Dependable Network Topologies

Joshi, Prashant

DOI
10.4233/uuid:c3958573-4de3-4e41-b512-e7a383a14a5e

Publication date
2019

Document Version
Final published version

Citation (APA)
Joshi, P. (2019). *Dependable Network Topologies*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:c3958573-4de3-4e41-b512-e7a383a14a5e

Dependable Network Topologies

Dependable Network Topologies

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof. dr. ir. T. H. J. J. van der Hagen,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op
dinsdag 7 October 2019 om 15:00 uur

door

Prashant Dattatraya JOSHI
Master of Science in Computer Science, University of South Carolina, Columbia,
South Carolina, USA
geboren te Pune, India.

Dit proefschrift is goedgekeurd door de

promotor: Prof. dr. ir. S. Hamdioui
promotor: Prof. dr. ir. K. L. M. Bertels

Samenstelling promotiecommissie:

Rector Magnificus,	voorzitter
Prof. dr. ir. S. Hamdioui	Technische Universiteit Delft
Prof. dr. ir. K. L. M. Bertels	Technische Universiteit Delft

Onafhankelijke leden:

Prof. dr. ir. P. Van Mieghem	Technische Universiteit Delft
Prof. dr. A. Sen	Arizona State University, Tempe, AZ
Prof. dr. D. Frank Hsu	Fordham University, New York, NY
dr. ir. F. A. Kuipers	Technische Universiteit Delft
dr. ir. S. Wong	Technische Universiteit Delft

ISBN/EAN 978-94-028-1709-6

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

“Gnyanam Paramam Dhyeyam”
- Knowledge is the Supreme Goal.

Motto IIT-B

Contents

Summary

Networks such as road networks, utility networks, computer and communication networks and even social networks are the backbone of human civilization. Network analysis enables quantitative measurement of the important criteria such as delays, ease of routing and fault tolerance, and is required to build efficient and robust networks. Computer networks have evolved over the last five decades in parallel with technology which has grown exponentially tracking 'Moore's Law', which projected exponential performance growth in computing. Notably though, the supercomputers of today pushing exascale performance are doing so, not primarily because of the improved performance of the microprocessors, but overwhelmingly due to the ability to network tens of millions of these microprocessors in systems. These systems depend very heavily on robust network topologies to achieve the exponentially growing performance seen over the last few decades. The network topologies in the world's top performing supercomputers have evolved with the focus towards boosting performance by binding together an increasing number of processors in efficient networks over the years. Popular topologies have included torus, hypercubes, fat trees and some combinations thereof. The biggest drawbacks of the rapidly increasing number of devices networked together are the increased message delays, the declining ability to withstand various faults, and security issues. Building such supercomputers of today has very high down costs, and it is imperative that their utilization is maximized. This requires these high performance systems to be highly dependable also. This forms the motivation for the work in this thesis.

This research delves into building the most efficient topologies to enable high performance by reducing message latencies, and at the same time showcasing their highly robust nature. This work coins a new structure called the 'torculant', which is based on the merger of the torus and the circulant. The work proposes a framework for a topology based on recursive line graphs of the torculant. It shows that if the proposed topology based networks are used instead of those in the supercomputers of the last ten years, the reduction in the message latencies, the number of I/O ports required, and the added robustness could have made their performance significantly better.

For example if the proposed network topology had been used by the IBM BlueGene/Cray machine which was the fastest machine a decade back, the peak delay would have seen a reduction by **86%**. The Fujitsu K supercomputer would have seen a **50%** improvement in the number of region failures it could tolerate. The fastest supercomputer of last year, Sunway TaihuLight, would have seen a peak delay reduction of possibly **50%**. Looking at it in a slightly different way, if the existing peak delay is acceptable, then the number of supernodes that could be connected in the Sunway TaihuLight would be **400X** the number in the existing

configuration. None of the topologies in use in the supercomputers of the last decade have optimal region based fault tolerance, while the proposed topology is not only optimal, it is region based optimal and in fact shows a peak delay degradation of only **one** with the maximum number of region failures. A unique feature of the proposed topology is that the routing table size is fixed irrespective of the network size, thus enabling many desirable features like security, fast routing including in the presence of faults. For instance a network of degree five will require tables of size of the order of 25, whether the number of nodes is in the hundreds or the millions. In comparison other methods will require routing tables based on the network size. A new metric called 'region based container' is proposed as a powerful tool to measure the degradation of networks with region based failures. The contribution of this work has three separate types of results in the areas of reliability, robustness and security.

Reliability is the assurance that the system will work per the design specifications. This research work enables network designs that achieve the best known message delays despite not requiring an increased number of I/O ports as the current designs do. The work extends the reliability constraints to allow optimal connectivity despite faults in the network.

Robustness is the ability for the system to function, albeit with a degraded performance when the conditions are out of the design specifications. This work showcases that the proposed topology has outstanding properties in being able to function with a large number of faults without degrading appreciably. In terms of region failures, the proposed family of networks go well beyond the robustness afforded by the topologies in use in today's supercomputers. The work also goes on to show how self-healing is possible when problems are identified, and bounds the efforts to achieve it.

Security is as important to designs as power and performance in today's age. Its definition is very broad in that it deals with fault detection, malicious or otherwise, misdirected messages for stealing or denial of service attacks, message deletion, etc. This work proposes a security protocol based on the properties of the 'seed' graph which can be orders of magnitude smaller than the final network. This makes it easy and cost effective to track misdirected messages with and without faults in the system.

In summary this work describes a robust network topology that when compared with the existing topologies of supercomputers of the last decade shows much better results on many of the important metrics for efficient computing by increasing the performance and robustness.

Samenvatting

Samenvatting in het Nederlands

Netwerken zoals wegennetwerken, nutsnetwerken, computer- en communicatienetwerken en zelfs sociale netwerken vormen de ruggengraat van de menselijke beschaving. Netwerkanalyse maakt kwantitatieve metingen mogelijk van belangrijke criteria zoals vertragingen, gemakkelijkheid van het routeren en fouttolerantie, en is vereist voor het bouwen van efficiënte en robuuste netwerken.

Computernetwerken zijn in de afgelopen vijf decennia samen geëvolueerd met de technologie die de exponentiële groei voorspeld door 'Moore's Law' heeft gevolgd, die exponentiële prestatiegroei in computers voorspelde. De supercomputers van vandaag die nabij de exascale prestaties komen, doen dit echter niet in de eerste plaats vanwege de verbeterde prestaties van de microprocessors, maar vooral vanwege het vermogen om tientallen miljoenen van deze microprocessors in systemen te netwerken. Deze systemen zijn erg afhankelijk van de robuuste netwerktopologieën om de exponentieel groeiende prestaties van de afgelopen drie decennia te bereiken

De netwerktopologieën in 's werelds best presterende supercomputers zijn geëvolueerd met de focus op het verbeteren van de prestaties door het koppelen van een groeiend aantal processoren in efficiënte netwerken. Populaire topologieën omvatten torussen, hypercubes, fat trees en enkele combinaties daarvan. De grootste nadelen van het snel toenemende aantal apparaten in het netwerk zijn de verhoogde berichtvertragingen, verminderde vermogen om verschillende fouten te weerstaan en beveiligingsproblemen. Het bouwen van dergelijke hedendaagse supercomputers heeft zeer hoge kosten, en het is absoluut noodzakelijk dat hun gebruik wordt gemaximaliseerd. Dit vereist dat ze zeer betrouwbaar zijn.

Dit onderzoek bestudeert de meest efficiënte topologieën om hoge prestaties mogelijk te maken door de berichtlatenties te verminderen en tegelijkertijd hun zeer robuuste aard te demonstreren. Dit werk stelt een nieuwe structuur voor gebaseerd op de samensmelting van de torus en de circulant, en definieert nieuwe 'torculant' structuur.Het toont aan dat als de voorgestelde topologie-gebaseerde netwerken werden gebruikt in plaats van die in de supercomputers van de afgelopen tien jaar, de vermindering van de berichtlatenties, het aantal benodigde I/O-poorten en de toegevoegde robuustheid hun prestaties aanzienlijk hadden kunnen verbeteren beter. Een nieuwe statistiek genaamd 'region based container' wordt voorgesteld als een krachtig hulpmiddel om de degradatie van netwerken met regionale fouten te meten. Dit werk heeft drie verschillende soorten resultaten op het gebied van betrouwbaarheid, robuustheid en veiligheid.

Reliability is de garantie dat het systeem volgens de ontwerpspecificaties werkt. Dit onderzoekswerk maakt netwerkontwerpen mogelijk die de beste bekendste berichtvertragingen bereiken, ondanks dat er geen groter aantal I / O-poorten nodig

is zoals in de huidige ontwerpen. Het werk breidt de betrouwbaarheidseisen uit om optimale connectiviteit mogelijk te maken ondanks storingen in het netwerk.

Robustness is de mogelijkheid van het systeem om te werken, zij het met een verminderde prestatie wanneer de omstandigheden buiten de ontwerpspecificaties vallen. Dit werk demonstreert dat de voorgestelde topologie uitstekende eigenschappen heeft om te kunnen functioneren met een groot aantal fouten zonder aanzienlijk te verslechteren. In termen van regio-fouten gaat de voorgestelde familie van netwerken veel verder dan de robuustheid van de topologieën die in de hedendaagse supercomputers worden gebruikt toestaat. Het werk gaat ook verder door te laten zien hoe zelfgenezing mogelijk is wanneer problemen worden geïdentificeerd, enbegrenst de inspanningen om dit te bereiken.

Security is vandaag de dag net zo belangrijk voor ontwerpen als kracht en prestaties. De definitie van veiligheid/security erg breed in die zin dat het raakt aan foutdetectie, kwaadwillend of niet, verkeerd geadresseerde berichten voor diefstal of denial of service-aanvallen, verwijderen van berichten, enz. Dit werk stelt een beveiligingsprotocol voor gebaseerd op de eigenschappen van de 'seed' graaf dat ordes van grootte kleiner dan het uiteindelijke netwerk kan zijn. Dit maakt het gemakkelijk en kosteneffectief om verkeerd geadresseerde berichten met en zonder fouten in het systeem te volgen.

Samengevat beschrijft dit werk een robuuste netwerktopologie die in vergelijking met de bestaande topologieën van supercomputers van het afgelopen decennium veel betere resultaten oplevert op veel van de belangrijke meeteenheden voor efficiënt computergebruik door het verbeteren van de prestaties en robuustheid.

Acknowledgements

I wish to take this opportunity to offer my heartfelt thanks to my promoter Professor dr. ir. S. Hamdioui for providing me the opportunity to pursue my Ph.D. thesis under his guidance. His understanding and encouragement was vital to enable me to work through these years while working full time in parallel in the industry. I shall forever be indebted to him for his time and efforts. Professor dr. ir. K.L.M. Bertels' enthusiasm and encouragement as my promoter was very valuable in keeping this effort going. My sincere thanks also go to the CE and Graduate School staff for all the management support especially since I was remote most of the time. Trisha, Lidwina, Joyce, Erik, and Petra have helped in various way over the years to make each step flow smoothly.

My sincere appreciation to all my committee members for their thoughful comments, time and effort to make this a better thesis. Professor Hsu's suggestions on how to extend the work, and his tireless help in proof reading papers and guidance is something I will always cherish. My heartfelt thanks go to Professor Sen, for his time and discussions which lead to some of the ideas in this research work. I would like to thank Professor dr. ir. P. Van Mieghem, dr. ir. F. A. Kuipers and dr. ir. S. Wong for their time, efforts and feedback. On a lighter note, I am honored to have the Erdős number two with my collaboration with Professor Hsu.

I take this opportunity to thank many of my friends at TU Delft who have helped me along the way, notably Jorik Oostenbrink, Dr. M. Taouil and Dr. I. Agbo, who answered many of my questions, helped in the translation, and were instrumental in helping me in the dessertation writing phase of this journey.

My association with Professor Abhijit Sengupta, from about thirty five years back, started this journey and kindled my love for graph theory and applications to fault tolerant networks. To him, I will always be indebted for his selfless help. To Professor Israel Koren, and late Professor Akers, I extend my sincere regards for being there at various times during the last three decades. The journey would not even have started without the association with numerous people both in the faculty and peers from the Indian Institute of Technology, Bombay from almost four decades back. It was there that the philosophy of '*Gnyanam Paramam Dhyeyam*' (Knowledge is the Supreme Goal) was ingrained into my psyche and till today acts as my guiding light in life.

I would like to also thank Intel and my upper management team for their support in this endeavour.

To my extended family and friends, there are no words to express the support and love I have received. The guidance from my parents all my life has had an immeasurable impact on me. My sisters have been there for me through the ups and downs. My regards to all my in-laws for being there all these decades and for their

love and support. My love to my children Atharva and Rucha, for their encouragement, and to my wife Madhavi, for her unwavering support, without whom I might not even have restarted my efforts to get a PhD after a break of three decades.

1

Introduction

Networks are a broad concept that can be applied to various aspects of life such as communications, power distribution, transportation etc. The performance and dependability of those networks depend on how networks have been designed. This work looks at the design of dependable high performance computer network topologies, where reliability, robustness and security are equally important.

1

1.1. Basic Overview of Networks and Topologies

As the demand for High Performance Computing has grown exponentially over the decades, the performance of the individual microprocessor itself has not kept pace with the needs. Instead the exponential performance growth has almost entirely been fueled by networking together an increasing number of processing elements using highly efficient and robust networks. Several types of network architectures and topologies have evolved over the years. Such networks are typically housed over a short distance, in a building connecting tens of millions of processing and storage elements, where the information is processed, and packets of data are sent and received over the network by switching elements.

The aim of this research is to design a family of network topologies that result in better performance across a range of metrics important to High Performance Computing. In this subsection we will cover the following details:

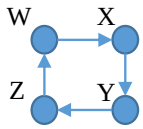
- Basic Definitions and Network Classification
- Popular Topologies
- Network Architectures of Recent Supercomputers

1.1.1. Basic Definitions and Network Classification

Traditionally, the use of the phrase *computer network* implied connections between processors; however, over the last few decades this term has evolved to include other devices as well especially in Storage Area Networks (SAN) and High Performance Computing (HPC). All interconnection networks very broadly speaking consist of two main elements, the network *nodes* and the network *links*. A node can be one that just helps route the data to the right receiver like a router, or could produce or consume the data as in a processor. Nodes in a direct network produce, consume, and route data, while in indirect networks the nodes can either route data or produce/consume it. Links or *edges*, on the other hand, are the communication medium used to connect the nodes. The links could be wired or wireless, directed or bidirectional. Since an undirected (or bidirectional) edge can be represented by two directed edges, the analysis using directed edges forms a superset of study using undirected edges. Hence, this work deals with networks with directed edges. In other applications where the networks are on the chip, directed ports are more often the norm.

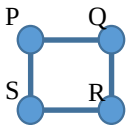
The work extensively uses the term 'region' to mean a set of nodes of the network, and the edges to and from those nodes. In literature the term region has been used primarily to depict a 'geometric' or a 'topological' region. In a geometric region the nodes and edges are those that lie inside a physical geographic region, such as a circle of a given radius centered at the middle of the region. Such nodes may not have any direct connection between them. On the other hand a topological region centered at a node, refers to those edges and nodes that can be reached from that node in a specified number of hops.

It is important to note that all references to the word 'region' in this work deals with topological regions and not geometric regions. In addition, all delays along the edges are considered equal, and hence the overall delay is a measure of the



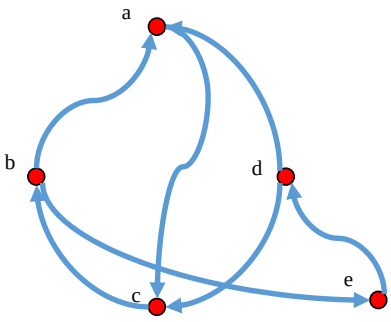
a. Directed graph

path: W → X → Y,
length of path = 2,
diameter of graph = 3,
node connectivity = 0

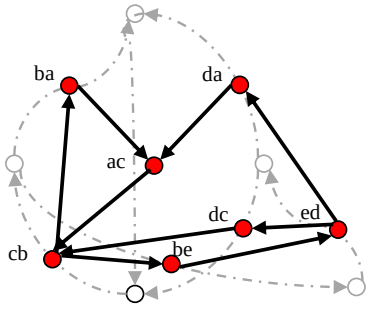


b. Undirected graph

path: P → Q → R,
length of path = 2,
diameter of graph = 2,
node connectivity = 1

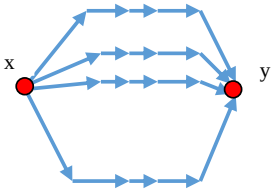


Graph G

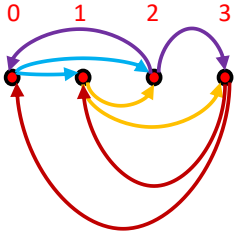


Line graph L(G)

c. Line graph of a directed graph, and node naming



d:4-wide container between x and y



e: circulant of degree two on directed graph of four nodes

Figure 1.1: Examples of some terms used in the study.

number of nodes hopped along the way. This means that there is no unequal weightage/costs to the edges.

The usage of other terms such as 'container of a graph' [61, 62] or graph 'circulants' [69] refer to typical usage in literature on such topics and are described briefly below.

Networks are typically modelled and analyzed using graph theory. So we will briefly introduce some terminology that will be used throughout the thesis. Figure 1.1 is used to illustrate some key terms.

- `Graph:` A graph G= (V,E) is a graph of *n* nodes, where |V| = *n*, and the edges (*p,q*) are elements of E when p and q are both elements of V. The graph is directed if the edges have a direction from *p* to *q*.
- `Degree:` The degree of a node is the number of edges incident on that node. The *indegree* (correspondingly *outdegree*) of the node is the number of edges incident *into* (correspondingly *out of*) that node.
- `Regular graph:` An undirected graph is regular if each node has the same degree. A directed graph is regular if every node's indegree and outdegree is the same.
- `Network topology:` The network topology is the arrangement of the elements of the network.
- `Path:` A path from a node *p* to a node *q* is a sequence of adjacent nodes and edges from the source node *p* to the destination node *q*.
- `Length of a path:` The length of a path from *p* to *q* is the number of edges along the path.
- `Distance:` The distance between two nodes *p* to *q* is the length of the shortest path between the two nodes.
- `Diameter:` The diameter of a graph is the largest distance between every pair of nodes in the graph.
- `Connected graph:` A connected graph is where every node is reachable by a path from every other node in the graph.
- `Fault:` A node fault occurs when a particular node cannot be used for message transfer. Correspondingly an edge fault is when that edge cannot be used.
- `Line graph:` A transformation of a graph G, such that the edges become the nodes of the line graph L(G), and there exists an edge between two line graph nodes, only if the corresponding edges in the graph G are adjacent. Figure 1.1c shows an example of a line graph L(G) of a directed graph G.
- `Ring:` A ring of a directed graph G, with nodes 0 to *n*-1, connects each node *i* to (*i* + 1) mod *n*.
- `Circulant:` A circulant of a directed graph G, with nodes 0 to *n*-1, connects each node *i* to (*i* + *j*) mod *n*, where *j* is equal to zero to *d*-1. Such a graph is also referred to as a D_d digraph. A ring is a circulant of degree one. More complex circulants can be defined with a fixed function to connect each node to the next *d* nodes per some formula that is applied to all nodes. Figure 1.1e shows an example of a circulant with degree two.
- `Region:` A region is a subset of nodes and attached edges.

- `Geographic region:` The set of nodes and edges that lie within a *geometrical* physical distance *r* from a node constitute the geographic region centered on that node of size *r*.
- `Topological region:` The set of nodes and edges that lie within a distance at least *r* based on the graph topology, from a node constitute the topological region of radius *r* centered on that node.
- `Node disjoint paths:` Two paths are node disjoint (correspondingly edge disjoint) if the two paths do not share any node (correspondingly edge). **Note** that two node disjoint paths are necessarily edge disjoint, but not the other way around.
- `Region disjoint paths:` Two paths are region disjoint if no node from one path shares the regions through which any of the nodes of the other path go through.
- `Node connectivity:` The minimum number of any arbitrary nodes that need to be removed to disconnect a graph is node connectivity of the graph. **Note** for a graph whose smallest node degree is *d*, the node connectivity cannot be more than *d*.
- `Edge connectivity:` The minimum number of any arbitrary edges that need to be removed to disconnect a graph is is edge connectivity of the graph.
- `Region based connectivity:` The minimum number of arbitrary regions of a given size that need to be removed to disconnect a graph is the region based connectivity or RBC of the graph. In other words, if the region based connectivity of a graph is *d* then there are at most *d* number of region disjoint paths between every pair of nodes in the graph.
- `Star container:` For some nodes *x*, y_1, y_2, … y_w of a graph G without self-loops or multiple edges where *w* is a positive integer and *x* is not equal to y_i, for any *i*, a collection of internally node disjoint paths from *x* to y_1, y_2, ..y_w one for each y_i , is defined as a star container from *x* to y_1, y_2, .. y_w. In case any node y_r is repeated *t* times then the container needs to have *t* internally node disjoint paths from *x* to y_r also.
- `Wide container:` In the special case where $t = w$ and hence $y_1 = y_2 = \ldots y_w$, equal to say *y*, the *w*-star container is called a *w*-wide container from *x* to *y*. Figure 1.1d shows an example of a 4-wide container.
- `Wide container length:` The length of a *w*-wide container is the maximum length *l* of all paths in that container.
- `Container distance:` The *w*-distance container distance from *x* to *y* is the minimum length of all possible container lengths between *x* and *y*.
- `Container diameter:` The *w*-wide diameter of a network is the maximum distance of *w*-wide containers across all pairs of nodes. **Note** that the *w*-wide diameter is different from the diameter in that it focuses on the worst delays in the network in the presence of *w*-1 faults. And in the special case of $w = 1$, the *w*-wide diameter boils down to the same as the network diameter.
- `Region based container:` A *region based container* is a new concept in network analysis that is being defined in this study. Similar to the node disjoint paths that determine the network container, region disjoint paths are

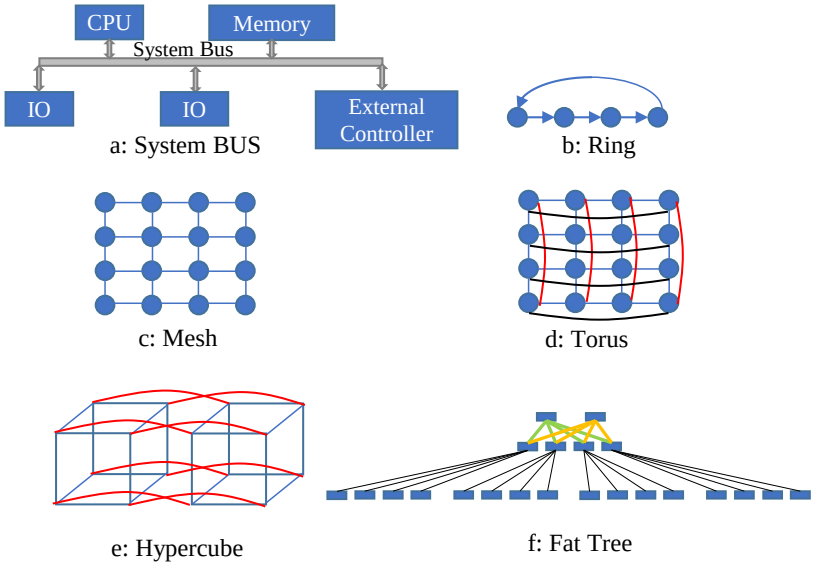


Figure 1.2: Common Topologies of recent networks.

used to define a region based container. A region based container is a set of region disjoint paths from a node x to y.

- `Region based container length, distance and diameter:` Similar to the normal container length, distance and diameter of a graph, the corresponding region based container length, distance and diameter are defined with region disjoint paths instead of node disjoint paths.

1.1.2. Popular Topologies

Topologies have evolved over the decades along with technology [4, 5, 6, 7, 8, 9, 10, 17, 20, 27, 29]. Many of the popular topologies are shown in Figure 1.2, with a qualitative comparison of their properties in Table 1.2 in a later section. The interesting topologies are as follows:

- `System Bus`
- `Ring`
- `Mesh`
- `Torus`
- `Hypercube`
- `Fat Trees`

These topologies are explained below.

1. **System Bus:** The simplest way to connect multiple devices like cores, memories, controllers, ports, etc. through a system bus is shown in Figure 1.2a. Although it has the advantage of a simple topology it has issues when multiple devices need to use the bus at the same time causing collisions. This design is feasible when the number of devices that need to connect are small and typically in close proximity, for example inside a chip [29, 41, 44].

2. **Ring:** The avoidance of such collisions gave way to the ring topology, where a higher number of devices could now be connected. Figure 1.2b shows an example of a directed ring. The messages are placed on the ring by each sender if a slot is available and the intended receiver would detect and consume the message ignoring those not intended for itself. This extended the simplicity, however the delays were still linearly proportional to the number of nodes. Such topologies were popular internal to the chip when multi core chips started to be designed a few decades back. If the topology happens to use a unidirectional ring, then this topology is not tolerant to even a single fault [29].

3. **Mesh:** With the success of Moore's Law, the number of cores within a single chip started growing and at a certain point such ring structures gave way to the mesh. Figure 1.2c shows an example of an undirected mesh (in the actual chips this might be in the form of two physical unidirectional links in the opposite direction, or some more complex control using one link and tristated logic) [10, 67]. Such a topology is very common in chip designs where an array of processors is required with applications typically dealing with arithmetic and floating point computations, such as graphical processors. This allowed the use of a large number of processors and the reduction of the delays from linear to O($D*\sqrt[D]{n}$) where the meshes were D dimensional with n number of nodes. This also made the topology optimally fault tolerant.

4. **Torus:** A logical extension of the mesh topology is the torus where each row along each dimension is actually a ring as shown in Figure 1.2d. This halves the worst delay in terms of the number of hops for undirected tori. In actual implementation these undirected links are often two unidirectional links in opposite directions. The use of a torus as a topology has multiple advantages in that the degree of each node does not need to increase very rapidly with an increasing number of nodes to connect. In bidirectional networks, an advantage over the ring is obviously the ability to withstand 2D-1 node failures for a D dimensional torus. The delay of the messages is at most $D*\frac{\sqrt[D]{n}}{2}$ for a network of n nodes and a D dimensional torus. This is an improvement over the ring and the mesh. However, for large number of nodes this is still a very high delay. Higher dimension torus topologies as high as 6-dimensions are popular among the supercomputers of the world [10, 27, 67].

5. **Hypercube:** The hypercube shown in Figure 1.2e is an esoteric topology that has some of the best properties of delays and fault tolerance. The delays were brought down to $\log_D n$, where D is the dimension and n is the number of nodes. However, a big drawback is that the degree of each node rises as the number of nodes required increase resulting in impractical designs for very large designs. Many researchers have come up with modifications of this topology to get around the issue, but this type of topology still suffers from the need of very high degree, and a very constrained number of nodes that are possible to be implemented [4, 9, 39, 41].

