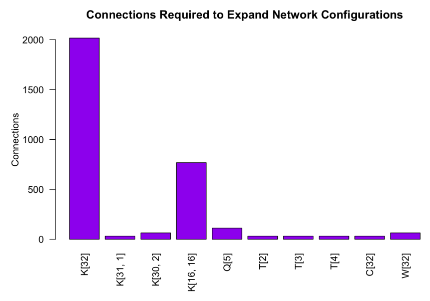
(a) Cost to implement each network



(b) Diameter of each network



(c) Reliability (both edge and vertex) of each network



(d) Cost of expanding each network

Figure 9: Summary of findings

Now that we've discussed different potential network configurations, we make a judgement call and determine the best configuration to implement in this scenario. Firstly, we eliminate K_{32} and $K_{16,16}$ from our choices because they cost far too much compared to all the other configurations (see Figure 9a). We also eliminate the

cyclic graph C_{32} because it is far too slow (Figure 9b) and the $K_{31,1}$ graph because it is very unreliable in the case of a computer failure (Figure 9c. Additionally, all the trees aren't very reliable and they don't contain either a Hamilton cycle or an Euler tour.
This leaves us with $K_{30,2}$, Q_5, and W_{32}. They all have identical reliability, so they are equal on that front. In terms of Hamilton cycles and Euler tours, $K_{30,2}$ has an Euler tour, W_{32} has a Hamilton cycle, and Q_5 has both of them. However, Q_5 is slightly slower and its expansion costs more than $K_{30,2}$ and W_{32}.
Each of these setups have a valid use-case. Firstly, the $K_{30,2}$ is a great setup for a master-slave configuration, which is a great choice for the intended use (parallel processing). Each computer can quickly access another, and even if the master goes offline, the backup master can take over.
Both Q_5 and W_{32} are great choices for a hive mind setup. These networks are also great choices for parallel processing. Depending upon the size of the data being processed, a Q_5 may make more sense than a $W32$. Q_5 offers 4 backup lines in case one line is congested with traffic, while W_{32} offers only 2. Yet, W_{32} is cheaper to implement than Q_5 so depending upon the client's requirements, there is real value in making a decision between these two. And of course, if a C2 (command and control) setup makes most sense for the kind of processing being done, then a $K_{30,2}$ makes the most sense.
In conclusion, the client should choose from either a $K_{30,2}$ layout, a Q_5 layout, or a W_{32} layout based upon their specific needs, such as data size and control placement (which machines—if any—are in charge).

References

[1] Brown University Math Department. *Counting the Edges Of Higher-Dimensional Cubes.* URL: `https://www.math.brown.edu/tbanchof/Beyond3d/chapter4/section05.html`.

[2] David Guichard. *Hamilton Cycles and Paths.* URL: `https://math.libretexts.org/Bookshelves/Combinatorics_and_Discrete_Mathematics/Combinatorics_and_Graph_Theory_(Guichard)/05%3A_Graph_Theory/5.03%3A_Hamilton_Cycles_and_Paths` (visited on 04/21/2024).

[3] Mark "MJD" Dominus. *Answer to "Finding the diameter of a n-cube".* Jan. 2014. URL: `https://math.stackexchange.com/a/625923` (visited on 04/27/2024).

[4] Joy Morris. *Deletion, Complete Graphs, and the Handshaking Lemma.* URL: `https://math.libretexts.org/Bookshelves/Combinatorics_and_Discrete_Mathematics/Combinatorics_(Morris)/03%3A_Graph_Theory/11%3A_Basics_of_Graph_Theory/11.03%3A_Deletion_Complete_Graphs_and_the_Handshaking_Lemma` (visited on 04/21/2024).

[5] Joy Morris. *Euler Tours and Trails.* URL: `https://math.libretexts.org/Bookshelves/Combinatorics_and_Discrete_Mathematics/Combinatorics_(Morris)/03%3A_Graph_Theory/13%3A_Euler_and_Hamilton/13.01%3A_Euler_Tours_and_Trails` (visited on 04/21/2024).

[6] OpenStax. *Hamilton Cycles.* URL: `https://math.libretexts.org/Bookshelves/Applied_Mathematics/Contemporary_Mathematics_(OpenStax)/12%3A_Graph_Theory/12.08%3A_Hamilton_Cycles` (visited on 04/21/2024).

[7] *Some Basic Theorems on Trees.* en-US. Section: Advanced Data Structure. Nov. 2018. URL: `https://www.geeksforgeeks.org/some-theorems-on-trees/` (visited on 04/27/2024).

[8] Wolfram. *Gray Code.* 2024. URL: `https://mathworld.wolfram.com/GrayCode.html`.

All figures were generated using the NetworkX and MatPlotLib packages for Python, the R statistical analysis language, or the Tikz LaTeX package.