6. **Fat Tree:** The fat tree topology, shown in Figure 1.2f, often does not have a uniform degree in its underlying graph. This results in some nodes having extremely high degrees which in some cases could be in the hundreds. Although this results in the delays coming down to $O(\log_d n)$, where d is the degree and n is the number of nodes, the network is often not optimally fault tolerant. More importantly, the number of ports on a node (degree) can become extremely large resulting in a much more complex design and control. This topology, however, is in use in the most recent and fastest supercomputers in the world due to the very small delays in message routing [12, 13].

The design of most network topologies consists of one of the above topologies at times with minor modifications. This study on the other hand uses a new topology as a 'seed graph', and then transforms it recursively multiple times to come up with an entirely new topology. The seed graph is a torus with each of its rows and columns replaced by circulants. This proposed topology is being named a '**torculant**'. This torculant then recursively goes through a newly defined transformation called '**Extended Line Graph**'. The Extended Line Graph enables the addition of a few nodes in a judicious way after the line graph transformation has been performed. This concept of Extended Line Graph gives a lot of freedom to ensure that the total number of nodes that can exist in the network is not constrained by some formula. This forms the crux of the step by step process to design the network.

1.1.3. Network Architectures of Recent Supercomputers

A review of the topology and performance of a few of the fastest supercomputers in recent years is discussed below. The number of cores in these systems has grown from a few hundred thousand a decade back to tens of millions in the most recent machines.

- IBM BlueGene/L ®: The fastest supercomputer around 2008

As the HPC systems started to lump even more processors, the multicore chips were packaged onto cards and multiple cards on a node card. Multiple node cards made up a rack and finally, multiple racks made up the full system. Figure 1.3 shows the structure of the IBM's Blue Gene® (image taken from [65]). The nodes are configured as a 32X32X64 3D torus, with each node connected to its six immediate neighbors along each dimension. The number of cores were 212K. The delay

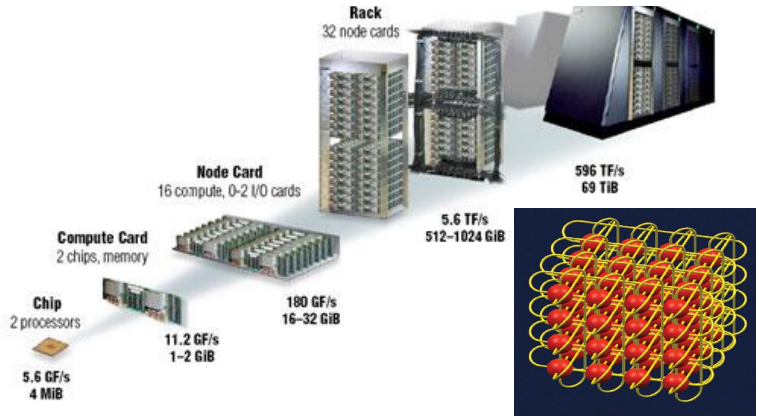


Figure 1.3: IBM BlueGene/ L ® system configuration and the 3D Torus topology.

between two nodes through the 3D torus would involve at most 16+16+32 = 64 hops. This delay would limit the performance and hence future generations of the BlueGene were built with a higher dimension torus to reduce this delay.

- Fujitsu K ®: The fastest supercomputer around 2012

The Fujitsu K supercomputer network architecture was designed with a TOFU (TOrus FUsion) interconnect. The original Fujitsu K supercomputer interconnect architecture is shown in Figure 1.4 (image taken from [67]). Each multi-core processor chip was connected through eight bidirectional ports to an interconnect controller which had an additional ten bidirectional ports. Of these ten ports, six were for the XYZ 3D torus. This XYZ torus was the 'global' scalable torus. Each node of this XYZ torus contained an abc torus with twelve nodes arranged as a 2X3X2 3D torus. The Cartesian product of the XYZ and abc produced the hybrid 6D architecture. About two thirds of the links were optical in nature for high bandwidth. Each node was a multi-cpu chip connected as a 6D topology of (X, Y, Z, a, b, c) = (24, 18, 17, 2, 3, 2) for a total of 705024 cores. To go from any node to any other node, a maximum of 12+9+8 = 29 hops along the XYZ torus followed by an additional three along the abc torus for a total of 32 hops was sufficient. In comparison with the IBM BlueGene/L, the number of nodes went up by a factor of three, and the peak delay dropped by a factor of two. This gave a peak performance of 20X though the core frequency only went up by 3X.

- Sunway TaihuLight: The fastest supercomputer in 2016/2017

The network architecture of the fastest supercomputer in the world today, the Sunway TaihuLight [12,13], is shown in Figure 1.5. It consists of 260 core nodes, and 256 such core nodes combine to form one supernode. Four supernodes form a cabinet and 40 cabinets make up the full system for a total of 260X256X4X40 =

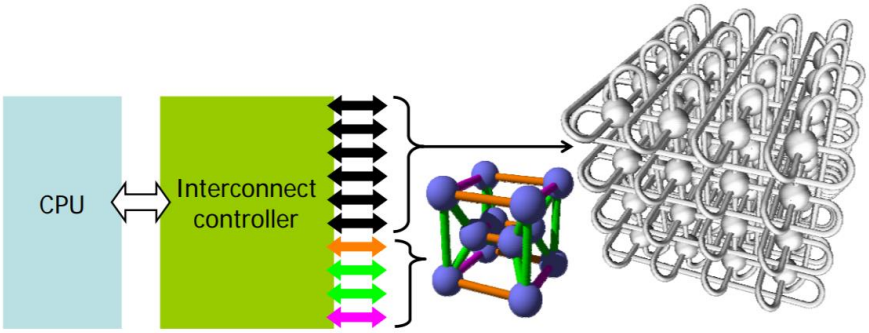


Figure 1.4: TOFU (TOrus FUsion) topology of the Fujitsu K ®.

10,649,600 cores. The upper levels are connected by mesh and NOC structures. Since not all details of the topology are public, an assumption can be made that the 256 nodes inside a supernode and the 40 cabinets are connected as torus, while the four supernodes are completely connected. If so, it is possible that the number of stages for one part of the system to communicate with another part of the system might have to go through a total of $\sqrt{256} + 1 + \sqrt{40}$ which is 24. However, the number of ports on each super node must be more than 256 which can get expensive in the design.

As can be seen from the examples listed, the exponential gains in the performances over the decades are ***not*** primarily from the performance of the microprocessors, but largely from the ***topology*** that enables a larger number of processors to be networked.

1.2. Importance of Network Dependability

Communication, a cornerstone of society, has developed new dimensions over the last half-century. The advent of computer networks has changed the landscape of our daily life. As a result, ensuring robust networks enables a seamless continuity in everyday life. Applications in the areas of banking, defense, weather prediction and space to name a few, have a great demand for high performance, low power, and robust computers. For critical applications, systems with faults must function reliably and correctly within the design specifications, possibly with reduced performance. Robust designs reduce the mean time to catastrophic failures, especially in sectors where there is no second chance to redo calculations or replace faulty parts. Ensuring reliable and robust communication between many processing units has become part of the design process at the chip, server, and much larger infrastructural levels.

The push from a few decades back for the fastest individual microprocessor

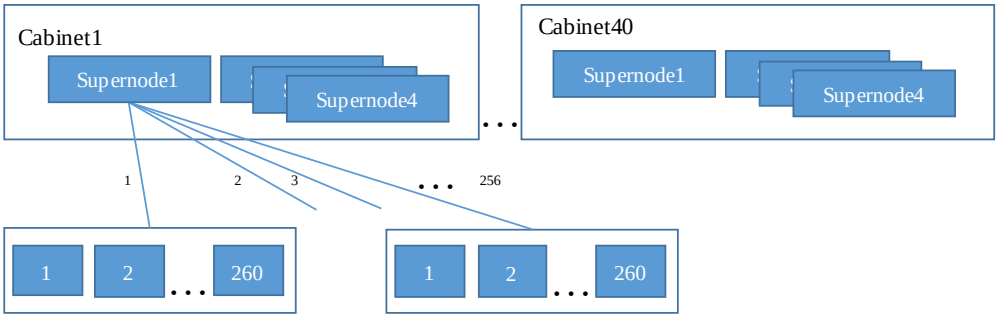


Figure 1.5: Network Architecture of the Sunway TaihuLight.

gave way to distributed and highly parallel computing machines as a better system for High Performance Computing (HPC) [1]. The remarkable success of Moore's Law has ensured that many of the tasks previously carried out external to the chip are now designed on-chip. This has resulted in hundreds of microprocessors, graphic cores, shared memories, memory and peripheral controllers being designed on-chip. This has enabled millions of chips working together to obtain exascale performance [2]. It is important to note that this performance gain has not been only due to higher performing processors, but largely due to the reliable and robust networks that connect these processors. The realization that this is the only way to solve today's scalability, power, and performance needs, leads to highly connected, robust, and secure networking.

For systems designed for outer space applications where neither replacement nor repair is possible, robust designs that can withstand failures are not just necessary but a requirement. Similarly, in applications of health or human conflicts, cyber and national security, there are no second chances to redo the results reached due to networking errors, and reliability is not just an added benefit but could be a matter of life and death. In many cases the problems tend to occur in physical proximity to each other. Dependable networks need to be cognizant of such factors in the analysis of the designs. However, in a viable and dependable product, simply ensuring fault tolerance is insufficient to ensure a commercially successful product. The metrics that gauge the integrity of a design must include the delay of the message passing, power requirements, memory requirements, ability to spot erroneous or malicious behavior, and complexity of the control for normal and abnormal situations.

A 'dependable' network is one which displays three important characteristics. It should be reliable, robust and have some good security features. The terms 'reliable' and 'robust' in the context of this work are defined as below.

- `Reliability:` The network is deemed *reliable* when it is confirmed to function correctly to the specifications for which it was designed during its lifetime. Just having reliable networks is not of much use if the specifications cannot be tuned to obtain the highest performance. With versatile network topologies, the specifications can be made increasingly constrained to obtain better performance.

  This work showcases the ability to design networks with specifications that enable high performance while at the same time enabling constraints for delays and the number of faults it can withstand (i.e. reliability).

- `Robustness:` A *robust* network is one which will work even outside of the specifications but possibly at a degraded level of performance. An important measure of the robustness of a network is how gracefully its performance degrades despite different types of faults that take the system outside of its design specifications.

  The proposed robust networks have some of the most graceful degradation in performance, can withstand many more faults than existing topologies, enable self-healing and bound the efforts required for changes and repair.

- `Security:` Security in computer networks has taken on a life of its own in recent years. Malicious attacks from denial of service, corruption, misdirection of data or stealing are aspects that need proper understanding and identification.

  This work analyzes the ability to identify misdirected messages, using a new protocol based on the topology.

1.2.1. Dependability Metrics

Systems cannot be compared unless the metrics of dependability are quantifiable. Many metrics have been proposed and are widely used to evaluate network performance [3, 6, 7, 8, 14, 16, 17, 18, 30]. Some common metrics and some being proposed in this study that gauge the integrity of the networks are shown in Table 1.1.

These metrics are explained in details below.

1. **Network Latency:** The metric of worst case network latency is easily modelled by the diameter of the network when there are no faults in the system. This allows the network designers and users to reliably plan for delays which will always be bounded above by a certain quantity in the worst case. This is one of the oldest and most commonly studied metric for network performance
2. **Fault tolerance:** The fault tolerance of a system is related to the node connectivity of the network. If a node has d neighbors, then quite obviously the network's fault tolerance will have to be less than d. For a regular network of degree d, the maximum fault tolerance hence will be bounded by d-1. If the network achieves this limit the network is called optimally fault tolerant. Comparing two network designs by this metric enables the designer to plan for robustness of the network.

Metric	Characteristic	Measure
Network Latency	Reliability	Delay in message passing bounded by $O(\log_d n)$.
Fault Tolerance	Reliability & Robustness	Is the network optimally fault tolerant and withstand *d*-1 node faults?
Degree, Port constraints	Reliability	Qualitatively, the number of ports per node should be low. Design and control become difficult with high degree per node.
Region Connectivity	Reliability & Robustness	Can the design withstand *d*-1 region faults?
Delay degradation	Robustness	How gracefully do metrics like delay degrade with *d*-1 node faults?
Delay degradation to region faults	Robustness	How gracefully do metrics like delay degrade with *d*-1 **region** faults?
Routing table memory, control complexity	Reliability	Are the routing tables bounded by $O(\log_d n)$, to ensure that the memory usage and complexity is reduced?
Misrouting	Security	Can misrouted messages be identified in $O(\log_d n)$ steps?

Table 1.1: Measureable metrics.

3. **Degree, Port constraint:** If the topology of the network requires the nodes to have a very high degree, then the number of ports to be designed will be very high. This not only makes the design more restrictive, but also makes the control more complex.
4. **Region connectivity:** Commonly used metrics for network analysis used to look at node failures as individual point failures without consideration to the locality. In real life however a problem on one part might affect a completely unrelated functionality of an otherwise perfectly working part in the vicinity. The robustness of the network could be analyzed using the topological or geographic region based connectivity. Examples of this would be hot spots on a chip in which a problem caused by one error tends to affect other devices in the physical neighborhood on the chip. These would be geographic region based faults. Similarly, a faulty node in a communication network puts extra burden on its immediately connected neighbors, but might not be in its physical neighborhood. This type of analysis is done by topological region based fault tolerance.
5. **Delay degradation:** Two robust networks may both function outside of the specifications in a degraded manner, but they need to be compared by quantifying how gracefully the two networks degrade in the presence of faults. This aspect is tested by the containers of the network which look into the

delay degradation in message passing in the presence of faults. Therefore, container based network analysis is especially important for analysis of the robustness of the networks where the system is expected to have very high functionality and performance, such as in space applications or cyber security and defense.

6. **Delay degradation in the presence of *region* faults:** This is a new metric being proposed in this thesis. As was shown in recent studies [18, 30, 31, 32, 34] network robustness is dependent not just of node disjoint paths, but on region disjoint paths since many real life networks are affected by the locality of the problems. The ability to not only have graceful degradation in point faults, but also in region faults is increasingly becoming more important as technology advances.
7. **Routing tables, memory and control complexity:** Message routing requires a set of rules or tables to analyze the next node in a path of shortest distance to the destination. These would be used for either load balancing, loop avoidance or alternate path determination in the presence of faults. The size of the routing tables and the ease of the analysis is a direct measure of the complexity of routing of messages. Smaller routing tables which include next node information for paths with and without the need for alternate paths will not only reduce memory requirements in each node, but also affect the energy required for such analysis.
   The degree of the nodes is another important factor to consider as very large degrees are not very practical. Theoretically the hypercube has some of the best features in most metrics but for very large number of nodes, the degree of each node becomes impractically high. The same is true of the fat tree architecture. Hence low degree nodes are desirable while at the same time maintaining high fault tolerance and low diameters.
8. **Misrouting:** The security of a network is of paramount importance in today's world where cybersecurity is on every nation's mind. The ability to identify misdirection of messages or denial of service attacks helps to keep a network robust in such situations. This is a very important metric for network security and robustness.

These metrics have been studied extensively in this research for the proposed family of robust networks and it compares very favorably with existing networks.

1.2.2. State of the art of Dependable Networks

Clearly, the reliability of the whole system is a function of the inherent reliability of the individual components, which can be affected by issues such as on-chip variations, age effects, or simply faulty manufacture. To make chips more robust, chip designs often include features such as redundancy, and the ability to detect and correct errors. Such an approach is important for network topologies as well with networks taking such a major role in the performance of today's supercomputers. Topologies that enable such features are dependable (reliable, robust) and are an important part of today's state of the art for high end systems.

Along with such reliable and robust features, the supercomputers of today have

Topology	Network Latency	Fault Tolerance	Degree/IO Port constraint	Region Connectivity	Delay Degradation with (d-1) Region Faults
System Bus	Bad	Bad	Very Good	N/A	N/A
Ring	Bad	Optimal	Very Good	N/A	N/A
Mesh	Bad	Optimal	Very Good	Suboptimal	∞
Torus	Bad	Optimal	Very Good	Suboptimal	∞
Hypercube	Very Good	Optimal	Very Bad	Suboptimal	∞
Fat Tree	Very Good	Optimal	Very Bad	Suboptimal	∞

Table 1.2: Qualitative Comparison of Recent Network Topologies.

evolved along the lines of protection, detection and resolution of attacks on the systems (security). Access to sensitive data is often controlled and multiple levels of authentication are required before the secure data is made available. Often there are multiple layers of protection to take care of different levels of attacks. Intrusion and malicious rerouting detection is part of the increased security features that have developed over time.

Table 1.2 shows a qualitative comparison of the features of the different network topologies. While one can interpret regions for the bus or the ring, the concepts are not of much consequence and hence listed as not applicable. Also as can be seen, it is possible to disconnect the networks with region faults on the other topologies.

As seen in Figure 1.6 the performance of the fastest supercomputers in the world has seen an exponential growth [11]. The green dots represent the sum of the performance of the top 500 supercomputers in that year. The brown triangles and the blue squares show the performance of the fastest and the slowest supercomputer in the top 500 supercomputers of the year. This rapid pace is expected to continue in the foreseeable future as well. As it can be seen in Table 1.3, along with the performance of individual microprocessors the ability to network almost 100X more processors over the last ten years has maintained the performance trajectory [12].

1.3. Challenges and Opportunities

Designing dependable networks for high performance computing still faces some major challenges. In the rest of this section we will highlight them. As in most cases, challenges perceived in achieving a goal end up being opportunities at the same time. Challenges like message latencies with and without faults in the

1

System	Site	Topology	Year	Cores	Core Freq.	Peak Perf (PFlops)
IBM Blue-Gene/L	Lawrence Livermore National Lab	3D Torus	2008	212K	700MHz	0.594
IBM Road-runner	Los Alamos National Lab	Fat-tree crossbars	2009	129K	3.2GHz	1.456
Cray Jaguar	Oak Ridge National Lab	3D Torus	2010	224K	2.6GHz	2.331
NUDT Tianhe-1A	National Supercomputing Center, Tianjin	Fat-tree	2011	186K	2.9GHz	4.701
Fujitsu K Computer	RIKEN Advanced Institute for Computational Science	6D Mesh/Torus	2012	705K	2GHz	11.28
IBM Sequoia Blue-Gene/Q	Lawrence Livermore National Lab	5D Torus	2013	1.5M	1.6GHz	20.132
Cray Titan	Oak Ridge National Lab	3D Torus	2014	560K	2.2GHz	27.112
Tianhe-2	National Supercomptuer Center, Guangzhou	Fat-tree	2015	3M	2.2GHz	54.902
Sunway TaihuLight	National Supercomputer Center, Wuxi	Multiple at different levels	2016-2017	10M	1.45GHz	125.435

Table 1.3: Recent Supercomputers Topologies, cores and frequencies.

Figure 1.6: Exponential trend in performance over the last few decades.

network, physical restrictions of building fault tolerant networks and the need for graceful degradation to enable a more robust network are some important challenges that face network designs.

1.3.1. Challenges

Some of the important challenges when designing networks for such high performance machines are considered below:

- `Reliability: Latency`
- `Reliability: Degree/Number of ports`
- `Reliability: Fault tolerance`
- `Reliability: Memory, power and control flow complexity`
- `Robustness: Graceful degradation with faults`
- `Robustness: Region based connectivity`
- `Robustness: Delay degradation with region faults`
- `Security: Detect misdirected messages quickly`

1. **Latency:** Along with technology the ability to enable topologies that will bring down the delays in message passing has been evolving. Network topologies

have further evolved from the days of a ring where the delays were linear, to a mesh and torus where the delays were proportional to $D*\frac{D\sqrt{n}}{2}$ where D is the degree of the network and n the number of nodes. The theoretically best possible values are and can only be achieved by more esoteric topologies like the hypercube or fat tree networks. However, these are at the expense of sharply increasing the number of I/O ports per node.

The challenge is to keep the delays as $O(\log_d n)$ where d is the degree and n is the number of nodes of the network at affordable cost.

2. **Degree/Number of ports:** The number of ports on any node (degree of the graph) has a direct effect on the complexity of the design. The ring, mesh and torus topologies lend themselves to keeping this parameter under check. On the other hand, in the hypercube and fat tree networks, this aspect can become very large. In today's technology some of the fastest supercomputers have degrees of the order of hundreds but the complexity of the design takes a hit. Most topologies tend to keep the degrees low to avoid implementation issues.

   The challenge is to keep the number of ports low.

3. **Fault tolerance:** The need for fault tolerance (node connectivity) arises as the down time of supercomputers can be costly and the ability to work around problems is essential. Obviously the system's fault tolerance is bounded above by the smallest number of ports d on a node. If it does tolerate d-1 number of faults, then it is an optimally fault tolerant system. Most topologies do try to meet this constraint. One notable exception is certain implementations of the fat tree topology.

   The challenge is to keep the topology optimally fault tolerant at affordable cost.

4. **Memory, power and control flow complexity:** Large and unique routing tables for each node will result in large memory and power requirements and add to the complexity of detecting misdirected messages. Moreover, the ability to find the shortest routes and enable load balancing in the presence of known faults can make the routing table very complex. If routing tables were required to identify the next node to which to send an outgoing message, the routing table sizes could become $O(n^2)$ at each node.

   The challenge is to keep the routing table size small for power and memory reduction, yet enable rerouting for load balancing or fault avoidance.

5. **Graceful degradation with faults:** When faults do occur and are detected, the routing control mechanism reroutes the messages. If the re-routed messages have delays greater than the paths without network faults, then performance gets affected adversely. The amount of the extra delay is the delay degradation. The degradation will depend on the number of faults, and it

is interesting to see the differences between topologies. For a bidirectional ring, this delay for one node fault will result in the delays going up linearly. However, with other topologies like the mesh and torus, this increase can be smaller. For the fat tree topology, depending on the actual details, it may disconnect the network entirely. The hypercube topology is an example of a topology that does not see any deterioration in the worst delay.

The challenge is to find a topology that is as close to a hypercube's behavior in terms of its delay degradation, while keeping the number of ports per node under check.

6. **Region based connectivity:** Just like the ability to tolerate point failures, the network should be able to withstand region failures. Region failures are important for various reasons such as hot spots on a chip that affect the geographical locality and degrade the functionality, or the increased message passing load on the topological neighbors of a failed network node. Region failures could also be caused by external events that could affect multiple nodes instead of one, in the vicinity of the fault. Such failures are now being seen as one of the most important ones to study in networks, [16, 19, 30, 31]. With that in place, it is worrisome to note that neither the ring, mesh, torus, hypercube nor the fat trees are robust enough to withstand d-1 region failures where the region size includes just a failed node and all its immediate topological neighbors.

   The challenge is to find a topology that will be able to withstand region failures, where the size of the regions is also deterministic.

7. **Delay degradation with region faults:** Like delay degradation with faults, topologies have different behaviors in the presence of region faults. It is possible for the networks to get disconnected with the supercomputer topologies in use today in the presence of region faults.

   The challenge is to have a topology that ensures the network remains connected, but still bounds the incremental delays by small amounts with region faults.

8. **Detect misdirected messages:** Faulty nodes could direct the messages along incorrect paths, or malicious attacks could result in misdirected messages. It is important for the system to detect misdirected messages and the node causing such misdirection. If this node is seen to perform such misdirection regularly, the node could be isolated and some self repair implemented.

   The challenge is to detect misdirected messages in a very short time to enable corrective action.

1.3.2. Opportunities

Each of the challenges listed in the previous subsection is an example of an opportunity to attain better results in the various metrics used for supercomputer network topologies.

A recent study [17] mentioned that finding a fault tolerant topology with low degree, small diameter, and high bandwidth in a network is like a "Die eierlegende Wollmilchsau" or a "egg-laying and milk-giving wooly-pig". This statement highlights the extreme difficulty and hence an opportunity. This research specifically shows a new topology based on modifications of '***torculants***' has many desirable properties for supercomputer networks.

Table 1.2 shown earlier gives a qualitative comparison of some of the challenges faced by the popular topologies prevalent today, and the desired topology. The desired topology should be such that the number of nodes is not restricted by any formula as in the case of mesh, torus or hypercubes. The maximum degree of a node (number of ports from/to each node) is not very high to keep the complexity of the design low. The topology should be such that it is optimally fault tolerant to both node and region faults. The delay degradation should be bounded and a very small number with the presence of node or regions faults. The ability to determine the shortest routes with and without the presence of node and region faults should be very efficient in terms of time, memory and power. The network topology should lend itself to quick diagnosis of misdirected faults, and in the event of a faulty node, enable very quick self-healing.

1.4. Contributions

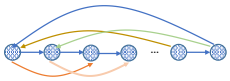
This section describes some of the goals of the work, the methodology used to provide the solutions, and the properties of the networks so devised.

1.4.1. Problem statement and methodology

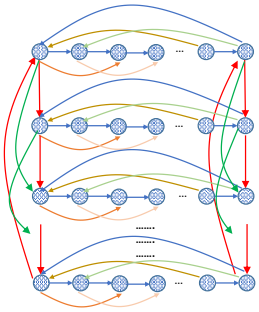
This research work focuses on devising reliable, robust and secure network topologies. As can be seen from the qualitative analysis and the state of the art, the existing network topologies have limitations when it comes to certain metrics.

The problem statement is to come up with a step by step mathematical procedure to determine the optimal topology of a directed regular network given the number of nodes desired and their degree, to

- minimize the peak delays as measured by the number of hops required along a path in the network during message passing
- maintain optimal fault tolerance as measured by the number of nodes that could be faulty and yet keep the remaining network connected
- maintain optimal region based fault tolerance as measured by the number of topological regions that can go faulty and yet keep the remaining network connected
- bound the maximum size of such regions that can go faulty and still keep the remaining network connected
- enable efficient fastest path determination from the source to the destination node in the network using small memory impact
- enable efficient alternate path determination from the source to the destination in the presence of node or topological region faults
- enable the efficient detection of incorrect message passing



a: Circulant with in/out degree two



b: Torus of circulants (only a subset of the edges shown for clarity)

Figure 1.7: Modified torus with circulants instead of rings.

The proposed topology combines the circulant and a torus, by proposing a new topology of a torus of circulants, called torus connected circulants (TCC), or briefly 'torculants'. The topology based on a circulant, as shown in Figure 1.7a, is also optimally fault tolerant, however the delays are linear in terms of the number of nodes. An extension to the torus by replacing each row with a circulant adds the benefits of the torus to that of the circulant. Figure 1.7b shows an extension to the torus by using circulants instead of rings along the *x* and *y* axes of the torus.

A torculant of in/out degree two is a normal torus in two dimensions. The rows and columns of the torculant are circulants instead of rings. This torculant is designed with a fixed diameter and optimal fault tolerance including with a very good delay degradation. *This structure then goes through recursive modified line graph transformations based on the number of nodes, of not more than* $\log_d n$ *steps. While each step increases the diameter by at most two, the number of nodes increases by a factor of d. Thus, the diameter remains O(*$\log_d n$*) without modifying the optimal fault tolerance and degree, yielding the best of both the worlds*.

The circulants along the row and column of a torculant are such that each node has a degree *d*. It is designed to ensure a message goes from a node to any other in the same row or same column in two hops each. Thus any node can be reached on the torculant from any other in at most four hops. However, the number of nodes to start off might not fit well in a torus of circulants in which case extra nodes are added. Figure 1.8 shows an example where four extra nodes are added. The sequence of nodes *JBK* along a longitude is broken up to insert the extra node *X* by making the sequence *JXBK*. In the latitude this is inserted after *B* changing the sequence *ABC* to *ABXC*. The final torus ensures that the degree of the extra nodes is maintained and the diameter of this torus is also kept at four.

1

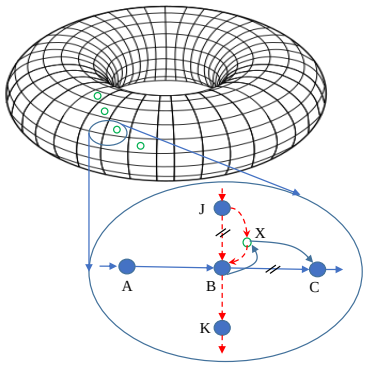


Figure 1.8: 2D Torus of circulants (torculant) with extra nodes

The final number of nodes required is first expressed in base d as follows:
$n = ((..(\ (\ (a_i*d + a_{(i-1)})*d + a_{(i-2)})*d + a_{(i-3)})*d\ldots)*d + a_0$
The number of nodes $a_i*d + a_{(i-1)}$ from the innermost bracket is used to design the initial torculant with diameter four. This torculant is also called the 'seed' or the 'base' graph. Each subsequent bracket is a recursive 'line graph' transformation starting with this torculant as the seed by multiplying the number of nodes by *d* and adding $a_{(j-1)}$ at the j^{th} bracket. At most $((\log_d n) - 2)$ such transformations are required. Each transformation increases the diameter by *at most* two and hence the diameter of the network with *n* nodes is bounded above by $2((\log_d n) - 2) + 4$ which is $2(\log_d n)$. *Thus this design achieves a diameter close to that of the hypercube without the overhead of the high degree*.

The features of this network obtained by the recursive modification of the seed torculant are listed below.

- `Degree:` The degree of the final topology is the same as that of the torculant seed graph. Since the initial graph was regular with degree *d*, the final graph is also regular with degree *d*.
- `Diameter:` The diameter increases by at most two each time the line graph transformation is applied. Since the seed torculant had a diameter of four and the increase was at most two during each transformation, the final diameter is bounded by $2(\log_d n)$.
- `Fault tolerance:` The torculant seed graph is optimally node *and* region based fault tolerant, and hence the final graph is also optimally node *and* region based fault tolerant.
- `Number of nodes:` There is no constraint on the number of nodes that can be in the network, unlike in the mesh, torus or hypercubes where the number is restricted by the topology.
- `Routing:` The shortest path routing in the final topology is dependent **only** on the original torculant seed graph, which is multiple orders of magnitude

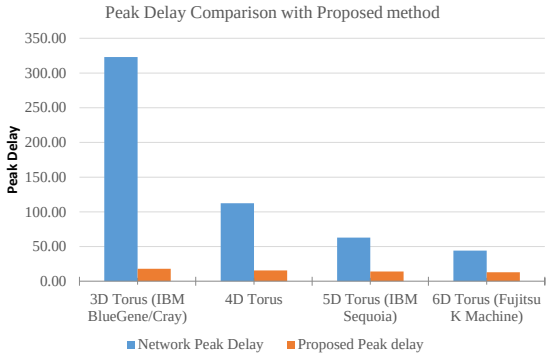


Figure 1.9: Diameter comparison to predict peak delays in various supercomputers to that of the proposed method.

smaller than the final network. This is also true for routing in the presence of node and region faults. This also helps in self-healing of the network.

- `Routing tables:` Since the shortest path routing only depends on the original graph, the routing tables are also orders of magnitude smaller than would otherwise be needed. This allows ease of alternate paths for load balancing, or for faulty node avoidance. This also reduces the memory and power needs in the routing controls. It also enables security features that can be implemented based on the smaller graph and hence more efficient to monitor.
- `Security:` The small routing table which is dependent only on the seed torculant enables some interesting security features to identify misdirected packets.
- `Delay degradation:` By the analysis of the container and the new concept proposed in this study called 'region based container' the delay degradation of the torculant seed graph is one. Thus the final graph also has a peak delay degradation of one despite d-1 node or region failures. This is a very powerful result as it very tightly bounds the delay degradation.

Thus if the original torculant seed graph's specifications have excellent reliability, robustness and security features, these are all automatically available in the final graph without affecting the metrics adversely.

For instance, if the network is to be designed using 10,000,000 nodes, then the comparison of the diameter of various topologies to that of the proposed method are shown in Figure 1.9 with matching degrees. Delays are a function of the number of stages required and would be $D*\frac{\sqrt[D]{n}}{2}$ for a network of n nodes and a D dimensional torus.

In the network architecture of the fastest supercomputer in the world today, the Sunway TaihuLight of ten million nodes, the delays are reduced by utilizing very large number of ports (greater than 256) on the supernodes. This leads to a

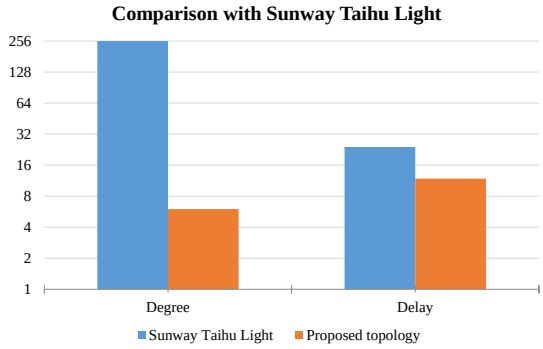


Figure 1.10: Comparison of the world's fastest supercomputer delays with proposed method.

peak delay of 24. The same system can however be connected with the proposed method with a degree only six, without the need for the added hierarchies in only $2*\log_6(256*4*40) = 12$ stages, as shown in Figure 1.10. This reduction in latency has two main effects, the reduction in power and latency as the data has to traverse lesser number of devices along the way.

Another way of looking at this is if a total of 24 stage delay were to be acceptable, then the number of supernodes that could be connected in the Sunway TaihuLight with the proposed topology would be 4^{12} = 16,777,216 instead of 40,960. ***This means the proposed method will enable 400X the number of nodes for the same peak delay in the existing configuration***. This shows the scale of possibilities with the proposed topology, in terms of power, latency and growth potential of future supercomputer networks.

This analysis uses the public architectural details of the Sunway TaihuLight as all details are not yet public knowledge. However, it serves to show that the gains are not just in the reduced delays, but in the power needed for the supernodes, cabinets, and reduced resource contention. The contributions of this research can be categorized into three broad areas of Reliability, Robustness, and Security and will be discussed in the next few subsections.

1.4.2. Reliability

Figure 1.11 compares the degree-diameter tradeoff of the proposed network topology to the existing topologies in use in various supercomputers of the recent past. The comparison is done for 500,000 nodes networked using the existing and proposed topologies. As shown, the proposed topology clearly has the best of both the diameter and the degree, which none of the existing topologies even come close to achieving.

Some of the contributions here can be listed follows.

- There always exist loop and deadlock free shortest paths of at most $2(\log_d n)$.

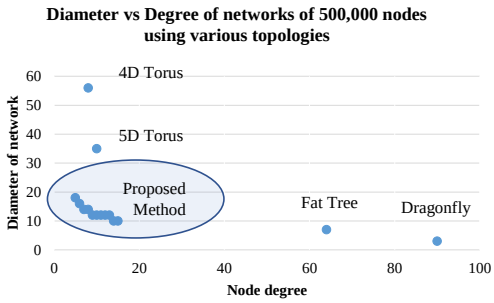


Figure 1.11: Diameter-Degree in recent supercomputer topologies vs. proposed method.

- The number of node disjoint paths available are always equal to the degree of the network, *d* [15].
- The difference in the longest paths with and without faulty nodes is at most one, ensuring very graceful degradation [16].
- The new metric proposed in this study 'Region Based Container' shows that this family of networks can even withstand region failures and still ensure a degradation of only one on its diameter [18].

To put the effectiveness of this method in perspective, some real examples are considered in Figure 1.12. One implementation of IBM's BlueGene/L has 65536 nodes (each node has multiple cores) connected in a 32X32X64 3D torus with each node connected to its six immediate neighbors. This means that these nodes can withstand up to five node failures, and the largest delay will be bounded by 16+16+32 = 64 stages. In comparison, the proposed method would have reduced the number of stages *from 64 to nine, an 86% reduction*, while keeping the node failures allowed the same. The Fujitsu K supercomputer [21] on the other hand has a hybrid topology where a 3D mesh/torus is merged into a 3D torus to give a 6D topology. The proposed topology would have reduced its peak delay *from 36 to 10, a 72% reduction*.

1.4.3. Robustness

Region based connectivity [19] or RBC was a concept introduced in INFOCOM 2006 which alluded to the connectivity of networks when failures are clustered. Real life errors are clustered instead of randomly placed. Examples of this would be hotspots on a chip, or an entire card or a midplane going bad. In cases of networks over much larger areas, manmade or natural disasters will affect entire regions. As such the robustness of a system by looking at region failures instead of random point failures is a very practical metric.

- The network topologies of this work are shown to have optimal RBC. This means that this family of networks is robust to not only *d*-1 node failures, but in fact *d*-1 **region** failures [20]. Each region can have at most $2(\frac{d^{r+1}-1}{d-1})$–1

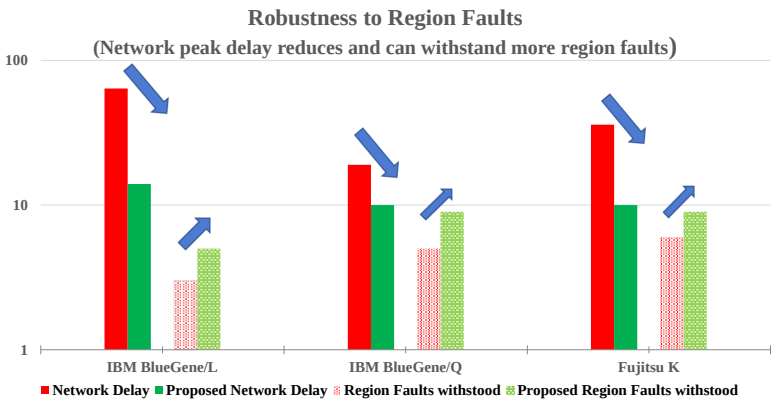


Figure 1.12: Comparison of Delays and Region Fault Connectivity with proposed method.

nodes, where r is the radius of the region *still* the diameter of the network is $O(\log_d n)$. This underscores the robustness of this method. The interesting thing to note here is that hypercubes, unlike the proposed topology, are not optimal in RBC, though they are the best in many other metrics such as diameter, shortest path routing and routing table sizes.

To put this in perspective, consider a network of degree 10 and 100,000 nodes. A fault tolerant network topology would be able to withstand 9 node faults. On the other hand, the proposed network of degree 10 will be able to withstand 9 *region* faults of depth two. This corresponds to at most 221 nodes per region and 9 such regions for a total of 1989 faulty nodes.

Also, the proposed topology would have increased the number of region faults the Fujitsu K supercomputer can tolerate *from six to nine, a 50% improvement*.

Self-healing is an important subject in networks in general. This term refers to the ability of the network to locally reroute paths when intermittently faulty nodes are known. Self-healing leads to increased robustness since this enables systems to work outside of the specifications and still ensure good communication.

- The proposed family of networks, ensures that the self-healed paths will be the shortest paths while ensuring there are no loops or deadlocks.
- The time required to compute the self-healed paths is also $O(\log_d n)$ [22].
- This work also shows how to design with spare nodes to enable the system to run with the desired number of nodes despite a few permanent failures.
- The work enumerates the effort required to re-assemble the hardware to bring the reliability back to specifications with a reduced network size in case of permanent faults.

1.4.4. Security

Routing of messages in the network could end up going along a path that is non-optimal due to a faulty routing or due to malicious behavior of a node. Such misdirection can result in delays, or denial of service due to incorrect routing between multiple malicious nodes. To determine if the previous node is faulty, one needs to be able to determine if the incoming message is being forwarded correctly in a very short time.

- The routing tables required for this family of networks are very small due to the nature of the design and the node naming. The size is dependent on the degree d, of each node of the seed torculant graph, and not on the number of nodes in the final network. There will be at most (d^2 + d - 1) rows and columns in each routing table. The routes are determined not by the full network, but by a very small set of nodes that have an 'influence' on the rest of the nodes. This enables very quick determination of a misdirected message.
- A new protocol 'WISH', proposed in this work [23] enables identification of misdirected messages by using the color of the nodes along the path in $O(\log_d n)$ steps. This is achieved by first finding the shortest path from the sender to the receiver and finding the sum of the colors mod k of all intermediate nodes, where k is the number of colors in the system. The message includes the value (Σc_i) mod k, where c_i is the individual color of each node along the path. In addition, each node keeps track of the running total mod k of the incoming sum as well as that of the colors of the forward path. By analyzing the color of the nodes visited up to the current node (i.e. What I Hear) and the colors of the nodes remaining in the paths (i.e. What I See), and its own color, the node can determine if the path has been misdirected. Information about the previous node that misdirected the path can then be shared with the system to take appropriate actions. The path can then be labeled with the corrected colors and sent to the destination.

1.5. Thesis Outline

This thesis is organized in the following manner. Chapter 2 deals with information on the construction and useful properties of this family of network topologies. This is followed by the reliability of this family of networks and the graceful degradation in the presence of faults which ensures a very tightly bound worst case delay. The work uses the concept of 'containers' of the underlying graph. This section concludes with a new proposal that extends the concept to 'containers of regions'. This is a very powerful metric that can bound the degradation of the delays in the networks in the presence of clustered failures instead of point faults.

Chapter 3 discusses the robustness of this family of networks and shows self-healing can automatically reroute a message along the next shortest error-free path. This rerouting is done without any looping or backtracking. This method can also be used to enable load balancing. The work discusses the bounds on the network modifications to accept a limited number of permanent faults. This section

measures the proposed networks against the metric of region based connectivity. It is shown that these networks are not only optimally fault tolerant, but also optimally region based fault tolerant.

Chapter 4 discusses faulty and possibly malicious routing. The work describes the WISH protocol where small routing tables enable each node along the path to identify the correct path. Each node identifies the shortest path from itself to the destination, and can catch misrouted paths by considering the colors assigned to nodes along the path. This analysis can be done in logarithmic time by making energy-efficient and fast decisions.

Chapter 5 concludes the work with a discussion on future topics that can be added to this research work.

2

Reliability

Papers published in this category:

- IEEE DFTS 2014 : Shortest Path Reduction in a Class of Uniform Fault Tolerant Networks
- ISPAN 2017 **(Best Paper Award):** Tight Bounds in Message Delays Despite Faults in a Class of Line Digraph Networks

Network analysis requires various stringent metrics. The first part of the research dealt with devising the supercomputer network topologies that would enable very low delays without having to increase the number of ports required on each node to connect the neighboring nodes. These networks must be highly reliable in terms of the delays experienced in the presence of faults.

2.1. Design for Optimal Fault Tolerance of Network Topology

The results of the paper ***Shortest Path Reduction in a Class of Uniform Fault Tolerant Networks*** introduced the initial methodology of constructing networks based on the desired number of nodes and the degree of each node. The flow describes a two-step process. The first step is an algorithm to devise the initial graph that will be modified in the subsequent step. The second step describes the recursive method to modify the existing network to produce the subsequent one, until the final desired network is created.

The paper also goes on to show that the diameter of the regular directed network designed is at most $2(\log_d n)$, can have any number of nodes n any degree d, and is the best known result in literature with these constraints. The shortest path between any two nodes can be determined in $O(\log_d n)$ steps with and without the presence of node failures. The shortest paths are guaranteed to be without loops or backtracking.

2.2. Reliable Networks with Graceful Degradation

The specifications for network reliability should include network degradation in the presence of faults. One way of analyzing this is by the use of 'containers' of the underlying graph of the network. Containers help set an upper limit on the delays seen in the presence of faults and hence form a powerful metric. The paper ***Tight Bounds in Message Delays Despite Faults in a Class of Line Digraph Networks*** indicates how the traditional metric of the diameter of a network is insufficient for fault tolerant networks. The *d*-wide diameters of the networks are better suited for this purpose. The paper further shows that the family of networks proposed by this study has a diameter degradation of at most one in the presence of faults. Thus this family of networks not only has one of the best known diameters but also the best degradation in diameter despite the presence of up to *d*-1 node failures. This feature helps in load balancing without significant impact in delays while rerouting by different paths.

Shortest Path Reduction in a Class of Uniform Fault Tolerant Networks

Prashant D. Joshi
Austin, Texas
prashant.joshi@gmail.com

Said Hamdioui
Delft University of Technology
Delft, The Netherlands
s.hamdioui@tudelft.nl

***Abstract*— Shortest path determination in a class of optimally fault tolerant networks designed using modified line graphs is described here. Appropriate node naming allows the shortest paths to be determined in O(log n) steps. This is applicable even in the presence of node failures, without loops or backtracking. The stretch of the network is maintained at the theoretically minimum value possible of one.**

Keywords—Shortest Path in Networks, Line Graphs, Node naming, Connectivity, Diameter of graph, Fault Tolerance

I. INTRODUCTION

Fault tolerant networks are designed to have the ability to withstand some failures without adversely affecting the network performance. The metric being considered in this study is the determination of the shortest paths with and without the presence of faulty nodes.

Past studies of network paths have involved graphs with nodes as computing elements and the edges as communicating links [4-18]. The smallest degree of any node is an upper bound on the maximum tolerable node failures to keep the network connected. A network achieving this is optimally fault tolerant. Knowledge of the network helps get the shortest path between any two nodes. To enable shortest path determination in the presence of faults, the good nodes locally redirect traffic as required. The network topology presented here also has the smallest known diameter for this class of networks in published literature, to the best of the authors' knowledge.

This paper is arranged as follows. Definitions of standard and specific terms used in this paper will be followed by a short preview of prior work, the graph construction, the node naming and shortest path determination with and without the presence of faults. Lastly, possible future work will be touched upon.

II. DEFINITIONS

The terms used in this study relating to graphs can be found in any standard text book on the subject like [1] and [2].

A graph G = (V,E), where V is a set of nodes, and E is the set of edges. The degree of a node is the number of edges incident on that node. In a directed graph (digraph) the indegree (outdegree) of a node is the number of edges in to (out of) that node. A uniform digraph has equal in and out degrees on all nodes.

A path is the sequence of adjacent nodes and intermediate edges from the start node to the destination node. The distance from a node 'u' to a node 'v' is the number of edges along the shortest path from 'u' to 'v'. The diameter of a graph k(G) is the maximum of the shortest paths from node 'u' to node 'v'. The diameter bounds the number of edges required to traverse from any node to any other node. Two nodes whose distance is equal to the diameter are called diametric nodes.

The node connectivity of the graph is the minimum number of nodes, when removed, disconnects the graph. The fault tolerance is one less than the connectivity. An optimally connected graph can tolerate up to d-1 node faults, where d is the minimum degree of any node.

A D_d digraph is a uniform digraph (V, E) such that V is a set of nodes {0, 1, …(v-1)} and E = {(a,b) | a, b elements of V; b = (a + k) mod v, $1 \leq k \leq d$}. An edge of the type x-hop is an edge of the D_d graph from any node y to the node (y+x) mod v. The diameter of a D_d graph is bounded by $\lceil((|V|-1)/d)\rceil$.

A line graph of a digraph G = (V, E) is L(G) = (V1, E1) such that V1 = E and E1 = {(a,b) | a = (u1, u2) is an element of E and b = (u2, u3) is an element of E }. The predecessor set P(u) and the successor S(u) of a node 'v', an element of V, are defined as P(v) = {u | (u, v) is an element of E} and S(v) = {w|(v,w) is an element of E}.

An Extended Line Graph EL(G) of a uniform digraph G=(V,E) of degree d and connectivity d, is L(G) with t additional nodes, t<d so that the number of nodes of EL(G) = n*d + t, such that the EL(G) also has degree d, connectivity d, and diameter $k(EL(G)) \leq k(G) + 2$. The construction of the EL(G) is described later.

The 'stretch' is the worst ratio of the shortest path determined to the actual shortest path in the graph, between two nodes.

III. PRIOR WORK

Computer network design for reliability and efficiency has led to many practical fault tolerant networks in use today. Reducing the diameter with and without node faults, increasing connectivity, enabling fast routing algorithms, self-healing, and re-configurability are some of the metrics used. Researchers in [4]-[6] studied generalized hypercubes, restricting nodes to powers of 2, while those in [7]-[9] concentrated on De Bruijn graphs and modifications. In some cases the degree of graphs

978-1-4799-6155-9/14/$31.00 ©2014 IEEE

was compromised as in [6] when the number of nodes is a prime number. Most studies concentrated on keeping the fault tolerance optimal and the diameter proportional to $\log_d n$. The work in [11] also has a suboptimal diameter and restrictions on its degree, while [12] had worse diameter than the method proposed. Properties of line graphs have been studied in other papers like [14], [19] and [20] where the Bruijn-Kautz graphs were analyzed.

This paper extends the work in [23] where the details of the graph construction were covered resulting in the best in class diameters for uniform, directed, and optimally connected graphs, with no constraints on the number of nodes or degree. Networks in this class from previous studies such as [4]-[6], [10]-[12], [18], have a suboptimal connectivity, larger diameter or restriction on the number of nodes or degrees. The work in [12] did not determine factors like the shortest path, self-healing, stretch factors, or re-configurability, and the diameter was marginally worse. This paper will go into the details of node naming and how to determine the shortest path, with and without node failures. The effort estimate for the shortest path determination is also outlined.

Studies in [27] have shown that the maximum stretch of random networks cannot be less than 3, while the average comes closer to 1. This work shows a stretch of 1 for this class of networks, which is the theoretically least possible value.

IV. Network Construction, Node Naming and Shortest Paths

A. Graphs G, L(G), and EL(G) construction

The line graph of a uniform and optimally connected digraph maintains the degree and connectivity, while the diameter increases by one and the number of nodes becomes n*d [17]. This study extends the concept of a line graph. Given the uniform digraph G=(V,E) of degree and connectivity d, and diameter k(G), we generate EL(G) with n*d + t nodes, $0 \le t < d$ without modifying the degree or the connectivity, though the diameter could go up by 2. These t nodes are referred to as X-nodes, and the other nodes are the non-X-nodes. First obtain the line graph L(G) and a completely connected graph of the t nodes. L(G) has connectivity and degree d, while the graph of the t nodes only has a degree t-1. To merge these graphs to make a uniform digraph, it requires d-(t-1) more edges to and from each X-node.

For some t unique nodes from G: N_i $1 \le i \le t$, randomly pick unique d-(t-1) predecessor nodes P_1, P_2, …$P_{d-(t-1)}$, and d-(t-1) successor nodes S_1, S_2, …$S_{d-(t-1)}$. Since the degree of each node is d, such unique nodes always exist on each of the t unique nodes chosen.

Now for each chosen N_i of G, $1 \le i \le t$, removed it's edges (P_j, S_j) $1 \le j \le d-(t-1)$ identified above, from L(G), and instead add the edges (P_j, x_i) and (x_i, S_j). This maintains the degree of the nodes of L(G), and increases the degree of the X-nodes from (t-1) to d-(t-1)+(t-1)=d. If t=0, then EL(G) is the same as L(G).

The so constructed graph is the required EL(G). Fig. 1 shows examples of a G, L(G), and EL(G) graphs.

The connectivity being maintained is proved below. If there are no X-nodes in all the paths between two non-X-nodes, then this part of the proof is trivial, since L(G) is known to maintain the connectivity [17]. If there are X-nodes, the same X-node cannot be on two paths by construction. Hence two non-X-nodes always have d node independent paths between them. Two X-nodes have a direct edge, so are trivially connected with d removed nodes. In the case of a path from an X-node to a non-X-node, even if d-1 X-nodes are removed, there still exists a path to a non-X-node and then the first case ensures connectivity. This logic in reverse applies to a path from a non-X-node to an X-node.

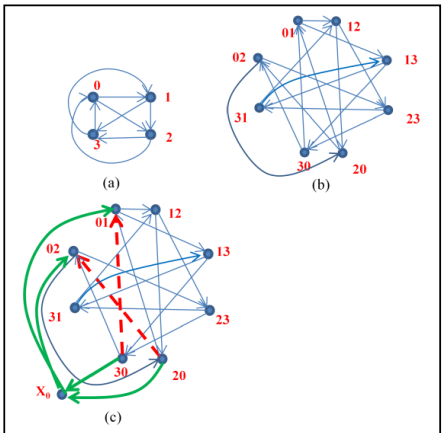


Fig. 1 (a) Graph G. (b) Graph L(G). (c) Graph EL(G) with 1 X-node.

B. Final graph construction

For a uniform digraph of degree d and n nodes, we can write the number 'n' to base d, as below for each $a_j < d$.

$$n=((\ldots((a_i d + a_{(i-1)})d + a_{(i-2)})d + a_{(i-3)})d\ldots)d+a_0 \qquad (1)$$

Equation (1) shows the EL(G) transformation being applied recursively on the graph of the inner bracket to generate the current bracket. Thus if we can get a good graph for the inner most bracket, then each subsequent bracket simply is the EL(G) of the previous graph, where the diameter of the next graph increases by at most 2.

The problem is now reduced to constructing a good 'base' graph for the innermost bracket of $a_i d + a_{(i-1)}$ number of nodes. The aim is to construct the base graph with a diameter bounded by 4. The details of the base graph design and proof that its diameter is bounded by 4, with a connectivity of d, are proved in [23]. With the innermost graph designed with diameter at most 4, each bracket on the outside is essentially a recursive EL(G) transformation as depicted in Fig. 2.

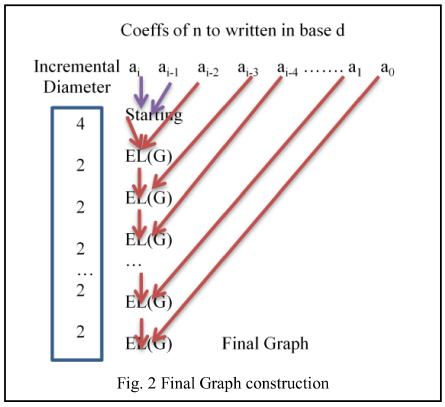


Fig. 2 Final Graph construction

The construction of the final graph starts with the base graph of diameter 4 and with each EL(G) transformation the diameter increases by 2, maintaining the degree and connectivity. The final diameter is 4 plus two times the number of times the EL(G) transformation is applied which is $\lceil \log_d n \rceil - 2$, and hence the final diameter is bounded above by $2*(\lceil \log_d n \rceil - 2)+4 = 2*\lceil \log_d n \rceil$. This result is the best known diameter for the class of optimally fault tolerant directed networks where there is no restriction on the number of nodes or the degree, to the best of the authors' knowledge.

C. *Node naming and shortest path example*

Any node formed after taking the line graph of a graph G, is named by concatenating the node names of the source and sink of the edge in G. For example if there is an edge between nodes named X and Y, where X and Y can be any sequence of characters, then the resulting node of this edge in the line graph would be the string X concatenated by Y. X is the left predecessor and Y is the right predecessor of this node XY. This identifies the descendant nodes of the base graph, among those of the final graph. Fig. 3 gives an example of the naming.

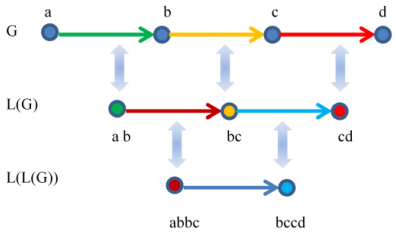


Fig. 3. Node naming with successive L(G) transformations

For example, let us consider Fig. 4 where a graph of n=14 and d=2 is constructed based on the described method above, and try to trace a shortest path between the nodes 0110 to 2112.

0110→ ? →2112

This implies that in the penultimate EL(G) graph, we have to find a path from 10 to 21, and then so on, from 0 to 2 as shown below.

0110 **(10→ ? →21)** 2112

0110(10**(0→ ? →2)**21)2112

Since 0→2 exists in the base graph, we have been able to go down recursively until the base graph to find the required path. Now we will trace back up to retrace the nodes at each step.

0110→(10→02→21)2112

0110→1002→0221→2112

14 base 2 = 1110
Base graph (3 nodes)
0→1,2 1→2,0 2→0,1
L(G) (6 nodes)
01→12,10 02→20,21 12→20,21
10→01,02 20→01,02 21→12,10
EL(G) (7 nodes)
01→x x→12
21→x x→10
01→~~12~~,10 02→20,21
12→20,21 10→01,02
20→01,02 21→12,~~10~~
L(G) (14 nodes)
01x → x12, x10 x12 →1220, 1221
21x →x12, x10 x10 →1001, 1002
0110→ 1001, 1002 0220 → 2001, 2002
0221 → 2112, 21x 1220 → 2001, 2002
1221 → 2112, 21x 1001 → 01x, 0110
1002 → 0220, 0221 2001 → 01x, 0110
2002 → 0220, 0221 2112 → 1220, 1221
Diameter is guaranteed to be bounded above by $2*\lceil \log_2 14 \rceil$ and in this case the diameter is 4.

Fig. 4 Example of a graph of 14 nodes with degree 2. The arrow indicates that the node to the left of the arrow has edges to node(s) after the arrow

If 0221 is known to be faulty, then avoiding either the edge 02 or 21 will never result in the node 0221 in use in any path. Hence we could then take 0→1→2 at the base graph level and thus get the path: 0110(10(0→1→2)21)2112

0110→1001→0112→1221→2112 thus avoids the faulty node. It is possible that we might not have to go down to the base graph since an intermediate level might have a direct edge and in which case the path is between non-diametric nodes.

D. *Field of Influence, Routing Table for shortest path in $O(\log_d n)$ steps.*

This leads us to the concept of 'field of influence' of a node of the base graph upon the nodes in the final graph. If a node in the final graph has the regular expression of a node of any previous transformation, then that node has a field of influence on the final node. As such, in the determination of the shortest path, if all final nodes know which nodes are faulty, then they know which regular expressions to avoid in the shortest path determination.

In networks, a routing table is a data structure in the form of a matrix, which lists the routes to all the destinations, next node hope information and other information. The routing table is generated out of the knowledge of the topology of the entire network which determines statically the next node, based on the current node and the final destination. These tables are used for packet forwarding.

Consider a typical table of n rows, one for each destination. To conserve memory, each node only keeps track of the next hop information from itself, based on the final destination, along with some other information. If there is a need to store multiple paths, there could be multiple next node options for each row's final destination.

base graph	EL(G)	EL(G)	EL(G)
a	ab	abbc	abbcbccd
b	bc	bccd	bccdcdde
c	cd	cdde	cddedeef
d	de	deef	deefeffg
e	ef	effg	*effgfggh*
f	fg	fggh	fgghghhi
g	gh	ghhi	ghhihiij
h	hi	hiij	hiijijjk
i	ij	ijjk	
j	jk		
k			

Fig. 5. Field of Influence and shortest path determination

This contrasts with the tables in the class of networks in this study. Consider the shortest path between two nodes say, abbcbccd to hiijijjk in Fig. 5. A node 'y' below another 'x' along a column means the edge x→y exits at that EL(G) level, and each column shows the result of the EL(G) of the previous column. Only the shortest path between the rightmost predecessor of the source node 'd' and the leftmost predecessor of the sink node 'h' is required to be known in routing tables. These tables do not need to be of n rows, but of at most d^2-1 rows (maximum number of nodes in the base graph). From the routing table, once the shortest path (shaded in yellow) from the first column of the base graph is known to be d→e→f→g→h, then the full path is automatically known due to the node naming. In addition, this knowledge enables each node to know how other nodes will behave, thus helping in identifying nodes that are not behaving correctly.

If it is known that effgfggh is faulty, then to avoid this node, the path in the base graph from d →…→h can avoid at least one of the edges: ef, fg, gh and by doing so this regular expression will not be present in any of the nodes of the path in the final graph thus ensuring the shortest path is still used despite the faulty node.

Consider again the example graph of Fig 4 of 14 nodes. The routing table to be stored in each of these 14 nodes is shown in Fig. 6.

To→ From↓	0	1	2
0	n/a	1,2	2,1
1	0,2	n/a	2,0
2	0,1	1,0	n/a

Fig 6. Sample routing table. Top row is the final destination, left column is the current node and the entries are prioritized next nodes.

This information is sufficient for us to determine the shortest path from any node to any other node in the final graph of Fig 4. The reason is that the node naming allows us to determine the shortest path, and if the recursively going back on the EL(G) columns, we end up at the base graph, we use the above routing table. If we see a path in an intermediate level, we don't even need the routing table.

Notice also that although the final graph might contain any number of nodes, the routing table needs to have only ($a_i d + a_{(i-1)}$) rows and columns (at most d^2-1). So in the example of Fig. 4 and Fig. 6, the same routing table would suffice if the final graph had more EL(G) transformations, since the base graph and its shortest paths would be the same. Each row indicates the starting node, and the column number represents the final node of the base graph. Every entry contains the next node to take, to get the shortest path. In case of known faulty nodes, the routing table will allow avoiding specific edges of the base graph, based on the regular expression of the faulty node.

Notice that this process ensures there are no loops or backtracking in the shortest path determination and the shortest path is always obtained. This means that the stretch of these graphs is always 1. This equals the theoretically minimum value possible. The ratio of the maximum amount of storage required to the number of nodes grows smaller as the number of nodes grows, since the maximum amount of storage is a fixed quantity irrespective of the number of nodes.

If the faulty nodes are having transient faults and the status at a given time is known to all the nodes, the routing can be done by avoiding these nodes. On the other hand, if this transient behavior is seen to be more permanent, then a decision can be made to make it as a permanent fault thus enabling actions to reconfigure the network if required.

In [23] it has been shown that the base graph of diameter 4 will see its diameter change by at most 1 in the presence of 1 node fault. This means that the increase in diameter of the final graph is also bounded by 1. Thus the determination of the shortest path is based off of finding the path between two nodes in a routing table of size at most d^2-1. However since the diameter is at most 4, the number of steps are limited to 4 + ($\log_d n - 2$). Thus the number of steps in the presence of one fault is limited to 5 + ($\log_d n - 2$). With more faulty nodes, the number of steps to determine the shortest path goes up in small

amounts and the overall number of steps is $O(\log_d n)$ which is due to the number of the EL(G) transformations for the final graph.

At this stage we have shown that this method using EL(G), and proper node naming enables us to find the shortest path with and without node faults in $O(\log_d n)$ steps.

To do a proper load balancing, it is possible to separate the entries in the routing table such that the next nodes which result in the same distance are grouped together. As an example say the entry from base node 'x' to base node 'y' contains : a,b; e,f,g; p; q,r. This would imply that to go from x to y, the best next nodes are a or b, as they result in the shortest paths of equal length. If either of these are not permissible, then the next shortest path is by taking either e or f or g. And so on, with p and then with q or r.

Notice that this table is generated a priori, and not a dynamically changing one unless a node is deemed to be permanently faulty and needs to be removed from the routing tables. Also, from [23] the diameter of the base graph is 4, as a result, the number of steps required to find the shortest path is also 4, at this level.

Consider the following steps, where a step means either recursively going back or up the graph generation by the EL(G) transformations, or finding the next node in the table.

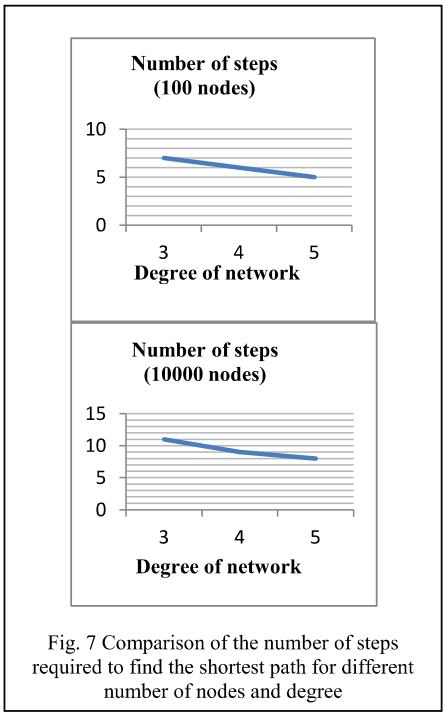


Fig. 7 Comparison of the number of steps required to find the shortest path for different number of nodes and degree

1. Recursively go back from the EL(G), finding the right predecessor of the start and left predecessor of the destination nodes at that level of EL(G).
2. If an edge exists between these two, at that level, then stop the recursive steps backwards, and instead go back up the recursive steps and you have the shortest path.
3. If the base graph is reached, then use the routing table to find the shortest path, and then go up recursively along the EL(G) and the shortest path is known.
4. If a faulty node is known to be on the path so found, avoid the edge with that regular expression and we still have the next best path.

A comparison of the number of steps required to find the shortest path without any faults is shown in Fig. 7. It is illustrative to note that these numbers are $O(\log_d n)$ as against the number being O(n) in the case of a normal routing table.

V. Conclusion

In this paper we described a class of uniform directed networks constructed based on Extended Line Graphs, which resulted in the best in class diameters and optimal connectivity of these networks. Further, we showed a novel node naming method such that the upper bound on the routing table size is independent of the size of the network, which enables $O(\log_d n)$ steps to determine the shortest path in the network, with or without the presence of node faults. The stretch of the network is the theoretically best possible value of one. The shortest path determination problem is reduced to finding the shortest path in at most d^2-1 nodes independent of the number of nodes. Future work will look into extending this research to help isolate faulty nodes, and estimate the effort to reconfigure the network in the presence of permanent faults.

References

[1] C.Berge, Graphs and Hypergraphs. Amsterdam, The Netherlands: North-Holland, 1973.

[2] A.J. Hoffman and R.R. Singleton, On Moore graphs with diameter 2 and 3, IBM J. Res. Develop. 4 (1960) 497–504.

[3] W. T. Tutte, A family of cubical graphs, Proceedings of the Cambridge Philosophical Society, 43 (1947) 459–474.

[4] J. R. Armstrong and F. G. Gray, "Fault diagnosis in Boolean n-cube array of microprocessors," IEEE Trans. Comput., vol. C-30, pp. 590-596, Aug. 1981.

[5] J. Kuhl and S. M. Reddy, "Distributed fault tolerance for large multiprocessor systems," in Proc. 7th Annual Symposium Computer Architecture, May 1980

[6] L. Bhuyan and D.P. Agrawal, "Generalized hypercube and hyperbus structure for a computer network," IEEE Trans Computers, vol. C-33, Apr. 1984

[7] M. L. Schlumberger, "DeBruijn communication networks," Ph.D. dissertation, Stanford Univ., Stanford, 1974.

[8] D.K. Pradhan and S.M. Reddy, "A fault tolerant communication architecture for distributed systems" IEEE Transactions Computers vol. C-31, Sept. 1982.

[9] D. K. Pradhan, Z. Hanquan, and M. L. Schlumberger, "Fault tolerant multibus architecture for multiprocessors," in Proc. 14th Int. Conference on Fault-Tolerant Computers, 1984, pp. 400-408.

[10] M. Imase, T. Soneoka, and K. Okada, "Connectivity of regular directed graphs with small diameters" IEEE Transactions on Computers, vol. C-34, March 1985.

[11] U. Schumacher, "An algorithm for k-connected graph with minimum number of edges and quasiminimal diameter," Networks, vol. 14, 1984.

[12] A.Sengupta, P. D. Joshi and S. Bandyopadhyay,"A Synthesis Approach to Design Optimally Fault Tolerant Network Architecture", IEEE Transactions on Computers, vol.40, January 1991.

[13] Daniela Ferrero and Carles Padro, "Connectivity and fault-tolerance of hyperdigraphs", Discrete Applied Mathematics, 2002.

[14] M. A. Fiol, I. Alegre, and J. L. A. Yebra, "Line digraph iterations and the (d . k) problem for directed graphs," in Proceedings of the 10th International Sympmosium on Computer Architecture, Stockholm, Sweden, 1983.

[15] M. A. Fiol, A. S. Llado, and J. L. Villar, "Digraphs on alphabets and the (d.N) digraph problem," Ars Combinatoria, vol. 25C, pp. 105-122, 1988.

[16] M. A. Fiol, J. L. A. Yebra, and I. Alegre, "Line digraph iterations and the (d . k) digraph problem," IEEE Transactions on Computers, vol C-33, pp. 400-403, May 1984.

[17] S.M. Reddy, J.G. Kuhl, and S.H. Hosseini, "On digraphs with minimum diameter and maximum connectivity," in the Proceedings of the 20th Annual Allerton Confernce, Oct 1982.

[18] D.K. Pradhan, "Fault tolerant multiprocessor link and bus network architecture," IEEE Transactions on Computers, vol. C-34, Jan. 1985

[19] Liu, S. Trajanovski, P. Van Mieghem, "Reverse Line Graph Construction: The Matrix Relabeling Algorithm MARINLINGA Versus Roussopoulos's Algorithm", Delft University of Technology submission to xrXiv.org. October 2010.

[20] J. Naor and M. B. Novick, "An efficient reconstruction of a graph from its line graph in parallel." J. of Algorithms, 11(1): 132-143, 1990.

[21] J. Suurballe and R. Tarjan, "A quick method for finding shortest pairs of disjoint paths," Networks, vol. 14, pp. 325–336, 1984

[22] Dahai Xu, Yang Chen, Yizhi Xiong, Chunming Qiao, Xin He, "On the Complexity of and algorithms for finding the shortest path with disjoint counterpart", IEEE/ACM Transactions on Networking, vol. 14, No. 1, February 2006.

[23] Prashant D. Joshi, Said Hamdioui, "Modified uniform line digraphs with optimal connectivity and small diameteres", Forty-Fifth Southeastern International Conference on Combinatorics, Graph Theory and Computing, 2014.

[24] Amitabh Trehan, "Algorithms for Self-Healing Networks", Ph.D. dissertation, University of New Mexico., New Mexico, 2010.

[25] Alper Mizrak,Yu-Chung Cheng, Keith Marzullo, Stefan Savage, "Fatih: detecting and isolating malicious routers", in the Proc. of The International Conference on Dependable Systems and Networks, 2005.

[26] Robert Poor, Charlotte Auburn and Cliff Bowman,"Self Healing Networks", ACM Queue, May 2003.

[27] Mihaela Enachescu, Mei Wang, Ashish Goel, "Reducing Maximum Stretch in Compact Routing", in the Proc. of IEEE INFOCOM, 2008.

[28] Dmitri Krioukov, Kevin Fall, Xiaowei Yang, "Compact Routing on Internet-Like Graphs", in the Proc. of IEEE INFOCOM, 2004

[29] Aifei Zhong, Srehari Nelakuditi, Yinzhe Yu, Sanghwan Lee, Junling Wang, Chen-Nee Chuah, "Faulure inferencing based fast rerouting for handling transient link and node failures", in the Proc. of IEEE INFOCOM, 2005.

[30] Saia, J., Trehan, A., "Picking up the pieces: self-healing in reconfigurable networks" IEEE International Symposium on Paralleland Distributed Processing, 2008.

2017 14th International Symposium on Pervasive Systems, Algorithms and Networks & 2017 11th International Conference on Frontier of Computer Science and Technology & 2017 Third International Symposium of Creative Computing

Tight Bounds in Message Delays Despite Faults in a Class of Line Digraph Networks

Prashant D. Joshi
Cadence Design Systems
Austin, USA.
joship@cadence.com

Arunabha Sen
School of CIDSE
Arizona Status University
Tempe, USA
asen@asu.edu

D. Frank Hsu
Department of Computer and Information Science
Fordham University
New York, USA
hsu@fordham.edu

Said Hamdioui & Koen Bertels
Computer Engineering, TU Delft
Delft, The Netherlands
shamdioui@tudelft.nl ; k.l.m.bertels@tudelft.nl

***Abstract*— Communication networks must be designed to withstand failures. The robustness of a network needs to account for graceful performance degradation with multiple failures in the network. The traditional delay metric of the diameter of the network is insufficient in this regards to represent the message passing delay in the presence of faults. This paper explores the d-wide diameter of a class of regular extended line digraphs which bound the delays despite faults. By studying the *container* of the network's underlying graph, a measure of the distance between nodes through multiple node disjoint paths is obtained. This study shows that the class of networks designed by extended line digraphs has very tight upper bounds on the d-wide diameter and is just one more than the normal diameter. Thus despite d-1 node failures the worst case delay is just one more than that of the original network. This tight bound has very powerful uses in areas of enabling load balancing and bounded delays despite multiple network faults.**

Keywords—Fault Tolerance, Routing delays, network diameter, graph containers, d-wide network diameter, load balancing, line digraphs

I. INTRODUCTION

Traditional network survivability deals with the connectivity of the underlying graph with node or edge failures. These could be any networks like transportation, waterways, power distribution or computer and communication networks. The metric of delay of message passing is measured by the number of edges between the source and sink nodes. The maximum delay along the network has been widely studied by the diameter of the graphs that represent the networks. A k-connected network can withstand k-1 faulty nodes. This however does not give any measure of the new delay that would be encountered by the remaining network in message passing. Design of critical networks needs to account for these situations and knowledge of upper bounds on the delays will enable proper network performance expectations and planning under various degrees of failures.

Networks based on line graphs have been studied for their properties of shortest delays, optimal connectivity and loopless routing. Fiol, Alegre, and Yebra [2] first studied the behavior of the diameter and the average distance between vertices of the line digraph of a given digraph. Fiol, Llado, and Villar [3] presented a solution to the (d, N) digraph problem and showed the ability to expand or condense the number of nodes in digraphs. Design of regular digraphs of any size starting from a seed digraph, using recurring extended line digraphs, with low diameters and optimal connectivity has applications in reliable networks [5, 6, 7]. The concept of *d-wide diameter* of a network was introduced by Hsu [8] and there are many examples of work based on this that linked connectivity and diameter in the presence of faults [9, 10].

To the authors' knowledge no study has ever been done of containers on extended line digraphs. This paper integrates the work on such digraphs with that of containers to show the graceful degradation seen by networks built on these digraphs. The term digraph and network is used interchangeably in this paper. The terms source and sink nodes will also be used interchangeably to represent the starting node and the ending node of a path respectively.

The resulting regular networks are very versatile in that the delays without faults are very small. In addition, these are optimally fault tolerant in that they can withstand d-1 failures where d is the regular degree of the digraph, and as this work shows, the degradation in the diameter is bounded by one.

The paper is organized as follows. Section II deals with a short description of some terms used. Section III discusses some lemmas required for the design of the seed digraph. The *d-wide* diameter calculation of the networks is discussed in

2375-527X/17 $31.00 © 2017 IEEE
DOI 10.1109/ISPAN-FCST-ISCC.2017.59

Section IV along with the implications towards load balancing with bounded penalty. Finally we conclude in Section V with some suggestions on possible future extensions.

II. Containers And Extended Line Digraphs

In this section we will review the terms used throughout the paper. Most of these can be found in West [1] and other works on *containers* [9] and extended line digraphs [5], and some are briefly described here also. A digraph $G = (V, E)$ has $n = |V|$ nodes and (p, q) is an element of E if there is a directed edge from the node p to node q. The node p is the predecessor of q, and q is the successor of p. The indegree (correspondingly outdegree) of a node is the number of edges incident into (correspondingly out of) that node. In a regular digraph the indegree and outdegree of all nodes are equal to the degree d. A path from a node p to node q is a sequence of adjacent edges that start from p and end in q. Two node disjoint paths from p to q have no common node except for p and q. A digraph is strongly connected if any node can be reached from any other node in it. The connectivity of a digraph is k if the removal of any $k-1$ nodes still keeps the remaining digraph strongly connected, and the removal of some specific k nodes results in the digraph becoming non-strongly connected. The connectivity is obviously bounded by the minimum degree of any node in the digraph. If a digraph achieves this connectivity, it is called an optimally connected digraph. The distance between two nodes is the number of edges in the shortest path between them. The diameter $k(G)$ of the digraph is the largest value of the distance between any two nodes of the digraph.

For some nodes $x, y_1, y_2, \ldots y_w$ of a graph G without self-loops or multiple edges where w is a positive integer and x is not equal to y_i, for any i, a collection of internally node disjoint paths from x to $y_1, y_2, ..y_w$ one for each y_i, is defined as a *star container* from x to $y_1, y_2, \ldots y_w$. In case any node y_r is repeated r times then the *container* needs to have r internally node disjoint paths from x to y_r also. In the special case where $r = w$ and hence $y_1 = y_2 = \ldots y_w$, the *w-star container* is called a *w-wide container* from x to y. The maximum length of the paths in the *container* is the length of the *container* and w is the width of the *container*. The *w-star* distance from x to $y_1, y_2, .. y_w$ is the minimum of all possible *container* lengths from x to $y_1, y_2, .. y_w$ and is denoted by $d(x; y_1, y_2, \ldots y_w)$. The *w-star* diameter of the graph G is defined to be the maximum of $d(x;y_1, y_2, \ldots y_w)$ for all possible combinations of x, and y_i and is denoted by $D_w(G)$.

In the special case of $y_1=y_2= \ldots y_w$, the *w-star* distance becomes the *w-wide* distance and is written as $d_w(x,y)$. When we take the *w-wide* distance between all distinct pairs of nodes in the graph x and y, we obtain the *w-wide* diameter of the graph denoted by $d_w(G)$. It is interesting to note that the traditional definition of the diameter of the graph is now just a special case of the definition of the *w-wide* diameter, when w=1.

For undirected graphs the above definitions hold without issue, but for digraphs the above would hold only for the paths starting from x to the various y_i nodes. We hence define a new type of container called *w-star-prime* which differs from *w-star* in that all the paths are now *to* x from the chosen y_i nodes. Hence similar to the w-star definitions, the *w-star-prime* length is the longest distance from any of the y_i nodes to x. The *w-star-prime* distance is the minimum length of all possible *w-star-prime* containers from the y_i nodes to x and is denoted by denoted by $d'(x; y_1, y_2, \ldots y_w)$. The *w-star-prime* diameter of the digraph G is defined to be the maximum of $d'(x;y_1, y_2, \ldots y_w)$ for all possible combinations of x, and y_i and is denoted by $D'_w(G)$. Fig. 1a gives examples of 3-star and 3-star-prime containers.

A line graph of a digraph $G = (V, E)$ is $L(G) = (V1, E1)$ such that $V1 = E$ and $E1 = \{(a,b) \mid a = (u1, u2)$ is an element of E and $b = (u2, u3)$ is an element of E $\}$. The diameter of the line digraph $D(L(G))$ is at most one more than the diameter of G. Also, from [11] the connectivity of $L(G)$ is maintained as the same as that of G.

An extended line digraph $EL(G)$, of a regular digraph $G=(V,E)$ of degree d and connectivity d, is $L(G)$ with t additional nodes, t<d so that the number of nodes of $EL(G) = n*d + t$, such that the $EL(G)$ also has degree d, connectivity d, diameter $k(EL(G)) \leq k(G) + 2$. [5]

A D_d digraph is a regular circulant digraph (V, E) such that V is a set of nodes $\{0, 1, (n-1)\}$ and $E = \{(a, b)$ s. t. a and b are elements of V; $b = (a + k) \bmod n$, $1 \leq k \leq d\}$. This digraph is known to be d-connected, and have a diameter bounded by $\lceil (n-1)/d \rceil$. It can also be shown easily that the *d-star*, *d-star-prime* for the case where the y nodes are all consecutive (mod n) is $\lfloor (n/d) \rfloor + 1$. The *d-wide* diameter is also $\lfloor (n/d) \rfloor + 1$. Fig. 1b shows an example of a D_2 digraph with five nodes.

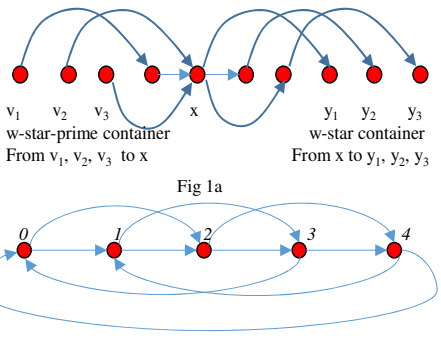


Fig. 1a: Examples of 3-star-prime and 3-star containers

Fig 1b: Example of a D_2 digraph with five nodes

III. Seed Digraph Construction And Some Proofs

In this section we will prove some lemmas which will be used in the next few sections.

Since a D_d digraph can be assumed to be placed with all nodes in a circle and each node having an edge to the next d nodes clockwise in front of it, if we prove a case for node 0, it will be applicable for all nodes.

Lemma 1: In D_d digraphs, any set S of consecutive w nodes, $1 \le w \le d$ can connect to another set T of w consecutive nodes with w node disjoint paths such that each path's start is in S and end is in T. The length of the paths will be either $\lfloor (n/w) \rfloor$ - 1 or $\lfloor (n/w) \rfloor$.

Proof: A short proof is by taking the edge $i \rightarrow (i+w)$ from each node i in S, and continuing hops of size w, each path will reach a node in T in $\lfloor (n/w) \rfloor$ hops if the number of nodes in the digraph is a multiple of w, else it will take one more hop. Since each path starts from a different node and jumps by w, no two nodes on the w paths will be the same.

Q.E.D

An example is show in Fig 2.

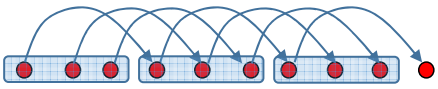


Fig. 2 Example of Lemma 1 where number of hops is $\lfloor (n/w) \rfloor$. S is the first three nodes and T the last three nodes.

Lemma 2: A D_d digraph has a diameter of $\lceil (n-1)/d \rceil$.

The proof of this simply follows from the fact that from the source node, any node can be reached in $(n-1)/d$ hops if $(n-1)$ is a multiple of d, else it will take one more hop.

Q.E.D

Lemma 3: A D_d digraph has a w-star diameter $\lfloor (n/d) \rfloor + 1$ when the y_d nodes are consecutive.

Lemma 4: A D_d digraph has a w-star-prime diameter of $\lfloor (n/d) \rfloor + 1$ when the y_d nodes are consecutive.

The proofs for Lemma 3 and Lemma 4 follow from Lemma 1 since there is one additional edge from the source node to the d consecutive nodes for Lemma 2, and from d consecutive nodes to the sink node for Lemma 3.

Q.E.D

The goal of this work is to be able to design a regular directed network of all indegrees and outdegrees d, given any number of nodes n, and prove that the resulting network has some very desirable properties of maximum delays in message passing, with and without the presence of faults by taking into consideration the containers of these digraphs. The design of the network follows a similar flow as previous studies on extended line digraphs. However, the design of the seed digraph is simplified.

The flow is recapped here for clarity. To design a network of degree d and n nodes, write the number n to base d, as below with each $a_j < d$.

$$n = ((..((a_i d + a_{(i-1)})d + a_{(i-2)})d + a_{(i-3)})d \cdots)d + a_0 \qquad (1)$$

Note that in Equation (1) each set of brackets is essentially performing an EL(G) operation on the next inner bracket. Recursively this can continue until we get to the innermost bracket which will require a separate construction. Thus the problem reduces to finding a good digraph for the innermost bracket which will be called the 'seed digraph'.

The EL(G) is obtained by taking the line digraph of the current digraph and then adding $a_j < d$ nodes. The detailed steps of this are given in Section IV. This is done by creating a separate completely connected digraph on a_j nodes. This digraph has a degree a_j-1 and since the regular degree is d, each node needs $d - (a_j\text{-}1)$ edges in and out. For a nodes of G, take $d - (a_j\text{-}1)$ predecessor and successor pairs which will form an edge in $L(G)$, and break the edges and insert one of the nodes from the completely connected digraph on a_j nodes. Repeat this for a total of a_j nodes from G. The resulting digraph now has an indegree and outdegree d on all nodes. The L(G) increases the diameter by one, and the insertion of the a_j nodes adds one more at most to the diameter, hence $k(EL(G))$ is at most two more than $k(G)$. Also, the connectivity of $EL(G)$ is maintained as that of G [5].

To design the seed digraph such that the diameter is bounded from above by four, consider the following two cases. Let s represent the number of nodes required from the seed digraph. If the innermost bracket has $a_i = 1$ and $a_{i-1} = 0$, then let

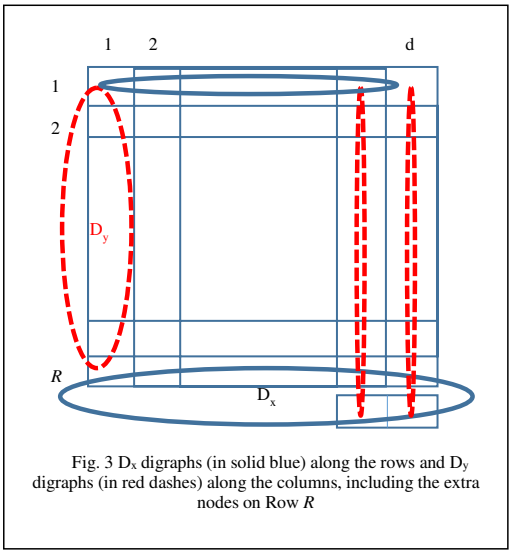


Fig. 3 D_x digraphs (in solid blue) along the rows and D_y digraphs (in red dashes) along the columns, including the extra nodes on Row R

s include the next bracket also and hence design the seed digraph with $s = d^2 + a_{(i-2)}$ number of nodes. Note that in this

case the number of times EL(G) is applied will be smaller by one.

Case 1: $s < 4d+2$. In this case a simple D_d digraph suffices. Note that when d equals two or three, this is the only case that is applicable, since for these constraints the diameter is bounded above by four in a D_d digraph.

Case 2: For $s > 4d+1$ design the seed digraph with d columns. There will be at least four complete rows. Any extra nodes on the last incomplete row are stacked to the right and on top of the last completed row. Each row will be a D_x digraph and each column will be a D_y digraph as shown in Fig. 3. The value of y is $(R-1)$ if the number of rows $R < (1 + \lfloor(d/2)\rfloor)$, else y is $\lceil d/2 \rceil$, while $x = (d - y)$.

The number of rows R can be found based on when Case 2 is used. Since this is applied only when $s > 4d+1$ there are at least four rows. In the case when $a_i = 1$ and $a_{(i-1)} = 0$, the seed digraph uses the next brackets from Equation 1 also resulting in the number of nodes equal to d^2+d-1. This is the maximum number of nodes that the seed digraph will have to be drawn with. Since each row has d nodes, there will be at most d rows, with the last row having at most $d-1$ extra nodes wrapped around. The values of x and y are so chosen in Case 2, so as to ensure that along the column there are no more than $d+1$ nodes and along the row no more than d except for the last row.

For Case 1, the diameter is equal to four, while the *d-wide* diameter is five from Lemmas 1, 2, and 3. For Case 2 consider the d paths from a node in row and column (i, j) to a node in the row and column (k, l). A path along the row i, from (i, j) can reach (i, l), in two hops by Lemma 4. Similarly, along column l, from the node (i, l) to (k, l) it will take two more hops, thus making the diameter of the seed digraph four. Note despite the last row having more than d nodes, it is not necessary to traverse more than d nodes to reach the correct column, thus ensuring that the diameter is maintained at four.

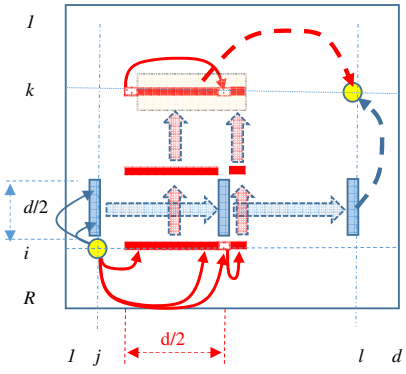


Fig. 4 Consider all paths from (i,j) to (k,l) to determine the *d-wide* diameter

To determine the *d-wide* diameter of the seed digraph from Case 2, consider Fig. 4 where all the node disjoint paths from node (i, j) to (k, l) have been shown. The $d/2$ edges along row i (red paths) are split up into two parts on either side of column $j + d/2$ by an additional hop from $(j + d/2)$ to $(j + d/2 + 1)$. This frees up column $(j + d/2)$ for the $d/2$ paths from the column j (blue paths) to hop in this column en route to column l and then by Lemmas 1-4 reach the sink node in at most five hops. The horizontal d/2 red edges of row i advance to row k and then by realigning into a block of d/2 consecutive nodes reach the sink node also in at most five hops in total based on Lemmas 1-3.

This ensures that the seed digraph designed by either Case 1 or Case 2 has it's diameter bounded by four and its *d-wide* diameter bounded by five. The seed digraph is a regular digraph of degree d, and node connectivity d as there are d node disjoint paths between any two nodes. In the next section the construction of the final digraph uses this seed digraph as the starting point.

The node disjoint paths from Fig. 4 can be listed as follows. This example shows the case when the degree d is six and there are three paths along the row and three along the column. The red paths that first go along the row i would be as follows.

P1: $(i,j) \rightarrow (i,j+1) \rightarrow (i-d,j+1) \rightarrow (k,j+1) \rightarrow (k,j+d+1) \rightarrow (k,l)$

P2: $(i,j) \rightarrow (i,j+2) \rightarrow (i-d,j+2) \rightarrow (k,j+2) \rightarrow (k,l)$

P3: $(i,j) \rightarrow (i,j+3) \rightarrow (i,j+4) \rightarrow (i-d,j+4) \rightarrow (k,l)$

The blue paths that go along the column j would be as follows.

P4: $(i,j) \rightarrow (i-1,j) \rightarrow (i-1,j+3) \rightarrow (i-1,l) \rightarrow (k,l)$

P5: $(i,j) \rightarrow (i-2,j) \rightarrow (i-2,j+3) \rightarrow (i-2,l) \rightarrow (k,l)$

P6: $(i,j) \rightarrow (i-3,j) \rightarrow (i-3,j+3) \rightarrow (i-3,l) \rightarrow (k,l)$

Note that P2, P4 and P5 might need one more hop along the way depending on the value of i, j, k and l. However, this will not alter the diameter of the seed digraph since there will be at least one path with length four. Note also that this will hold even if the nodes are in the last row with the extra nodes wrapped around. Note also that this will still hold fine if $i = k$ or $j = l$ using the results of Lemmas 1-3, or if $i-k$ or $j-l$ is less than $d/2$. This is done by appropriate juggling of the paths to ensure that along the row or column either two or three hops are needed, but never three for the same path on both the row and column.

Lemma 5: A regular digraph of at most $d^2 + d - 1$ nodes can be drawn with a diameter at most four, d-wide diameter at most five and node connectivity d.

The proof for this follows from the described construction method.

Q.E.D

IV. D-WIDE DIAMETER OF FINAL NETWORK

This section describes the construction method of the digraph *EL(G)* given a regular digraph $G = (V, E)$. Recall that an *EL(G)* is essentially a line digraph of G, namely *L(G)* with some additional edges and $a < d$ extra nodes.

A. *Construction of EL(G)*

The construction of the *EL(G)* is described below from a digraph *G*, such that the number of nodes of *EL(G)* are $d*n + a$, where *d* is the regular degree of *G*, *n* is the number of nodes in *G* and $a < d$

Step 1: First form the line digraph $L(G) = (V1, E1)$ of the seed digraph $G = (V, E)$.

Step 2: Next form a separate fully connected digraph with *a* nodes. These nodes are called the X-nodes. Note, these X-nodes have a degree (*a*-1), and need *d*-(*a*-1) additional edges to get to the regular degree *d*.

Step 3: Consider *a* unique nodes from *V*, say $v_1, v_2, \ldots v_a$. For each v_i $1 \le i \le a$, arbitrarily pick unique *d*-(*a*-1) predecessor nodes $P_1, P_2, \ldots P_{d-(a-1)}$, as well as *d*-(*a*-1) successor nodes $S_1, S_2, \ldots S_{d-(a-1)}$. Since the degree of each node is *d*, such unique nodes always exist on each of the *a* unique nodes chosen.

Step 4: Now for each chosen v_i of V, $1 \le i \le a$, from the corresponding line digraph L(G) remove the edge (P_j, v_i) to (v_i, S_j) $1 \le j \le d$-(*a*-1) and add the edges (P_j, v_i) to x_i and x_i to (v_i, S_j). This maintains the degree of the nodes of L(G), and increases the degree of the X-nodes from (*a*-1) to $d - (a-1) + (a-1) = d$.

Step 5: The digraph EL(G) is now formed by the union of the set of nodes of L(G) and the *a* *X*-nodes. The edges of the digraph EL(G) is the set of edges of L(G) with the modifications from Step 4. If $a = 0$, then EL(G) is the same as L(G).

The resulting digraph at the end of Step 5 takes the digraph *G*, constructs the line digraph *L(G)* and then adds $a < d$ nodes to *L(G)* to finally generate the extended line digraph *EL(G)*.

B. *Wide diameter of EL(G)*

By construction, *EL(G)* is a regular digraph of degree *d* if *G* was a regular digraph of degree *d* as well. Also if the node connectivity of *G* was *d* then *EL(G)* maintains the node connectivity. To generate the final digraph, the *EL(G)* transformation is applied to the seed digraph multiple times.

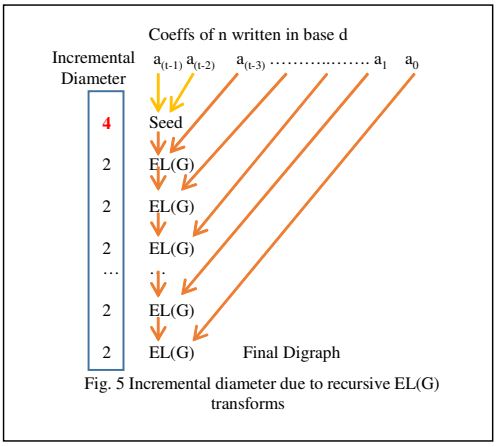


Fig. 5 Incremental diameter due to recursive EL(G) transforms

The *L(G)* step increases the diameter by one. If the addition of the *X*-nodes is on the path that determines the diameter, then the diameter increases by one more. As such every application of *EL(G)* will increase the diameter by at least one, but can increase by two as shown in Fig. 5.

This increase in the length of paths is applicable to not only the diameter but to all paths post *L(G)*. This will hence affect the d-wide and d-star diameters as well.

Since the number of times the *EL(G)* transformation is applied is at most $\log_d n - 2$, and the seed diameter is four, the final digraph's diameter is at most $4 + 2(\log_d n - 2) = 2\log_d n$. However, if all the additional a nodes that are added are not on the diameter determining path at each stage of *EL(G)*, then the best case diameter would be at the very least $4+ (\log_d n - 2) = \log_d n + 2$.

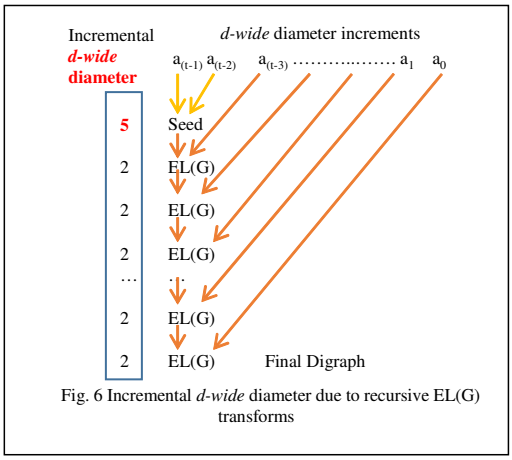


Fig. 6 Incremental *d-wide* diameter due to recursive EL(G) transforms

To find the d-wide diameter however, we have to find how much worse the other paths can get. In the extreme case that every X-node addition takes place on the path that determines the *d-wide* diameter at each EL(G) stage then the *d-wide* diameter will instead go up by $2(\log_d n - 2)$ from that of the seed digraph. So the *d-wide* diameter would then be $5 + 2(\log_d n - 2) = 2\log_d n + 1$. Fig. 6 shows the similarity of the increase in this d-wide diameter with each EL(G) much like that for the diameter. The only difference being that the starting value in the seed digraph is off by one. This difference is maintained through to the final digraph's construction.

Lemma 6: Given a seed regular digraph of diameter four, at most $d^2 + d - 1$ nodes, *d*-wide diameter at most five and node connectivity *d*, it is possible to generate a digraph larger than $d^2 + d - 1$ nodes, such that the diameter is bounded by $2\log_d n$, the *d-wide* diameter bounded by $2\log_d n + 1$, and connectivity equal to *d*.

The proof of this Lemma follows from the fact that the seed digraph is as defined by Lemma 5 and the recursive application

of the *EL(G)* as defined in the construction ensures all the conditions of connectivity, diameter and *d-wide* diameter.

Q.E.D

C. *Extremely tight bounds for diameter and d-wide diameters*

Taking the two extreme cases for the diameter and the *d-wide* diameter, we can see that the worst case diameter is $2\log_d n$. In contrast the worst case *d-wide* diameter is $2\log_d n + 1$. This implies that in the final digraph, even if *d*-1 nodes were to fail, the path will not be longer than $2\log_d n + 1$. Just as importantly, the degraded diameter of the new digraph with the faulty nodes removed will be increased only by one. This extremely tight bound in the delays on *d* node disjoint paths gives networks designed on this class of digraphs some very powerful properties. Note that individual source sink pairs may see a delay increase of more than one if their shortest paths were not the diameter determining paths. The point to keep in mind is that this applies to the diameter and *w-wide* diameters of the digraphs.

Very few other practical networks have such tight bounds on the diameter and *d-wide* diameters.

D. *Applications*

With tight bounds in the worst case delays across multiple node independent paths, with and without failures, it is much easier to resource plan the routes for many practical applications. Proper use of resources in computer networks linking various memory units, routers etc. ensures proper load balancing without much degradation in performance. This will help in the reliability, availability and serviceability (RAS) of a system by using the redundancy in it. Transportation networks and power distribution networks which often have to plan for problematic situations due to congestion, weather or repairs, the availability of alternate routes without much degradation will help the consumers. These methods are also very applicable to design of network-on-chips especially with a very large number of microprocessors.

V. CONCLUSION

In this work we have defined a new concept '*d-star-prime* container' for directed paths "to" a node, as against the regular definition of a star container which relates to paths "from" a node. To our knowledge this is also the first time the concepts of *d-wide* diameters was applied to the class of networks designed by extended line digraphs. The work showed that the diameter and the *d-wide* diameter of the networks differ by one. This extremely tight bound on the two diameters shows that the increase in the delay in message passing in networks based on this family of digraphs is very graceful even with *d*-1 node failures.

Future studies will focus on the degradation of the diameters when *d*-1 regions instead of individual nodes are deemed to have failed. Additional work needs to be done regarding the *w-star* diameters of this class of digraphs as this would give some information on delays regarding broadcast from a node to multiple nodes in parallel.

In conclusion, these regular directed networks can be designed using any number of nodes, will have a degree and a node fault tolerance of *d*, and the increase in diameter despite *d*-1 node failures is just one.

REFERENCES

[1] Douglas West, *Introduction to Graph Theory*, Prentice Hall 1996.

[2] M. A. Fiol, I. Alegre, and J. L. A. Yebra, "Line digraph iterations and the (d.k) problem for directed graphs," *Proceedings of the 10th International Symposium on Computer Architecture*, Stockholm, Sweden, 1983, pp. 174-177.

[3] M. A. Fiol, A. S. Llado, and J. L. Villar, "Digraphs on alphabets and the (d.N) digraph problem," *Ars Combinatoria*, vol. 25C, 1988, pp. 105-122.

[4] L. Gewali, H. Selvaraj, and D. Mazzella, "Constrained Disjoint Paths in Geometric Networks," *International Conference on Computational Intelligence and Multimedia Applications, vol. 2,* 2007, pp. xxiii-xxiii

[5] P. D. Joshi and S. Hamdioui, "Line Graph Based Fast Rerouting and Reconfiguration for Handling Transient and Permanent Faults," *IEEE 15th International Conference on High Performance Switching and Routing,* 2014, pp. 167-172.

[6] P. D. Joshi and S. Hamdioui, "Modified Regular Line Digraphs for Optimal Connectivity and Small Diameters," *Society for Industrial and Applied Mathematics (SIAM) Discrete Mathematics*, 2014, pp.45-46.

[7] P. D. Joshi, A. Sen, S. Hamdioui, and K. Bertels, "Region Disjoin Paths in a Class of Optimal Line Graph Networks," *International Symposium on Pervasive Systems, Algorithms and Networks*, 2014, pp. 1256-1260.

[8] D. F. Hsu, "On Container Width and Length in Graphs, Groups and Networks," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 1994, pp.668-680.

[9] S. Gao and D. Frank Hsu, "Short Containers in Cayley Graphs," Discrete Applied Mathematics 157, 2009, pp. 1354-1363.

[10] M. S. Krishnamoorthy and B. Krishnamurthy, "Fault diameter of interconnection networks," Computers and Mathematics with Applications, vol 13,1987, pp. 577-582.

[11] S. M. Reddy, J. G. Kuhl, and S. H. Hosseini, "On digraphs with minimum diameter and maximum connectivity," in the Proceedings of the 20th Annual Allerton Confernce, Oct 1982.

3

Robustness

Papers published in this category:

- IEEE HPSR 2014 : Line Graph Based Fast Rerouting and Reconfiguration for Handling Transient and Permanent Node Failures
- IEEE CSE (ISPAN) 2014 : Region Disjoint Paths in a Class of Optimal Line Graph Networks
- IEEE DFTS 2017 : Region Based Containers – A new paradigm for the analysis of Fault Tolerant Networks

While a lot of work goes into network design and specifications for high throughput and low latency, attention is also paid to network behavior when faults occur. Robustness deals with the ability of the network to realign its resources to allow fault-free communication between surviving nodes in an effective manner. This might even involve some surplus hardware designed into the system to allow such self-repair in the event of some failures. If the failures are widespread and permanent, the built-in redundancy may be insufficient and require external inputs to get around the issue.

The research work shows the resiliency of the proposed networks to the presence of point and region failures. Topological region failures trigger immediate neighbors up to a certain depth to also fail. The work shows that the proposed network can withstand point and region failures and shows how self-healing takes place to enable rerouting when such errors are detected.

3.1. Self-Healing

The work in this subsection describes the main results of the paper ***Line Graph Based Fast Rerouting and Reconfiguration for Handling Transient and Permanent Node Failures***. It describes how the construction of the network using a recursive procedure leads to a method of node naming. It becomes easy to identify the shortest paths by using a routing table with at most O(d^2) rows irrespective of the number of nodes where d is the degree of the network. Each entry in the table can keep track of not only the next node in the shortest but also alternate paths as well. This helps in load balancing and routing in the presence of faults. The 'stretch of a network' has different definitions in literature. Some researchers refer to the stretch as 'the difference between the best possible path through the network and the actual path the traffic takes through the network' [83]. Others define it as 'the maximum ratio between the length of the path actually traversed by a message and the length of the shortest path possible between its source and its destination' [57]. For the purpose of this work, the later definition is being used where the ratio of the path actually taken to the shortest path possible is referred to as the stretch of the network.

When faulty nodes are known, the paths avoid specific regular expressions in the node names, enabling automatic rerouting of packets. If the faulty nodes are intermittent then this information is sufficient. In case of permanent faults, there are two courses of action. The first is that the network can stay in the self-healing mode and reroute by avoiding the known regular expression corresponding to the faulty node name. Second, if spare nodes were incorporated into the system at design time, a physical rewiring of the network can be done to return the system to the optimally fault tolerant state as if no faulty nodes exist. The work describes how both these options are done and shows that the upper limit on the number of physical changes required for the second option is O(d).

3.2. Region disjoint routing in the network

The results in the paper ***Region Disjoint Paths in a Class of Optimal Line Graph Networks*** focus on the property of these networks to deal with not just point failures, but entire region failures. The metric of region disjoint connectivity for networks is a recent one and it emphasizes the fact that faults tend to cluster due to the nature of problems. As a result, considering connectivity without reference to locality does not show the real usability. On the other hand, if networks were calibrated to the number and size of region breakdowns they can withstand, they would better reflect reality. The results of the paper show that this family of networks is optimally robust, meaning that it not only withstands d-1 node disjoint failures, but in fact d-1 region failures, where each region has a size up to $2(\frac{d^{r+1}-1}{d-1})-1$ where $r = \lfloor \frac{\lceil log_d n \rceil}{2} - 1 \rfloor$. To put this in perspective, a network of 9999 nodes and degree 10 would normally have been deemed to accept up to nine node failures to be optimal. It is shown that this topology can withstand not up to nine node failures, but nine region failures where regions can have up to 21 nodes each, or a total of up to 189 node failures across nine regions. This is a very powerful

result and existing topologies do not have such robustness, including the hypercube topology.

3.3. Region Based Containers

The paper ***Region Based Containers – A new paradigm for the analysis of Fault Tolerant Networks*** proposes new concept called Region Based Containers. This effort merges the concept of containers with that of region disjoint routing as this is a more practical way of analyzing the degradation when regions failures occur in the network instead of point failures.

Similar to the concept of node disjoint paths in 'containers', the paths of a region based container are 'region disjoint'. Because many practically occurring failures are often clustered, studying the degradation of a network in the presence of region failures is important. It is also shown that the proposed methodology of network design results in networks that are not only optimally region fault tolerant, but that the degradation in the delays in spite of d-1 region faults is also bounded by one. This is a very powerful result showing the efficacy of the proposed family of networks. The recommendation is that this metric be adopted in network analysis of supercomputer networks.

Line Graph Based Fast Rerouting and Reconfiguration for Handling Transient and Permanent Node Failures

Prashant D. Joshi
Austin, Texas
prashant.joshi@gmail.com

Said Hamdioui
Delft University of Technology
Delft, The Netherlands
s.hamdioui@tudelft.nl

***Abstract*— Design of a class of optimally fault tolerant networks is described using modified line graphs. Appropriate node naming allows the shortest paths to be determined in log time. Self-healing in the presence of transient node failures can also be done in log time, and the rerouting takes place without loops or backtracking. The stretch of the network is maintained at the theoretically minimum value possible of one. The diameters of these networks are best in this class. In addition, the work describes how to reconfigure node connections to make the network optimally fault tolerant once more, in the presence of permanent faults. The changes so required are quantified and shown to be minimal. We demonstrate a class of optimally fault tolerant networks which provide minimal rerouting and reconfiguring overhead while maintaining optimal routing distances in the presence of faults.**

Keywords—Line Graphs, Node naming, Connectivity, Diameter of graph, Fault Tolerance

I. INTRODUCTION

Dependable networks in today's world are designed to have the ability to withstand some failures without adversely affecting the network performance. Some of the metrics include message delays with and without the presence of faulty nodes, flexibility in network design without restriction on the number of nodes or degree per node, self-healing during transient failures, and re-configurability during permanent node failures.

Past studies of graphs with nodes as computing elements and the edges as communicating links have been done [4-18]. The smallest degree of the nodes bounds the maximum tolerable node failures to keep the network connected. A network achieving this is optimally fault tolerant. Knowledge of the network helps get the shortest path between any two nodes. To enable self-healing in the presence of faults, the good nodes locally redirect traffic as required, and to counter permanent failures, measures of physical reconfiguration are required. The network topology presented here has the smallest known diameter for this class of diameters in published literature, enabling easy self-healing and potential reconfiguration with minimal changes.

This paper is arranged as follows. Definitions of standard and specific terms used in this paper will be followed by a short preview of prior work, the graph construction, the node naming and self-healing in the presence of faults. The method of re-configurability to regain optimality will then be discussed.

II. DEFINITIONS

The terms used in this study relating to graphs can be found in any standard text book on the subject like [1] and [2].

A graph G = (V,E), where V is a set of nodes, and E is the set of edges. The degree of a node is the number of edges incident on that node. In a directed graph (digraph) the indegree is the number of edges in to that node, and the outdegree is the number of outgoing edges from the node. A uniform digraph has equal in and out degrees on all nodes.

A path is the sequence of adjacent nodes and intermediate edges from the start node to the destination node. The distance from a node u to a node v is the number of edges along the shortest path from u to v. The diameter of a graph k(G) is the maximum of the shortest paths from node u to node v. The diameter bounds the number of edges required to traverse from any node to any other node. Two nodes whose distance is equal to the diameter are called diametric nodes.

The node connectivity of the graph is the minimum number of nodes, when removed, disconnects the graph. The fault tolerance is one less than the connectivity. An optimally connected graph can tolerate up to d-1 node faults, where d is the minimum degree of any node.

A D_d digraph is a uniform digraph (V, E) such that V is a set of nodes {0, 1, …(v-1)} and E = {(a,b) | a, b elements of V; b = (a + k) mod v, $1 \leq k \leq d$}. An edge of the type x-hop is an edge of the D_d graph from any node y to the node (y+x) mod v. The diameter of a D_d graph is bounded by $\lceil((|V|-1)/d)\rceil$.

A line graph of a digraph G = (V, E) is L(G) = (V1, E1) such that V1 = E and E1 = {(a,b) | a = (u1, u2) is an element of E and b = (u2, u3) is an element of E }. The predecessor set P(u) and the successor S(u) of a node 'u', an element of V, are defined as P(u) = {v | (v, u) is an element of E} and S(u) = {v|(u,v) is an element of E}.

978-1-4799-1633-7/14/$31.00 ©2014 IEEE

An Extended Line Graph EL(G) of a uniform digraph G=(V,E) of degree d and connectivity d, is L(G) with t additional nodes, t<d so that the number of nodes of EL(G) = n*d + t, such that the EL(G) also has degree d, connectivity d, and diameter k(EL(G)) ≤ k(G) + 2. A Modified Extended Line Graph MEL(G) can have t additional nodes with t≥d. The construction of the EL(G) and MEL(G) is described later.

The 'stretch' is the worst ratio of the shortest path determined to the actual shortest path in the graph, between two nodes.

III. PRIOR WORK

Computer network design for reliability and efficiency has been studied by many researchers. Study of various graphs and their properties have led to efficient network designs. Reducing the diameter with and without node faults, increasing connectivity, enabling fast routing algorithms, self-healing, and re-configurability are some of the metrics used. Researchers in [4]-[6] studied generalized hypercubes, restricting nodes to powers of 2, while those in [7]-[9] concentrated on De Bruijn graphs and modifications. In some cases the degree of graphs was compromised as in [6] when the number of nodes is a prime number. Most studies concentrated on keeping the fault tolerance optimal and the diameter proportional to $\log_d n$. The work in [11] also has a suboptimal diameter and restrictions on its degree, while [12] had worse diameter than the method proposed. Properties of line graphs have been studied in other papers like [14], [19] and [20] where the Bruijn-Kautz graphs were analyzed.

This paper extends the work in [23] where the details of the graph construction were covered resulting in the best in class diameters for uniform, directed, and optimally connected graphs, with no constraints on the number of nodes or degree. Networks in this class from previous studies such as [4]-[6], [10]-[12], [18], have a suboptimal connectivity, larger diameter or restriction on the number of nodes or degrees. The work in [12] did not determine factors like the shortest path, self-healing, stretch factors, or re-configurability, and the diameter was marginally worse. These issues are being covered in this study. A brief comparison of past work is shown in Table 1.

TABLE 1 REVIEW OF PRIOR WORK

Reference	Nodes	Degree	Fault tolerance	Diameter	
[4], [5]	2^m	m	m-1	m	dir
[6]	Πx_i	$\Sigma(x_i - 1)$	$\Sigma(x_i - 1)-1$	k	undir
[10]	$n>d^3$	d	d-2	$\lceil \log_d n \rceil$	dir
[11]	Any n	d≠2	d-1	$2\lceil \log_{(d-1)} n \rceil$	undir
[12]	Any n	Any d<n	d-1	$2\lceil \log_d n \rceil + 1$	dir
[18]	d^k	d, d+1	d-1,d	2k-1	undir
Proposed work	Any n	Any d<n	d-1	$2\lceil \log_d n \rceil$	dir

Studies in [27] have shown that the maximum stretch of random networks cannot be less than 3, while the average comes closer to 1. This work shows a stretch of 1 for this class of networks, which is the theoretically least possible value.

IV. NETWORK CONSTRUCTION, NODE NAMING AND SELF HEALING

A. Graphs G, L(G), EL(G) and MEL(G) construction

The line graph of a uniform and optimally connected digraph maintains the degree and connectivity, while the diameter increases by one and the number of nodes becomes n*d [17]. The construction of the extended line graph EL(G) will now be described. Given the uniform digraph G=(V,E) of degree and connectivity d, and diameter k(G), we need to add t nodes, 0≤t<d without modifying the degree or the connectivity, though the diameter could go up by 2. These t nodes are referred as X-nodes, and the other nodes are the non-X-nodes. First obtain the line graph L(G) and the completely connected graph of the t nodes. L(G) has connectivity and degree d, while the graph of the t nodes only has a degree t-1 and needs d-(t-1) more edges to and from each X-node.

For some t unique nodes from G: N_i $1 \le i \le t$, randomly pick unique d-(t-1) predecessor nodes $P_1, P_2, \ldots P_{d-(t-1)}$, and d-(t-1) successor nodes $S_1, S_2, \ldots S_{d-(t-1)}$. Since the degree of each node is d, such unique nodes always exist on each of the t unique nodes chosen.

Now for each chosen N_i of G, $1 \le i \le t$, removed it's edges (P_j, S_j) $1 \le j \le d-(t-1)$ identified above, from L(G), and instead add the edges (P_j, x_i) and (x_i, S_j). This maintains the degree of the nodes of L(G), and increases the degree of the X-nodes from (t-1) to d-(t-1)+(t-1)=d. If t=0, then EL(G) is the same as L(G).

The so constructed graph is the required EL(G). It is easy to see that the degree of EL(G) is d by construction, and that the number of nodes is equal to d*|V| + t. The distance between any two non-X-nodes of EL(G) with no X-nodes in its path will not increase from the corresponding L(G). If there is one X-node in the path, then the distance can go up by one. If there are two X-nodes in the path, the distance will not increase further since all the X-nodes are completely connected. Hence the diameter of the EL(G) can be 1 more than that of the L(G). It follows that the EL(G) has a diameter of k(G)+2.

A Modified Extended Line Graph can have d or more X-nodes, hence instead of a fully connected graph on these nodes, build a $D_{(d-1)}$ graph of these nodes. Then as before (taking 1 predecessor and successor for each X-node), add edges to bring the degree of this $D_{(d-1)}$ graph to d as well. If the number of these nodes is restricted to 2d+1, it is readily seen that the new diameter will be bounded by k(G)+3, since any two nodes are a distance of at most two now. Fig. 1 shows examples of a G, L(G), EL(G) and the MEL(G) graphs.

The connectivity being maintained is proved below. If there are no X-nodes in all the paths between two non-X-nodes, then this part of the proof is trivial, since L(G) is known to maintain the connectivity [17]. If there are X-nodes, the same X-node cannot be on two paths by construction. Hence two non-X-nodes always have d node independent paths between them. Two X-nodes have a direct edge, so are trivially connected with d removed nodes. In the case of a path from an X-node to a non-X-node, even if d-1 X-nodes are removed, there still exists a path to a non-X-node and then the first case

ensures a path. This logic in reverse applies to a path from a non-X-node to an X-node.

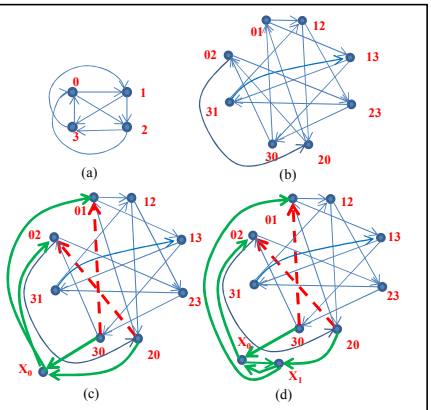


Fig. 1 (a) Graph G. (b) Graph L(G). (c) Graph EL(G) with 1 X-node. (d) Graph MEL(G) with 2 X-nodes.

B. *Final Graph and node naming*

For a uniform digraph of degree d and n nodes, we can write the number 'n' to base d, as below for each a_j<d.

$$n=((..((a_i d + a_{(i-1)})d + a_{(i-2)})d + a_{(i-3)})d\ldots)d+a_0 \qquad (1)$$

Equation (1) shows the EL(G) transformation being applied recursively on the graph of the previous bracket to generate the current bracket. Thus if we can get a good graph for the inner most bracket, then each subsequent bracket simply is the EL(G) of the previous graph, where the diameter of the next graph increases by at most 2.

The problem is now reduced to constructing a good 'base' graph for the innermost bracket of $a_i d + a_{(i-1)}$ number of nodes. The aim is to construct the base graph with a diameter bounded by 4. The details of the base graph design and proof that its diameter is bounded by 4, with a connectivity of d, are proved in [23]. With the innermost graph designed with diameter at most 4, each bracket on the outside is essentially a recursive EL(G) transformation as depicted in Fig. 2.

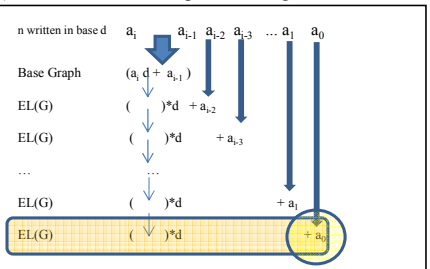


Fig. 2 Final Graph construction

The construction of the final graph starts with the base graph of diameter 4 and with each EL(G) transformation the diameter increases by 2, maintaining the degree and connectivity. The final diameter is 4 plus two times the number of times the EL(G) transformation is applied which is $\lceil \log_d n \rceil - 2$, and hence the final diameter is bounded above by $2*(\lceil \log_d n \rceil - 2)+4 = 2*\lceil \log_d n \rceil$. This result is the best known diameter for the class of optimally fault tolerant directed networks where there is no restriction on the number of nodes or the degree, to the best of the authors' knowledge.

Let us look at a good node naming. Any node formed after taking the line graph of a graph G, is named by concatenating the node names of the source and sink of the edge in G. For example if there is an edge between nodes named X and Y, where X and Y can be any sequence of characters, then the resulting node of this edge in the line graph would be named XY. X is the left predecessor and Y is the right predecessor of this node XY. This helps in identifying the descendant nodes of the base graph, among those of the final graph. For example, let us consider Fig. 3 where a graph of n=14 and d=2 is constructed, and try to trace a shortest path between the nodes 0110 to 2112.

0110→ ? →2112

This implies that in the penultimate EL(G) graph we have to find a path from 10 to 21, and then so on, from 0 to 2 as shown below.

0110 **(10→ ? →21)** 2112

0110(10**(0→ ? →2)**21)2112

Since 0→2 exists in base graph, we have been able to go down recursively until the base graph to find the required path. Now we will trace back up to retrace the nodes at each step.

0110→(10→02→21)2112

0110→1002→0221→2112

14 base 2 = 1110
Base graph (3 nodes)
0→1,2 1→2,0 2→0,1
L(G) (6 nodes)

01→12,10	02→20,21	12→20,21
10→01,02	20→01,02	21→12,10

EL(G) (7 nodes)

01→x	x→12
21→x	x→10
01→~~12~~,10	02→20,21
12→20,21	10→01,02
20→01,02	21→12,~~10~~

L(G) (14 nodes)

01x → x12, x10	x12 →1220, 1221
21x →x12, x10	x10 →1001, 1002
0110→ 1001, 1002	0220 → 2001, 2002
0221 → 2112, 21x	1220 → 2001, 2002
1221 → 2112, 21x	1001 → 01x, 0110
1002 → 0220, 0221	2001 → 01x, 0110
2002 → 0220, 0221	2112 → 1220, 1221

Diameter is guaranteed to be bounded above by $2*\lceil \log_2 14 \rceil$ and in this case the diameter is 4.

Fig. 3 Example of a graph of 14 nodes with degree 2. The arrow indicates that the node to the left of the arrow has edges to node(s) after the arrow

If 0221 is known to be faulty, then avoiding either the edge 02 or 21 will never result in the node 0221 in use in any path. Hence we could then take 0→1→2 at the base graph level and thus get the path: 0110(10(0→1→2)21)2112

0110→1001→0112→1221→2112 thus avoids the faulty node. It is possible that we might not have to go down to the base graph since an intermediate level might have a direct edge and in which case the path is between non-diametric nodes.

C. Field of Influence, Routing Table for Self-Healing and log time

This leads us to the concept of 'field of influence' of a node of the base graph upon the nodes in the final graph. If a node in the final graph has the regular expression of a node of any previous transformation, then that node has a field of influence on the final node. As such, in the determination of the shortest path, if all final nodes know which nodes are faulty, then they know which edges (regular expressions) to avoid in the shortest path determination.

Here, we are not going into the details of faulty node determination (in this class of EL(G) based networks, periodic broadcasts enable all nodes to know which are faulty nodes, assuming them to be non malicious, in O(log n) steps. This proof is beyond the scope of this paper [23]).

base graph	**EL(G)-1**	**EL(G)-2**	**EL(G)-3**
a	ab	abbc	abbcbccd
b	bc	bccd	bccdcdde
c	cd	cdde	cddedeef
d	de	deef	deefeffg
e	ef	effg	*effgfggh*
f	fg	fggh	fgghghhi
g	gh	ghhi	ghhihiij
h	hi	hiij	hiijjjk
i	ij	ijjk	
j	jk		
k			

Fig. 4. Field of Influence and shortest path determination

Consider a path between two nodes of a graph, abbcbccd to hiijijjk in Fig. 4. A node 'y' below another 'x' along a column means the edge x→y exits at that EL(G) level, and each column shows the result of the EL(G) of the previous column. Only the shortest path between the rightmost predecessor of the source node 'd' and the leftmost predecessor of the sink node 'h' is required to be known in routing tables. From the routing table, once the shortest path (shaded in yellow) from the first column of the base graph is known to be d→e→f→g→h, then the full path is automatically known due to the node naming.

If it is known that effgfggh is faulty, then to avoid this node, the path in the base graph from d →…→h can avoid at least one of the edges: ef, fg, gh and by doing so this regular expression will not be present in any of the nodes of the path in the final graph thus ensuring automatic self-healing.

Notice that although the final graph might contain any number of nodes, the routing table needs to have only ($a_i d + a_{(i-1)}$) rows and columns (at most d^2-1). Each row indicates the starting node, and the column number represents the final node of the base graph. Every entry contains the next node to take, to get the shortest path. In case of known faulty nodes, the routing table will allow avoiding specific edges of the base graph.

Notice that this self-healing process ensures there are no loops or backtracking in the shortest path determination and the shortest path is always obtained. This means that the stretch of these graphs is always 1. This equals the theoretically minimum value possible. The ratio of the maximum amount of storage required to the number of nodes grows smaller as the number of nodes grows, since the maximum amount of storage is a fixed quantity irrespective of the number of nodes.

If the faulty nodes are having transient faults and the status at a given time is known to all the nodes, the routing can be done by avoiding these nodes. On the other hand, if this transient behavior is seen to be more permanent, then a decision can be made to make it as a permanent fault thus enabling actions to reconfigure the network if required.

In [23] it has been shown that the base graph of diameter 4 will see its diameter change by at most 1 in the presence of 1 node fault. This means that the increase in diameter of the final graph is also bounded by 1. Thus the determination of the shortest path is based off of finding the path between two nodes in a routing table of size at most d^2-1. However since the diameter is at most 4, the number of steps are limited to 4 + ($\log_d n - 2$). Thus the number of steps in the presence of one fault is limited to 5 + ($\log_d n - 2$). With more faulty nodes, the number of steps to determine the shortest path goes up in small amounts and the overall number of steps is $O(\log_d n)$ which is due to the number of the EL(G) transformations for the final graph.

At this stage we have shown that this method using EL(G), and proper node naming enables us to find the shortest path with and without node faults in $O(\log_d n)$ steps. Refer to the example routing table in Fig. 5. Each entry of the routing table at location (a,b) gives the prioritized list of the next nodes to take.

If the entry in (a,b) of the routing table contains (x,y,z,…) it means that to go from a to b the next best node is x. If the edge ax is to be avoided then the next best node is y, and so on. There are d nodes in each entry since we allow up to d-1 failures in the network. It is illustrative to note that the stretch of these networks is fixed at 1, irrespective of the number of nodes, degree, or the number of failures up to d-1 (since it is possible to come up with a fixed routing table on a much smaller number of nodes a priori). In contrast with the stretch values ranging from 1 to more than 3 in various other networks [27]-[29], the stretch value of these networks is always 1.

The example routing table in Fig. 5 shows the next node to take when going from a node (row#) to the final destination (col#). The col# never changes, as it is the final node, but the rows change as we go from one node to another based on the routing table. The nodes entered are in sorted order in case a node has to be avoided for self-healing purposes.

To→ From↓	0	1…..	(n-1)
0	1,2..	2,3,…	2,1,…
1 … …	0,2…	2,0,…	2,0,…
n-1	2,4,…	1,0,…	0,1,…

Fig 5. Sample routing table. Top row is the final destination, left column is the current node and the entries are prioritized next nodes.

V. RECONFIGURATION DUE TO PERMANENT FAULTS

In the presence of permanent node failures, it might be desired that the network be reconfigured with the available good nodes. Consider the last stage of Fig. 2 where a_0 nodes are being added. If the faulty node happens to be one of those a_0 nodes, then it is as good as adding a_0-1 nodes in that step. From the EL(G) construction, the a_0 nodes are the X-nodes of the last step. Hence to remove one X-node, we will remove t-1 edges to/from the removed X-node, insert these remaining t-1 nodes into t-1 edges of the non-X-nodes per the EL(G) construction, and reconnect (d-(t-1)) non-X-edges which had this X-node inserted in the EL(G) step. Hence we need to delete/alter at most 3(t-1) + d –(t-1) = d+2t-2 edges, and since t can be at most d-1, we will need to modify at most 3d-4 edges.

If the faulty node happens to be a non-X-node, then we will remove one of the X-nodes by above procedure and insert it in place of the faulty non-X-node. This will mean that in addition to the 3d-4 changes, we will have to reconnect 2d edges to this inserted node, meaning the maximum edge changes we would be required to do is 5d-4.

Thus with at most 5d-4 edge changes we have a network with the n-1 nodes identical to what we would have started with in the first place if we had to design a network with n-1 nodes instead.

These two options make an assumption that a_0 is non-zero. This would not work if the number of node failures were greater than a_0. A decision has to be made at the time of network design, whether or not to take a hit on the diameter to enable the flexibility to repair a certain number of future node failures. This is achievable in the last EL(G) step to allow the addition of t≥d nodes. As defined in Section IV, we will then have a MEL(G) as the last step of our final network design step in Fig 2.

The number of changes required in this situation is slightly different from the case where we did not have the MEL(G) stage. We will need 2(d-2) edges to be reconnected for the D_{d-1} part, 2 edges from non-X side for a total of 2d edges for the reconfiguration involving a faulty X-node, and 4d for a faulty non-X-node.

Now we will express m = n-d nodes to base d, and apply recursive EL(G) steps, but add d in the last MEL(G) step as in Eq.2 and Fig. 6. This ensures that we have at least d, and at most 2d-1 nodes available for reconfiguration at a later stage. An example of the number of edge modifications has been shown in Fig. 7 for various values and types of node failures. Note that these numbers are a function of the type of node failure (X-node, non-X-node) and the degree, not the number of nodes in the graph. The percentage of the edges required to be changed however is a function of the number of nodes.

$$n=((..(a_i d + a_{(i-1)})d + a_{(i-2)})d + a_{(i-3)})d\ldots)d+(a_0+d) \qquad (2)$$

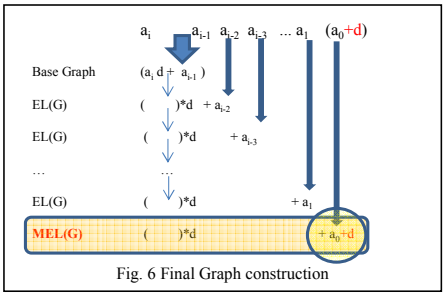
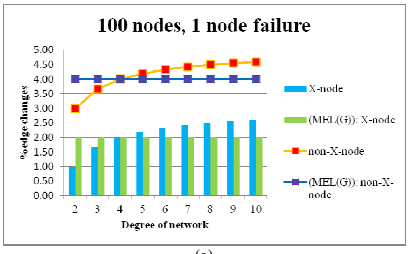
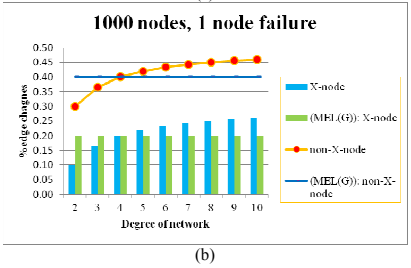


Fig. 6 Final Graph construction



(a)



(b)

Fig. 7. Percent of network edge changes to reconfigure for X-nodes or non-X-nodes fail, with and without the MEL(G)

Notice that re-configurability results in the network being restored as if it had been designed with the new value of the number of nodes. Notice that these changes are $O(\log_d n)$, there is no change in the degree of the nodes or the diameter and the changed network has optimal connectivity. This is in contrast

with some other methods where the degree or the diameter can change [30]. Slight routing table changes will have to be made in this case to cater for the fact that some nodes have been removed.

VI. CONCLUSION

In this paper we described a class of uniform directed networks constructed based on Extended Line Graphs, which resulted in the best in class diameters and optimal connectivity of these networks. Further, we showed a novel node naming method such that the upper bound on the routing table size is independent of the size of the network, which enables $O(\log_d n)$ steps to determine the shortest path in the network, whether self-healing is being performed or not. The stretch of the network is the theoretically best possible value of 1 irrespective of the number of node failures up to (d-1) and the size of the routing tables. The shortest path determination problem is reduced to finding the shortest path in at most d^2-1 nodes independent of the number of nodes. We further analyze the percentage of edge changes required to reconfigure the network to disconnect a permanently faulty node. This number is a fixed function of the degree of the network, but is dependent on which type of node is affected. Future work will look into extending this research to help isolate malicious nodes as well.

REFERENCES

[1] C.Berge, Graphs and Hypergraphs. Amsterdam, The Netherlands: North-Holland, 1973.

[2] A.J. Hoffman and R.R. Singleton, On Moore graphs with diameter 2 and 3, IBM J. Res. Develop. 4 (1960) 497–504.

[3] W. T. Tutte, A family of cubical graphs, Proceedings of the Cambridge Philosophical Society, 43 (1947) 459–474.

[4] J. R. Armstrong and F. G. Gray, "Fault diagnosis in Boolean n-cube array of microprocessors," IEEE Trans. Comput., vol. C-30, pp. 590-596, Aug. 1981.

[5] J. Kuhl and S. M. Reddy, "Distributed fault tolerance for large multiprocessor systems," in Proc. 7th Annual Symposium Computer Architecture, May 1980

[6] L. Bhuyan and D.P. Agrawal, "Generalized hypercube and hyperbus structure for a computer network," IEEE Trans Computers, vol. C-33, Apr. 1984

[7] M. L. Schlumberger, "DeBruijn communication networks," Ph.D. dissertation, Stanford Univ., Stanford, 1974.

[8] D.K. Pradhan and S.M. Reddy, "A fault tolerant communication architecture for distributed systems" IEEE Transactions Computers vol. C-31, Sept. 1982.

[9] D. K. Pradhan, Z. Hanquan, and M. L. Schlumberger, "Fault tolerant multibus architecture for multiprocessors," in Proc. 14th Int. Conference on Fault-Tolerant Computers, 1984, pp. 400-408.

[10] M. Imase, T. Soneoka, and K. Okada, "Connectivity of regular directed graphs with small diameters" IEEE Transactions on Computers, vol. C-34, March 1985.

[11] U. Schumacher, "An algorithm for k-connected graph with minimum number of edges and quasiminimal diameter," Networks, vol. 14, 1984.

[12] A.Sengupta, P. D. Joshi and S. Bandyopadhyay,"A Synthesis Approach to Design Optimally Fault Tolerant Network Architecture", IEEE Transactions on Computers, vol.40, January 1991.

[13] Daniela Ferrero and Carles Padro, "Connectivity and fault-tolerance of hyperdigraphs", Discrete Applied Mathematics, 2002.

[14] M. A. Fiol, I. Alegre, and J. L. A. Yebra, "Line digraph iterations and the (d . k) problem for directed graphs," in Proceedings of the 10th International Sympmosium on Computer Architecture, Stockholm, Sweden, 1983.

[15] M. A. Fiol, A. S. Llado, and J. L. Villar, "Digraphs on alphabets and the (d.N) digraph problem," Ars Combinatoria, vol. 25C, pp. 105-122, 1988.

[16] M. A. Fiol, J. L. A. Yebra, and I. Alegre, "Line digraph iterations and the (d . k) digraph problem," IEEE Transactions on Computers, vol C-33, pp. 400-403, May 1984.

[17] S.M. Reddy, J.G. Kuhl, and S.H. Hosseini, "On digraphs with minimum diameter and maximum connectivity," in the Proceedings of the 20th Annual Allerton Confernce, Oct 1982.

[18] D.K. Pradhan, "Fault tolerant multiprocessor link and bus network architecture," IEEE Transactions on Computers, vol. C-34, Jan. 1985

[19] Liu, S. Trajanovski, P. Van Mieghem, "Reverse Line Graph Construction: The Matrix Relabeling Algorithm MARINLINGA Versus Roussopoulos's Algorithm", Delft University of Technology submission to xrXiv.org. October 2010.

[20] J. Naor and M. B. Novick, "An efficient reconstruction of a graph from its line graph in parallel." J. of Algorithms, 11(1): 132-143, 1990.

[21] J. Suurballe and R. Tarjan, "A quick method for finding shortest pairs of disjoint paths," Networks, vol. 14, pp. 325–336, 1984

[22] Dahai Xu, Yang Chen, Yizhi Xiong, Chunming Qiao, Xin He, "On the Complexity of and algorithms for finding the shortest path with disjoint counterpart", IEEE/ACM Transactions on Networking, vol. 14, No. 1, February 2006.

[23] Prashant D. Joshi, Said Hamdioui, "Modified uniform line digraphs with optimal connectivity and small diameteres", Forty-Fifth Southeastern International Conference on Combinatorics, Graph Theory and Computing, 2014.

[24] Amitabh Trehan, "Algorithms for Self-Healing Networks", Ph.D. dissertation, University of New Mexico., New Mexico, 2010.

[25] Alper Mizrak,Yu-Chung Cheng, Keith Marzullo, Stefan Savage, "Fatih: detecting and isolating malicious routers", in the Proc. of The International Conference on Dependable Systems and Networks, 2005.

[26] Robert Poor, Charlotte Auburn and Cliff Bowman,"Self Healing Networks", ACM Queue, May 2003.

[27] Mihaela Enachescu, Mei Wang, Ashish Goel, "Reducing Maximum Stretch in Compact Routing", in the Proc. of IEEE INFOCOM, 2008.

[28] Dmitri Krioukov, Kevin Fall, Xiaowei Yang, "Compact Routing on Internet-Like Graphs", in the Proc. of IEEE INFOCOM, 2004

[29] Aifei Zhong, Srehari Nelakuditi, Yinzhe Yu, Sanghwan Lee, Junling Wang, Chen-Nee Chuah, "Faulure inferencing based fast rerouting for handling transient link and node failures", in the Proc. of IEEE INFOCOM, 2005.

[30] Saia, J., Trehan, A., "Picking up the pieces: self-healing in reconfigurable networks" IEEE International Symposium on Paralleland Distributed Processing, 2008.

2014 IEEE 17th International Conference on Computational Science and Engineering

Region Disjoint Paths in a Class of Optimal Line Graph Networks

Prashant D. Joshi
Cadence Design Systems
Austin, Texas
prashant.joshi@gmail.com

Arunabha Sen
Arizona State Univerisity
Tempe, Arizona
asen@asu.edu

Said Hamdioui; Koen Bertels
TU Delft
Delft, The Netherlands
shamdioui@tudelft.nl
k.l.m.bertels@tudelft.nl

***Abstract*—Communication networks are one of the backbones to society, and it is important that they withstand failures. Improving the robustness of a network involves good algorithms for network connectivity and routing in the presence of faults. The importance of being able to connect the good parts of the network when catastrophic failures, natural or manmade, affect the system cannot be underestimated in either a military or a natural disaster situation. Traditional network robustness has been studied where point failures occur, with no reference to their locality. In reality, if regional failures are taken into consideration as the metric to evaluate the robustness of a network, then we can apply them to situations where simultaneous failures take place, but clustered in regions.**

Region Based Connectivity (RBC) was introduced in INFOCOM 2006, subsequent to which there have been some researchers who have looked at various aspects of this metric. In this paper, we look at the RBC of a class of uniform networks produced by recursive modified line graphs. This work deals with topological regions, and not geometric regions. The study shows that these networks display optimal RBC and calculates the upper bounds on the radius of these regions for optimal RBC.

Keywords—Network connectivity; Line Graphs; Region Based Connectivity; Topological Regions

I. INTRODUCTION

The network survivability traditionally has been studied as the connectivity of the underlying graph despite node or edge disconnections. The network could represent any application like transportation, communication, waterways, power and computer networks. The metric of the maximum delays along the network with and without some node failures has widely been studied by the diameter and connectivity of the graphs represented by these networks. A k-connected network can withstand k-1 nodes becoming bad. The connectivity of a network has no inherent location information. Real life situations however result in localization of problems due to natural or manmade catastrophes like earthquakes, tsunamis or war zones. In such situations, the goodness of a network needs to look at the robustness when possibly a much larger number of localized nodes are eliminated.

This leads to the concept of Region Based Connectivity (RBC) which was first introduced in INFOCOM 2006 [1] The work was expanded [2] by looking at the concepts of Multiple Region Fault Models. It was proved that the problem of finding the maximum number of region-disjoint paths between a pair of nodes, as well as finding the minimum number of regions whose removal disconnect a pair of nodes, are both NP-complete problems. Some researchers [5] considered the problem in a planar graph to find a circular region of a given radius which would cause the biggest network degradation destroyed, if all nodes in that region were destroyed, and proposed a polynomial time algorithm to find such critical regions. The work in [8] introduced a geographical variant of the max-flow min-cut problem, and the researchers in [6] gave a polynomial time algorithm for some of the conjectures in [8] on the relationship between the geographic min-cut and the max-flow.

On a different front, line graphs have been used to model families of networks which display some very desirable properties of shortest delay in message passing, graceful degradation of the delay with faulty nodes, optimal connectivity of the networks, self-healing and routing without loops etc. [3], [4], [10]-[12].

To the authors' knowledge there has been no study of the region based connectivity of networks based on modified line graphs.

This study describes the RBC of a class of uniform line digraphs with no restriction on the number of nodes. The final network is designed by recursive application of extended line graphs on a seed graph. The properties of connectivity of the seed graph are preserved, while the diameter increases linearly with the recursive application of the extended line graphs step. This results in the best known diameters in literature for this class of networks. The nodes of the seed graph have a field of influence on subsets of final graphs, and this leads to the concept of the topological regions of the final network controlled by the original seed graph's nodes. If the seed graph is k-connected, the final network will also be k-connected [4]. However, in this paper we prove that it will also be k-region

978-1-4799-7981-3/14 $31.00 © 2014 IEEE
DOI 10.1109/CSE.2014.242

connected. The region connectivity cannot exceed k. Hence we prove that this class of graphs is both optimally k-connected and k-region connected.

This study indicates how the region based connectivity comes out of the line graphs and also computes the maximum sizes of the regions to ensure k-region connectivity. The term graph and network is used interchangeably in this paper.

The paper is organized as follows. A short description of some terms used is followed by the construction of the graph based on iterative modification of the line graphs. The concept of the field of influence and the topological regions is used to show how the graph is k-region connected. Region size calculations are used to give an upper bound on the radius of each region. Some ideas for extension of this work are then proposed.

Formally, this paper aims to prove that the studied class of networks designed by extended line graphs have a diameter of $2\lceil \log_d n \rceil$ and are optimally region connected for an upper bound on the region's radius.

The geometric and topological regions address different types of problems, and each has merits. Study of Ebola transmission due to airline connections, or power outages due to surges, are examples where topological regions are appropriate and not geometric.

II. Terminology

Most of the graph theory related terms used in this paper can be found in any standard text book on the topic [9]. A few of the terms used in this study are also defined here. A digraph G = (V, E) has n = |V| nodes, and (p,q) is an element of E, if there is a directed edge from node p to node q. The indegree (correspondingly outdegree) of a node is the number of edges incident into (correspondingly out of) that node. In a uniform digraph all indegrees and outdegrees of all nodes are equal to the degree d. A path from a node p to node q is a sequence of adjacent edges that start from p and end in q. Two node disjoint paths from p to q have no common node except for p and q. A graph is strongly connected if any node can be reached from any other node in it. The connectivity of a graph is k if the removal of any k-1 nodes still keeps the remaining graph strongly connected, and the removal of some specific k nodes results in the graph becoming non strongly connected. The connectivity is obviously bounded by the minimum degree of any node in the graph. If a graph achieves this connectivity, it is called an optimally connected graph. The distance between two nodes is the number of edges in the shortest path between them. The diameter of the graph is the largest value of the distance between any two nodes of the graph.

A topological region of radius 'r' centered at a node 'p' contains all the nodes at a distance 'r' from 'p' and all intermediate edges, as if the graph is undirected. Removal of this region will render some edges without a source or a sink and will also be removed.

Fig. 1 shows a topological region of r = 2 centered at a node p. The red colored nodes and edges, and the dotted brown edges are removed. The blue nodes and edges will remain. The number of nodes in a region of radius r is equal to $1 + 2(d^1 + d^2 + \ldots d^r)$. Hence the number of nodes in a region of radius r is given by (1).

$$2[(d^{(r+1)} - 1)/(d-1)] - 1 \quad (1)$$

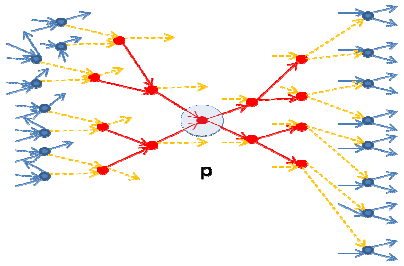


Fig. 1. Region of radius 2, centered at node p.

Analogous to the concept of the vertex cut, the region cut is the set of regions when removed results in the remaining graph becoming disconnected. The region based connectivity of a graph or RBC, implies up to any RBC-1 regions can be removed without making the remaining graph disconnected. Since any graph of minimum degree d cannot have an RBC > d, a graph that achieves the RBC of the minimum degree will be called an optimally region based connected graph. For a uniform graph this would be d.

A line graph of a digraph G = (V, E) is L(G) = (V1, E1) such that V1 = E and E1 = {(a,b) | a = (u1, u2) is an element of E and b = (u2, u3) is an element of E }.

An Extended Line Graph EL(G) [4], of a uniform digraph G=(V,E) of degree d and connectivity d, is L(G) with t additional nodes, t<d so that the number of nodes of EL(G) = n*d + t, such that the EL(G) also has degree d, connectivity d, diameter $k(EL(G)) \leq k(G) + 2$.

III. Graph Construction

A. Graph Construction

The iterative application of the modified line graph transformation on the seed graph results in a final graph which has some very interesting properties [4]. If the seed graph was a uniform k-connected digraph of diameter D and degree d, the resulting final graph also is a uniform k-connected digraph of degree d. If this transformation is applied t times, then the final graph's diameter is at most (D+2t).

According to [4] it is always possible to generate a uniform d-connected digraph of degree d, diameter 4 and the number of nodes at most d^2-1. To generate the final uniform d-connected digraph of n nodes let us say we had a seed graph which was already a d-connected uniform digraph of at most d^2 -1 nodes and diameter 4. Expressing the number 'n' in base 'd' as shown in (2), it is apparent that each bracket is the EL(G) of the previous inner bracket.

$$n = (\ldots(((a_i d + a_{(i-1)}) d + a_{(i-2)}) d + a_{(i-3)})\ldots)d + a_0 \quad (2)$$

Each transformation multiplies the previous number of nodes by d and adds a number a_i <d. The seed graph has a diameter 4, and the EL(G) transformation is applied ($\log_d n$ - 2) times increasing the diameter by at most 2 each time. Hence the final graph's diameter is bounded above by 4 + 2($\log_d n$ -2) = 2$\log_d n$. This is the best known diameter in literature for the class of networks built using extended line graphs. Fig. 2 gives an idea of what is being done.

Since the value of the innermost bracket is at most d^2-1, the seed graph has to be drawn with at most d^2-1 nodes. There are multiple corner cases, but each result in an optimally d-connected uniform digraph of diameter as has been described in a previous study [4]. The extended line graph first takes the line graph and then adds a_i<d nodes maintaining the connectivity and degree, but adds 1 more to the diameter in each step. Thus the EL(G) adds 2 to the original diameter. The previous work went into the details of this, but an example is given here in the next subsection to show the construction.

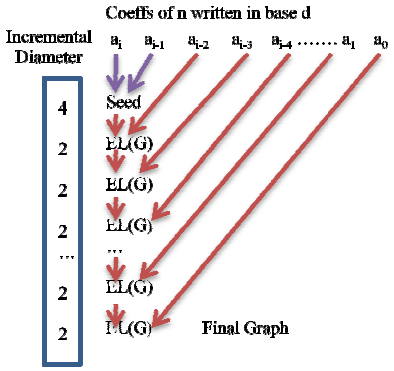


Fig. 2. Diameter of network using recursive EL(G) transformation.

B. *Node Naming*

Any node formed after taking the line graph of a graph G, is named by concatenating the node names of the source and sink of the edge in G. For example if there is an edge between the nodes named X and Y, where X and Y can be any sequence of characters, then the resulting node of this edge in the line graph would be named XY. X is the left predecessor and Y is the right predecessor of this node XY. This helps in identifying which nodes of the final graph are the descendants of which set of nodes in the seed graph. Fig. 3 shows an example of a line graph and its naming. Note that the number of nodes in the line graph are equal to the number of edges of G. Since the example does not show a uniform graph, the number of nodes is L(G) is not d times the number of nodes of G.

As an example of the work in [4] to construct a graph using a proper seed graph is shown below with n=14 and d=2.

14 base 2 = 1110

Hence the base graph is formed with 11 base 2 nodes, then an EL(G) Stage 1 multiples the number of nodes by 2 and adds 1 more node, and lastly the Stage 2 multiplies the number of nodes by 2 and adds 0 nodes. (a → b,c means the node a has a directed edge to b and c)

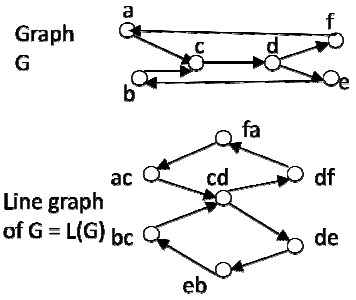


Fig. 3. A line graph and the node naming.

Stage 0:
Base graph (3 nodes)
0→1,2 1→2,0 2→0,1

Stage 1: EL(G)-1
L(G) (3*2 = 6 nodes)
01→12,10 02→20,21 12→20,21
10→01,02 20→01,02 21→12,10

EL(G) (7 nodes) (add 1 node to the L(G))
01→x x→12
21→x x→10
01→~~12~~,10 02→20,21
12→20,21 10→01,02
20→01,02 21→12,~~10~~

Stage 2: EL(G)-2
L(G) (7*2 = 14 nodes)
01x → x12, x10 x12 →1220, 1221
21x →x12, x10 x10 →1001, 1002
0110→ 1001, 1002 0220 → 2001, 2002
0221 → 2112, 21x 1220 → 2001, 2002
1221 → 2112, 21x 1001 → 01x, 0110
1002 → 0220, 0221 2001 → 01x, 0110
2002 → 0220, 0221 2112 → 1220, 1221

Once the final graph is constructed, let us see how to find the shortest path between two nodes, say 0110 and 2112.

0110→ ? →2112

This implies that recursively in the EL(G) inverse of the graph (end of Stage 1) we have to find a path from 10 to 21.

0110 **(10→ ? →21)** 2112

And in turn further up the chain, we have to find a path from 0 to 2 in the Seed graph.

0110(10**(0→ ? →2)**21)2112

Since 0→2 exists in the seed graph, we have been able to recursively go up till the seed graph to find the required path. Now we will trace back recursively up to retrace the nodes at each step.

0110→(10→02→21)2112

0110→1002→0221→2112

If 0221 is known to be faulty, then avoiding either the edge 02 or 21 will never result in the node 0221 in use in any path. Hence we could then take 0→1→2 at the seed graph level and thus get the path:

0110(10(0→1→2)21)2112.

Thus, 0110→1001→0112→1221→2112 avoids the faulty node.

C. Field of Influence

This leads us to the concept of '*field of influence*' of a seed node on the nodes of the final graph. If a final graph's node name contains the regular expression of the node from the seed graph then that seed graph node has an influence on the final graph's node.

TABLE I
FIELD OF INFLUENCE AND SHORTEST PATHS

seed graph	EL(G)-1	EL(G)-2	EL(G)-3
a	ab	abbc	abbcbccd
b	bc	bccd	bccdcdde
c	cd	cdde	cddedeef
d	de	deef	deefeffg
e	ef	effg	*effgfggh*
f	fg	fggh	fgghghhi
g	gh	ghhi	ghhihiij
h	hi	hiij	hiijijjk
i	ij	ijjk	
j	jk		
k			

Consider a path between two nodes of Table 1. It shows the relevant part of the subgraph from abbcbccd to hiijijjk and how it was created from the seed graph by successive EL(G) transformations. A node 'y' below 'x' along a column means the edge x→y exits at that EL(G) level, and each column shows the result of the EL(G) of the previous column. Only the path between the rightmost predecessor of the source node 'd' and the leftmost predecessor of the sink node 'h' is required to be known. Suppose the shortest path (shaded in yellow) from the seed graph is known to be d→e→f→g→h, then the full path is automatically known due to the node naming, as shown in the final column. Note that the node 'e' of the seed graph, has an influence on the nodes bccdcdde, cddedeef, deefeffg and effgfggh. Now if it is known that the node *effgfggh* is faulty, then if the path from d→…→h in the seed graph avoids the node e, then the faulty node *effgfggh* will never be used in the path in the final graph. Also, since the seed graph is d-connected, there always exists a path from d→…→h that does not go through the node e. Since the seed graph is drawn to be d-connected, the final graph is also d-connected.

IV. Region Connectivity and Radius

Given a region we know which seed graph node(s) have a field of influence on the region. Hence if we have to avoid not just a node, but a region, we know which nodes of the seed graph to avoid in the seed graph path, thereby ensuring that we avoid the region in the path of the final network.

It is clear that if a node p occurs in a path between the nodes x and y of the seed graph, then p is not on any other of the d-1 node disjoint paths between x and y of the seed graph. We can prove that the corresponding node disjoint paths between a node whose right predecessor is x and a node whose left predecessor is y, will also not have the regular expression 'p' in more than 1 node disjoint paths.

This can be proved by contradiction by assuming that the regular expression p was present in the d node disjoint paths in at least 2 paths in the graphs whose EL(G) generated the final graph as shown in Fig. 4. Recursively going backwards on these two paths until we reach the seed graph it would mean that p occurs on two node disjoint paths between x and y which is a contradiction.

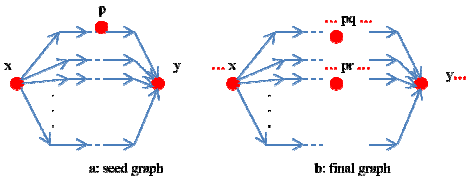


Fig. 4. Proof by contradiction that the regular expression p cannot occur in more than 1 path of the final graph.

As such, if we could identify a region which had a field of influence from p, then we can say that removal of that region would correspond to the removal of the node p from the seed graph. If the removal of p still leaves paths between x and y, then there will be paths which between two nodes of right and left predecessors as x and y respectively in the final graph.

Consider a path involving a node 'x' in the seed graph and the subsequent EL(G)s:

a→b→c→x→e→f→g→h
(seed): 'x' occurs in 1 node

ab→bc→cx→xe→ef→fg→gh
(EL(G)-1: 'x' occurs in 2 nodes

abbc→bccx→cxxe→xeef→effg→fggh
(EL(G)-2: 'x' occurs in 3 nodes etc.

If we keep going we can see that after the L^{th} EL(G) iteration, the regular expression x would be in L+1 nodes. It

also means that a region centered at a node at the center of this path of length L+1 would be influenced by the node x of the seed graph. So, if we have to avoid this region of radius $\lfloor L/2 \rfloor$ centered as above, it means we should avoid the node x in the seed graph.

It follows that since the seed graph is d-connected, we can avoid d-1 regions of $\lfloor L/2 \rfloor$ radius, of the final graph and still stay connected. This would mean that the final graph is d-region based connected for regions of radius $\lfloor L/2 \rfloor$.

For a graph of n nodes where $n = (((..(((a_i d + a_{(i-1)})d + a_{(i-2)})d + a_{(i-3)})d \ldots)d + a_0$ the number of times the EL(G) transformation is applied is $L = \lceil \log_d n \rceil - 2$, since the innermost bracket is the seed graph. In short, from our construction method we know that the number of EL(G) transformations $L = \lceil \log_d n \rceil - 2$, hence we can tolerate d-1 regions being disconnected, with each regions having at most a radius $r = \lfloor (\lceil \log_d n \rceil / 2) - 1 \rfloor$. The number of nodes in each of these regions is given by $(2[(d^{(r+1)} - 1)/(d-1)] - 1)$, where $r = \lfloor (\lceil \log_d n \rceil / 2) - 1 \rfloor$.

It is illustrative to see that the metric of connectivity would have meant these modified line graphs were treated as d-connected. In the context of region based connectivity however, these same networks are d-region based connected and can withstand up to d-1 region failures each region having $2(d^{(r+1)} - 1)/(d-1) - 1$ nodes. To put this in perspective, consider a country wide network of 9999 nodes and degree 10. It would have been deemed to be theoretically robust to up to ***9 specific node failures***; however in real life clustered fault situations it would be robust to 9 regions each of 21 nodes for a total of ***189 node failures***. Or if 9 regions, instead of specific nodes, in the battlefield get catastrophically affected, the rest of the network would still be connected. This illustrates the importance of region based fault tolerance as an important metric for network robustness, and the properties of the modified line graphs as robust networks even with this metric.

V. Conclusion and Future Work

In conclusion, we have formally proved that the class of networks designed by extended line graphs with a diameter of $2\lceil \log_d n \rceil$, are d-connected and are d-region connected, each region of at most $\lfloor (\lceil \log_d n \rceil / 2) - 1 \rfloor$ radius. The total number of nodes in these d-1 regions is bounded by $(d-1)*(2(d^{(r+1)} - 1)/(d-1) - 1)$ which is $2d^{(r+1)} - d - 1$. This work underscores the need for the use of region connected fault tolerance to study real life network robustness.

Future work will consider the alternate definition of a region, namely the geometric region which takes into consideration the physical placement of the nodes and edges. Another metric to study would be how gracefully the diameter degrades with the successive removal of regions.

References

[1] A. Sen, L. Zhou, and B. Hao, "Fault-tolerance in sensor networks: A new evaluation metric" *INFOCOM*, 2006

[2] A. Sen, S. Murthy, S. Banerjee, "Region-Based Connectivity – A New Paradigm for Design of Fault-tolerant Networks", *IEEE 10th International Conference on High Performance Switching and Routing*, 2009.

[3] P. D. Joshi, S. Hamdioui, "Line Graph Based Fast Rerouting and Reconfiguration for Handling Transient and Permanent Faults", *IEEE 15th International Conference on High Performance Switching and Routing,* 2014

[4] P. D. Joshi, S. Hamdioui, "Modified Regular Line Digraphs for Optimal Connectivity and Small Diameters", *Society for Industrial and Applied Mathematics (SIAM) Discrete Mathematics*, 2014.

[5] S. Trajanovski, F. A. Kuipers, P. V. Mieghem, A. Ilic, J Crowcroft, "Critical regions and region-disjoint paths in a network", *IFIP Networking Conference*, 2013

[6] Y. Kobayashi, O. Kensuke, "Max-flow min-cut theorem and faster algorithms in a circular disk failure model", *INFOCOM* 2014.

[7] L. Gewali, H. Selvaraj, D. Mazzella, "Constrained Disjoint Paths in Geometric Networks", *International Conference on Computational Intelligence and Multimedia Applications* 2007.

[8] S. Neumayer, A. Efrat, E. Modiano, "Geographic Max-Flow and Min-Cut Under a Circular Disk Failure Model", *INFOCOM* 2012

[9] C. Berge, *Graphs and Hypergraphs*, Amsterdam, The Netherlands: North-Holland, 1973

[10] M. A. Fiol, I. Alegre, J. L. A. Yebra, "Line digraph iterations and the (d . k) problem for directed graphs," *Proceedings of the 10th International Sympmosium on Computer Architecture*, Stockholm, Sweden, 1983

[11] M. A. Fiol, A. S. Llado, J. L. Villar, "Digraphs on alphabets and the (d.N) digraph problem," *Ars Combinatoria*, vol. 25C, pp. 105-122, 1988

Region Based Containers – A new paradigm for the analysis of Fault Tolerant Networks

Prashant D. Joshi
Cadence Design Systems
Austin, Texas 78759
joship@cadence.com

Arunabha Sen
School of CIDSE
Arizona State University, Tempe, AZ 85281
asen@asu.edu

D. Frank Hsu
Department of Computer and Information Sciences
Fordham University
New York, New York 10458
hsu@cis.fordham.edu

Said Hamdioui; Koen Bertels
Computer Engineering Laboratory
TU Delft, Delft, The Netherlands
shamdioui@tudelft.nl; k.l.m.bertels@tudelft.nl

***Abstract*—Network Fault Tolerance has classically focused on the connectivity of the underlying graph of the network. A k-connected graph will tolerate up to k-1 node or edge failures allowing the remaining nodes to still communicate between them. The introduction of 'Containers' of the underlying graph enabled the measurement of the graceful degradation of the remaining network with the removal of faulty nodes and edges. This metric was required to bound the diameter degradation of the network. Recently, another major metric 'Region Based Connectivity', was introduced to study the locality of the faults in network robustness, by studying the resilience of networks with the loss of regions instead of individual nodes. Since real life networks often have localized outages, it is important to study losses of regions at a time. In this study, we introduce a new concept called 'Region Based Containers' of graphs. This framework will enable the analysis of fault tolerant networks where the two paradigms are brought together to study the graceful degradation of networks when multiple regions are affected.**

In this paper we propose a framework for network QoS using Region Based Containers and its application to fault tolerant design of networks. We then describe an example of networks built by regular Extended Line Graphs and present tight bounds in network degradation with multiple region failures. In the example the diameter of these networks degrade by at most one, despite the failure of d-1 regions where d is the regular degree of the network. The upper bounds on the size of these regions is presented.

This metric is especially applicable to networks where faults are either localized by nature, or faults tend to result in cascading errors in their vicinity, such as power distribution networks, server clusters, or in extreme environments where redundancy of paths is necessary rather than a bonus.

Keywords—Fault Tolerant Networks, Graph Containers, Region Based Connectivity, diameter degradation, Extended Line Graphs

I. INTRODUCTION

Networks have been studied for decades due to their societal applications in civilian and military areas like communication, transportation, power distribution, etc. Recently, applications in sensor networks and applications in low power mobile devices have renewed interest in fault tolerance of these networks. New ideas to measure the degradation of the quality of the results in the presence of faults help better analyze the network, over the traditional metric of connectivity and diameter. A k connected network can tolerate k-1 node or edge failures. The diameter of the underlying graph of the network determines the worst case delay between two communicating nodes. However, these metrics do not indicate what the delay would be in the presence of some faults.

The concept of *d-wide diameter* of a network was introduced by Hsu [2] and a lot of work based on this linked connectivity and diameter in the presence of faults [3, 4]. For some nodes $x, y_1, y_2, \ldots y_w$ of a graph G without self-loops or multiple edges where w is a positive integer and x is not equal to y_i, for any i, a collection of internally node disjoint paths from x to $y_1, y_2, ..y_w$ one for each y_i, is defined as a *star container* from x to $y_1, y_2, \ldots y_w$. In the special case $y_1 = y_2 = \ldots y_w$, the *w-star container* is called a *w-wide container* from x to y. The maximum length of the paths in the *container* is the length of the *container* and w is the width of the *container*. The *w-wide* distance from x to y is the minimum of all possible *container* lengths from x to y and is denoted by denoted by $d_w(x, y)$. The w-wide diameter of a connected graph G is denoted by $D_w(G)$ and defined as $D_w(G) = \max\{d_w(u, v): u, v \in V\}$. A container of a graph helps quantify the degradation of the diameter of the network with faults.

In some practical networks, faults at times get localized. The impact of locality of faults was captured by the new metric Region Based Connectivity introduced in [5]. The metric of region based connectivity helps analyze the connectivity by

978-1-5386-0362-8/17/$31.00

analyzing multiple region failures as against individual nodes or edges.

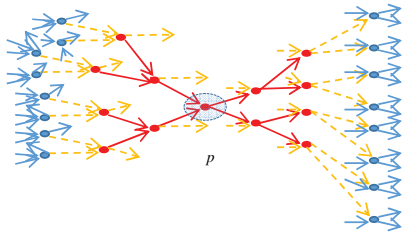


Figure 1. Region of radius two centered at node *p*

A region that fails, results in all nodes in that region becoming incommunicable. A region can be defined in a geometric or a topological sense. A geometric region that fails refers to an actual three dimensional space that fails, while a topological region refers to a set of nodes in the graph in the vicinity of a failing node. Figure 1 shows an example of a topological region.

The red nodes and edges around the node *p* a distance of two behind and ahead of the node *p* (encased in between the dotted edges) are considered to be in the topological region centered at p. The maximum number of nodes in a region is bounded above by $2[(d^{(r+1)}-1)/(d-1)] - 1$ in a regular graph of degree *d* and radius *r*. This work will consider topological regions only.

To the authors' knowledge no work has been done that ties these two metrics together. This paper proposes the use of a framework to analyze networks using the concept of 'Region Based Containers' which will quantify the degradation of the message passing delays in the presence of multiple region failures.

This paper is organized as follows. Section II deals with a short description of some terms used. Section III proposes the framework that should be used and Section IV gives an example of Extended Line Graphs which can be proved to have excellent bounds in Region Based Container diameters. Section V concludes with ideas for extension of this work.

II. Basic Definitions

Most of the terms used in this work are from standard graph theory terminology and can be found in West [1] and other works on *containers* [2-4] and extended line digraphs [6-8]. A digraph $G = (V, E)$ has $n = |V|$ nodes and (p, q) is an element of E if there is a directed edge from the node p to node q. The node p is the predecessor of q, and q is the successor of p. The indegree (correspondingly outdegree) of a node is the number of edges incident into (correspondingly out of) that node. In a regular digraph the indegree and outdegree of all nodes are equal to the degree d. A path from a node p to node q is a sequence of adjacent edges that start from p and end in q. Two node disjoint paths from p to q have no common node except for p and q. A digraph is strongly connected if any node can be reached from any other node in it. The connectivity of a digraph is x if the removal of any $x-1$ nodes still keeps the remaining digraph strongly connected, and the removal of some specific x nodes results in the digraph becoming non-strongly connected. The connectivity is obviously bounded by the minimum degree of any node in the digraph. If a digraph achieves this connectivity, it is called an optimally connected digraph. The distance between two nodes is the number of edges in the shortest path between them. The diameter $k(G)$ of the digraph is the largest value of the distance between any two nodes of the digraph.

A Line Graph of a digraph $G = (V, E)$ is $L(G) = (V1, E1)$ such that $V1 = E$ and $E1 = \{(a,b) \mid a = (u1, u2)$ is an element of E and $b = (u2, u3)$ is an element of $E\}$. The diameter of the line digraph $k(L(G))$ is at most one more than the diameter of G. Also, the connectivity of $L(G)$ is the same as that of G [7].

An Extended Line Graph $EL(G)$, of a regular digraph $G=(V,E)$ of degree d and connectivity d, is $L(G)$ with t additional nodes, t<d so that the number of nodes of $EL(G) = n*d + t$, such that the $EL(G)$ also has degree d, connectivity d, diameter $k(EL(G)) \leq k(G) + 2$. [6, 7].

The *w-wide* diameter of a graph is denoted by $D_w(G)$ and is defined as the $\max(d_w(x, y)$: $x, y \in V)$.

A D_d digraph is a regular circulant digraph (V, E) such that V is a set of nodes $\{0, 1, (n-1)\}$ and $E = \{(a, b)$ s. t. a and b are elements of V; $b = (a + k) \bmod n$, $1 \leq k \leq d\}$. This digraph is known to be d-connected, and have a diameter bounded by $\lceil (n-1)/d \rceil$. The *d-wide* diameter is $\lfloor (n/d) \rfloor + 1$.

Like the concept of a vertex cut, the region cut is the set of regions which when removed results in the remaining graph becoming disconnected. The region based connectivity of a graph or RBC = k, implies up to any $k-1$ regions can be removed without making the remaining graph disconnected. Since any graph of minimum degree d cannot have an RBC > d, a graph that achieves the RBC of the minimum degree will be called an optimally region based connected graph. For a regular graph this would be d.

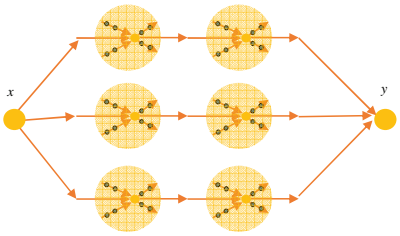


Figure 2. Region Based Container between nodes *x, y*

We define the concept of *d-wide Region Based Container* of a network as follows. For distinct nodes *x, y* of a graph *G* without self-loops or multiple edges and *w, r* positive integers, a collection of *w* internally node disjoint paths from *x* to *y*, such that two nodes on different paths cannot lie in the same region of radius *r* of any other node in the paths. The *d*-wide lengths of such containers across all $x, y \in V$ is the *d*-wide diameter of the Region Based Container of this graph. Figure 2 shows an example of a Region Based Container where there are three region disjoint paths with each region based on each node along the paths having no common node with any other region.

The term network and graph is used interchangeably and all graphs considered here are digraphs.

III. Proposed framework.

The quality of service (QoS) of a network not only should include information on the connectivity of the network and hence the reliability of sending messages, but also the delay in doing so. This delay is a not only a function of the network topology itself but also of the failures at any given time in the network. The number and location of the faults will affect the QoS.

The *w*-wide distance of containers of the underlying graph of the network will bound the degradation of the delay with failures. In a network, $d_w(x, y) = l$ denotes the longest delay in a set of *w* node independent paths between *x* and *y* and is the container length.

By taking all possible containers between the source and the destination, we are indicating that $D_w(G)$ will bound the worst delay in the network with *w*-1 failures no matter what set of *w* node independent paths are used by the network for communication. This is a powerful metric that bounds the QoS with *w*-1 individual failures irrespective of their locality. Note the difference of this metric from the diameter of a network which indicates the largest of the shortest paths between all pairs of nodes. It is illustrative to note that the two definitions coincide when $w = 1$.

On a different note, the Region Based Connectivity is a powerful metric which is especially useful for networks that can have spatially correlated faults, or faults that cascade onto the neighboring nodes, such as in a military environment or a power distribution network. Since the faults in such scenarios often are localized, the analysis might unfairly give pessimistic QoS analysis of such networks by assuming a random location of individual faults. Hence Region Based Connectivity helps give a practical measure of goodness to such networks. The faults localized in one region only are termed Single Region Fault Models (SRFM) while analysis of multiple regions is termed Multiple Region Fault Models (MRFM).

A framework that would consider a QoS of a network that takes into consideration not just the connectivity in terms of multiple faulty regions, but the message delays with region faults also would help network designers bound the QoS despite any type of failures. This captures the diameter degradation in the presence of not just point, but region failures.

This framework would have applications in most areas of networking to help design and quantify up front best case and worst case QoS.

IV. Example Network with Tight bounds

Design of networks using recursive applications of the Extended Line Digraphs has been studied previously [6-8]. The network is designed by first designing a 'seed' digraph which is modified recursively by applying the Extended Line Graph transformation to obtain a regular directed network with a required number of nodes. These networks have been shown to have some very good network qualities of very low diameters, $2\log_d n$, where *n* is the number of nodes with regular degree *d*, and the $D_w(G)$ is known to be $2\log_d n + 1$.

The design method for this family is briefly mentioned here for clarity. To design a network of degree *d* and *n* nodes, write the number *n* to base *d*, as below with each $a_j < d$.

$$n = ((..((a_i d + a_{(i-1)})d + a_{(i-2)})d + a_{(i-3)})d \cdots)d + a_0 \qquad (1)$$

The innermost bracket (highlighted in green) is the seed graph and each Extended Line Graph transformation is the next bracket recursively, the first of that recursion being shown as the highlighted yellow part in Equation 1. Since each Extended Line Graph maintains the connectivity and degree, and increases the diameter and container length by at most two each time, the seed graph becomes the determining factor for the diameter and the $D_w(G)$.

The seed digraph construction consists of two cases resulting in a diameter of four. Let *s* represent the number of nodes required from the seed digraph. If the innermost bracket has $a_i = 1$ and $a_{i-1} = 0$, then let *s* include the next bracket also and hence design the seed digraph with $s = d^2 + a_{(i-2)}$ number of nodes. Note in this case the number of times EL(G) is applied will be less by one.

Case 1: $s < 4d+2$. In this case a simple D_d circulant digraph suffices. Note that when *d* equals two or three, this is the only case that is applicable, since for these constraints the diameter is bounded above by four in a D_d graph.

Case 2: For $s > 4d+1$ design the seed digraph with *d* columns. There will be at least four complete rows. Any extra nodes on the last incomplete row are stacked to the right and on top of the last completed row. Each row will be a D_x digraph and each column will be a D_y digraph as shown in Fig. 3. The value of *y* is (*R*-1) if the number of rows $R < (1 + \lfloor (d/2) \rfloor)$, else *y* is $\lceil d/2 \rceil$, while $x = (d - y)$.

Figure 3 shows an example of building the seed graphs, while Figure 4 shows the process for construction of the final graph.

Any node formed after taking the line graph of a graph G, is named by concatenating the node names of the source and sink of the edge in G. For example if there is an edge between the nodes X and Y, where X and Y can be any sequence of characters, the resulting node of this edge in the line graph would be named XY. X is the left predecessor and Y is the right predecessor of this node XY. This helps in identifying

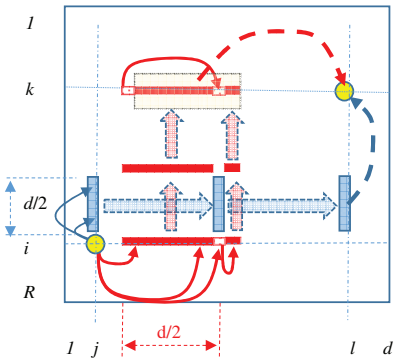


Figure 3 Consider all paths from (i,j) to (k,l) to determine the *d-wide* diameter of the seed graph

which nodes of the final graph are the descendants of which set of nodes in the seed graph.

It can be seen that a node occurring in a path of a seed graphs between two nodes will not occur in any other node disjoint path between the same nodes. By the node naming [6,8] with the Line and Extended Line Graphs, it follows that the final graph will also not share the same node in its node disjoint paths. It is beyond the scope of this paper to show the detailed proof that if a node '*p*' occurs in only one path between two nodes *x, y* of a seed graph, then the final graph will be able to avoid a region based on the node *p*, for the path based on the node *x* to the node based on the node *y*. Also, the size of the region will be at most equal to $2[(d^{(r+1)}-1)/(d-1)] - 1$, where $r = \log_d n - 2$ is the number of times the Extended Line Graph transformation was applied.

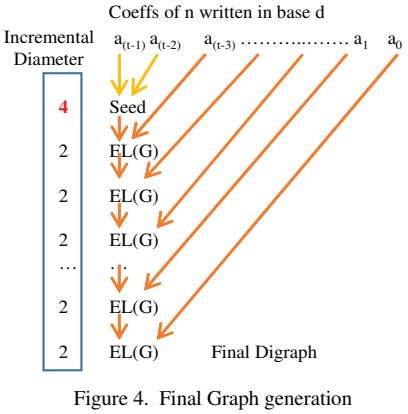


Figure 4. Final Graph generation

This example shows a real network example that has the following properties. The diameter of the network is bounded by $2\log_d n$, while the *d*-wide diameter is bounded above by $2\log_d n + 1$. This shows a very tight bound in the degradation of the message passing delay with not only *d*-1 node failures, but *d*-1 'region failures', where each region size can have a maximum number of nodes equal to $2[(d^{(r+1)}-1)/(d-1)] - 1$ where $r = \log_d n - 2$. It can hence be seen that this network has a Region Based Container of diameter $2\log_d n + 1$ also.

V. Conclusion

This work proposes a framework using a new concept of 'Region Based Container' and is an effort to showcase the usefulness of the merging of two distinct metrics in use today in determining the QoS of networks, namely the Region Based Connectivity and the diameter of the containers based on the underlying graph of the network. Looking at each of them separately does not give the more important metric of how the network would degrade in the presence of region failures, instead of node or link failures.

An example was shown of a family of networks having not just extremely tight diameter and region based connectivity, but also very tight diameters based on the Region Based Containers, showing that this family of networks is very reliable despite region failures without sacrificing QoS. This framework will allow network designers to analyze good estimates of delays under extreme conditions, especially in cases where multiple regions of faults can occur.

As future work, analyses of existing network topologies and comparison with the networks based on recursive extended line graphs will be done.

References

[1] Douglas West, *Introduction to Graph Theory*, Prentice Hall 1996.

[2] D. F. Hsu, "On Container Width and Length in Graphs, Groups and Networks," IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 1994, pp.668-680.

[3] S. Gao and D. Frank Hsu, "Short Containers in Cayley Graphs," Discrete Applied Mathematics 157, 2009, pp. 1354-1363.

[4] Daniela Ferrero, San Marcos, Manju K. Menon, Kalamassery, A. Vijayakumar "Containers and Wide Diameters of P3(G)," Mathematica Bohemica 2012, No. 4, pp. 383–393.

[5] A. Sen, L. Zhou, and B. Hao, "Fault-tolerance in sensor networks: A new evaluation metric," *INFOCOM*, 2006.

[6] P. D. Joshi and S. Hamdioui, "Line Graph Based Fast Rerouting and Reconfiguration for Handling Transient and Permanent Faults," *IEEE 15th International Conference on High Performance Switching and Routing*, 2014, pp. 167-172.

[7] P. D. Joshi and S. Hamdioui, "Modified Regular Line Digraphs for Optimal Connectivity and Small Diameters," *Society for Industrial and Applied Mathematics (SIAM) Discrete Mathematics*, 2014, pp.45-46.

[8] P. D. Joshi, A. Sen, S. Hamdioui, and K. Bertels, "Region Disjoin Paths in a Class of Optimal Line Graph Networks," *International Symposium on Pervasive Systems, Algorithms and Networks*, 2014, pp. 1256-1260.

[9] M. A. Fiol, I. Alegre, and J. L. A. Yebra, "Line digraph iterations and the (d.k) problem for directed graphs," *Proceedings of the 10th International Symposium on Computer Architecture*, Stockholm, Sweden, 1983, pp. 174-177.

[10] M. A. Fiol, A. S. Llado, and J. L. Villar, "Digraphs on alphabets and the (d.N) digraph problem," *Ars Combinatoria*, vol. 25C, 1988, pp. 105-122.

[11] P. D. Joshi, D. Frank Hsu, A. Sen, S. Hamdioui, Koen Bertels, "Tight Bounds in Message Delays Despite Faults in a Class of Line Digraph Networks", *14th International Symposium on Pervasive Systems, Algorithms, and Networks*, 2017.

4

Security

Papers published in this category:

- IEEE DFTS 2014 : Security Methods in Fault Tolerant Modified Line Graph based Networks

Networks require the mitigation of threats to security due to faults and malicious attacks. These attacks can be of many types like Denial of Service, misdirection of data packets or even getting access to sensitive data. At the level of the topology of a network, the types of issues to look into include intentionally or unintentionally misdirected paths. This might require the knowledge of what is the 'correct' path and next node, given the original source and destination of the data.

This class of networks lends itself with being able to identify misdirected data packets due to the small size of the routing tables, and the ability to keep alternate paths in the presence of faults. The method to identify a misdirected packet and forward to the correct one can be done in $O(\log_d n)$ steps.

This study defines a new method called WISH (What I See and Hear) which passively analyzes the paths at each step to ensure that the right path is being traversed. Each node has a color associated with it based on the nodes from the seed torculant graph which was used to generate the final network. The routing table kept in each of the final network's nodes, is based only on the number of nodes in this seed torculant graph.

4.1. Control security with WISH protocol ('What I See and Hear')

The work in this subsection describes the main results of the paper ***Security Methods in Fault Tolerant Modified Line Graph based Networks***. Networks are often designed to withstand failures, which in turn can make them vulnerable to unintentional or malicious attacks. Extensive cryptographic key distribution systems or trusted databases are often used to mitigate the threats. This paper describes a unique way of identifying misdirected messages very quickly using a method called WISH (What I See and Hear).

The procedure uses the fact that the networks designed using the method described in this work use only the routing table based on the seed graph and not the final graph, and hence analyzing it is not a very time consuming task. Each of the nodes of the seed graph are colored with a number based on their names. The message passing along the shortest path hence is a function of the colors along the path. An algorithm determines what the color of each intermediate node should be based on the colors of the shortest forward path and those of the nodes already travelled. Each node performs this check and if that color does not match it's own color then an error is identified. This is practical because of the very small routing tables for these networks.

Security Methods in Fault Tolerant Modified Line Graph based Networks

Prashant D. Joshi
Austin, Texas
prashant.joshi@gmail.com

Said Hamdioui
Delft University of Technology
Delft, The Netherlands
s.hamdioui@tudelft.nl

***Abstract*— Many routing protocols make some assumptions on the correctness of the routing information in the router. This at times allows faults and malicious attacks in the networks. This paper describes a class of networks based on modified line graphs with many features to authenticate the data and controls of the message routing, and having properties of the shortest diameters and easy shortest path calculations.**

We describe this class of fault tolerant networks and the relevant properties. This helps understand the security mechanism, WISH ('What I See and Hear') that probes the data and the routing information to reduce the possibility of router problems, malicious or otherwise.

Keywords—Secure networks, Shortest Path, Route determination, control and data plane checks, Line Graphs, Node naming.

I. INTRODUCTION

Networks are designed to withstand some failures without adversely affecting the network performance. Some of these failures make the networks vulnerable to unintentional as well as malicious attacks. To mitigate these threats security measures have been proposed which at times require extensive cryptographic key distributions systems or a trusted database. Setting a goal of totally error free networks might make them impractical, and so many methods ensure the reduction in the threats instead. In this paper we describe a class of networks which lend themselves to much easier security measures, by enabling checks on the controls and the data which the adversaries would find it very difficult to get around.

This class of networks is based on modified line graphs and a naming structure that limits adversaries from propagating invalid routes in the control domain. We also define a method called WISH (What I See and Hear) which passively analyzes the paths to ensure that the right paths are being traversed.

Studies of networks have involved graphs with nodes as computing elements and the edges as communicating links [4-18]. The smallest degree of any node in the graph is an upper bound on the maximum tolerable node failures to keep the network connected. A network achieving this is optimally fault tolerant. Knowledge of the network helps get the shortest path between any two nodes. To enable shortest path determination in the presence of faults, the good nodes locally redirect traffic as required. The network topology presented here also has the smallest known diameter for this class of networks in published literature, to the best of the authors' knowledge.

This paper is arranged as follows. Definitions of standard and specific terms used in this paper will be followed by the details of the construction of the class of graphs used in this work. This is followed by the threat models that are considered, and how the WISH approaches mitigates many of these threats. We briefly will touch upon future extensions of the work.

II. DEFINITIONS

The terms used in this study relating to graphs can be found in any standard text book on the subject like [1] and [2]. Table 1 briefly gives some of these definitions.

Graph G = (V,E)	V is set of nodes, E is set of edges. Degree is the number of edges at a node. A Uniform graph has all nodes of equal degree. A uniform directed graph (digraph) has equal indegree and outdegree
Path	Path is a sequence adjacent nodes and intermediate edges. The *distance* from a node to another is the shortest number of intermediate edges along any path from the starting to the ending nodes.
Diameter	The diameter of the graph is the maximum distance between any two points of the graph.
Node Connectivity	Minimum number of nodes to be removed to disconnect a graph (that means there is at least 1 pair of nodes which don't have a path between them). *Fault Tolerance* is one less than the node connectivity.
D_d digraph	Uniform digraph (V, E) such that V = {0, 1, …(v-1)} and E = {(a,b) | a, b elements of V; b = (a + k) mod v, $1 \le k \le d$}. An edge of the type x-hop is an edge of the D_d graph from any node y to the node (y+x) mod v. The diameter of a D_d graph is bounded by $\lceil((\lvert V\rvert-1)/d)\rceil$.
Line Graph L(G)	L(G) of a digraph G = (V, E) is L(G) = (V1, E1) such that V1 = E and E1 = {(a,b) | a = (u1, u2) is an element of E and b = (u2, u3) is an element of E }.

Table 1. Graph definitions

978-1-4799-6155-9/14/$31.00 ©2014 IEEE

The predecessor set P(u) and the successor S(u) of a node 'v', an element of V, are defined as P(v) = {u | (u, v) is an element of E} and S(v) = {w|(v,w) is an element of E}.

An Extended Line Graph EL(G) of a uniform digraph G=(V,E) of degree d and connectivity d, is L(G) with t additional nodes, t<d so that the number of nodes of EL(G) = n*d + t, such that the EL(G) also has degree d, connectivity d, and diameter k(EL(G)) ≤ k(G) + 2. The construction of the EL(G) is described later.

Invalid routes in control plane refer to nonexistent paths or incorrect nodes while invalid routes in the data plane refer to routing of data along paths not appropriate shortest paths.

The range of problems that will be addressed here are intermittent and permanent faults as well as misconfigurations and single adversaries.

III. Prior Work

Computer network design for reliability and efficiency has led to many practical fault tolerant networks in use today. Reducing the diameter with and without node faults, increasing connectivity, enabling fast routing algorithms, self-healing, and re-configurability are some of the metrics used. Researchers in [4]-[6] studied generalized hypercubes, restricting nodes to powers of 2, while those in [7]-[9] concentrated on De Bruijn graphs and modifications. In some cases the degree of graphs was compromised as in [6] when the number of nodes is a prime number. Most studies concentrated on keeping the fault tolerance optimal and the diameter proportional to $\log_d n$. The work in [11] also has a suboptimal diameter and restrictions on its degree, while [12] had worse diameter than the method proposed. Properties of line graphs have been studied in other papers like [14], [19] and [20] where the Bruijn-Kautz graphs were analyzed.

In terms of related networking problems, it has been shown that configurations have caused up to 1% of issues in routing tables [31]. One of the main problems has been shown to be the lack of the knowledge of the network topology [31-33] which cause control related problems. On the data plane researchers have built the ability to trace routes involving cryptographic keys, and some active and passive probing to test for correctness [31-35]. Extensive cryptography and key distribution has issues with computation as well as secure distribution of keys [36]. In this study the network topology is inherently encrypted in the node naming which allows quick confirmation of some configuration problems as well as hash function calculations.

IV. Network Construction, Node Naming and Shortest Paths

A. Network Construction using Line Graphs

The line graph of a uniform and optimally connected digraph maintains the degree and connectivity, while the diameter increases by one and the number of nodes becomes n*d [17]. This study extends the concept of a line graph. Given the uniform digraph G=(V,E) of degree and connectivity d, and diameter k(G), we generate EL(G) with n*d + t nodes, 0≤t<d without modifying the degree or the connectivity, though the diameter could go up by 2. These t nodes are referred to as X-nodes, and the other nodes are the non-X-nodes. First obtain the line graph L(G) and a completely connected graph of the t nodes. L(G) has connectivity and degree d, while the graph of the t nodes only has a degree t-1. To merge these graphs to make a uniform digraph, it requires d-(t-1) more edges to and from each X-node.

For some t unique nodes from G: N_i $1 \le i \le t$, randomly pick unique d-(t-1) predecessor nodes P_1, P_2, …$P_{d-(t-1)}$, and d-(t-1) successor nodes S_1, S_2, …$S_{d-(t-1)}$. Since the degree of each node is d, such unique nodes always exist on each of the t unique nodes chosen.

Now for each chosen N_i of G, $1 \le i \le t$, remove its edges (P_j, S_j) $1 \le j \le d-(t-1)$ identified above, from L(G), and instead add the edges (P_j, x_i) and (x_i, S_j). This maintains the degree of the nodes of L(G), and increases the degree of the X-nodes from (t-1) to d-(t-1)+(t-1)=d. If t=0, then EL(G) is the same as L(G).

The so constructed graph is the required EL(G). Fig. 1 shows examples of a G, L(G), and EL(G) graphs. The connectivity is maintained across L(G) and EL(G) from that of G and can be proved, and is not being described in detail here.

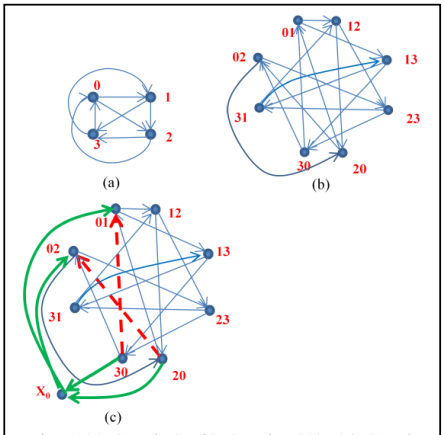


Fig. 1 (a) Graph G. (b) Graph L(G). (c) Graph EL(G) with 1 X-node.

B. Final graph construction

For a uniform digraph of degree d and n nodes, we can write the number 'n' to base d, as below for each a_j<d.

$$n=((\ldots((a_i d + a_{(i-1)})d + a_{(i-2)})d + a_{(i-3)})d\ldots)d+a_0 \qquad (1)$$

Equation (1) shows the EL(G) transformation being applied recursively on the graph of the inner bracket to generate the current bracket. Thus if we can get a good graph for the inner most bracket, then each subsequent bracket simply is the EL(G) of the previous graph, where the diameter of the next graph increases by at most 2.

The last step is constructing a good 'base' graph for the innermost bracket ($a_i d + a_{(i-1)}$). The details of the base graph design and proof that its diameter is bounded by 4, with a connectivity of d, are proved in [23]. With the innermost graph designed with diameter at most 4, each bracket on the outside is essentially a recursive EL(G) transformation as depicted in Fig. 2.

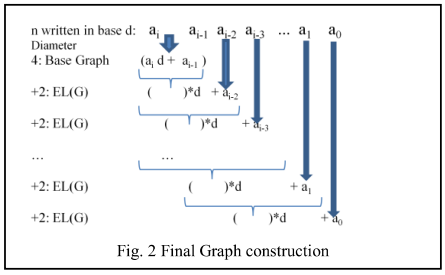


Fig. 2 Final Graph construction

The final diameter is 4 plus two times the number of times the EL(G) transformation is applied which is $\lceil \log_d n \rceil - 2$, and hence the final diameter is bounded above by $2*(\lceil \log_d n \rceil - 2)+4 = 2*\lceil \log_d n \rceil$. This result is the best known diameter for the class of optimally fault tolerant directed networks where there is no restriction on the number of nodes or the degree, to the best of the authors' knowledge.

C. *Node naming and shortest path example*

Any node formed after taking the line graph of a graph G, is named by concatenating the node names of the source and sink of the edge in G. For example if there is an edge between nodes named X and Y, where X and Y can be any sequence of characters, then the resulting node of this edge in the line graph would be the string X concatenated by Y. X is the left predecessor and Y is the right predecessor of this node XY. This identifies the descendant nodes of the base graph, among those of the final graph. Fig. 3 gives an example of the naming.

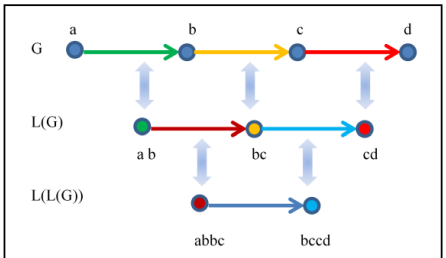


Fig. 3. Node naming with successive L(G) transformations

For example, let us consider Fig. 4 where a graph of n=14 and d=2 is constructed based on the described method above, and try to trace a shortest path between the nodes 0110 to 2112.

0110→ ? →2112

This implies that in the penultimate EL(G) graph, we have to find a path from 10 to 21, and then so on, from 0 to 2 as shown below.

0110 **(10→ ? →21)** 2112

0110(10**(0→ ? →2)**21)2112

Since 0→2 exists in the base graph, we have been able to go down recursively until the base graph to find the required path. Now we will trace back up to retrace the nodes at each step.

0110→(10→02→21)2112

0110→1002→0221→2112

14 base 2 = 1110
Base graph (3 nodes)
0→1,2 1→2,0 2→0,1
L(G) (6 nodes)
01→12,10 02→20,21 12→20,21
10→01,02 20→01,02 21→12,10
EL(G) (7 nodes)
01→x x→12
21→x x→10
01→~~12~~,10 02→20,21
12→20,21 10→01,02
20→01,02 21→12,~~10~~
L(G) (14 nodes)
01x → x12, x10 x12 →1220, 1221
21x →x12, x10 x10 →1001, 1002
0110→ 1001, 1002 0220 → 2001, 2002
0221 → 2112, 21x 1220 → 2001, 2002
1221 → 2112, 21x 1001 → 01x, 0110
1002 → 0220, 0221 2001 → 01x, 0110
2002 → 0220, 0221 2112 → 1220, 1221
Diameter is guaranteed to be bounded above by $2*\lceil \log_2 14 \rceil$ and in this case the diameter is 4.

Fig. 4 Example of a graph of 14 nodes with degree 2. The arrow indicates that the node to the left of the arrow has edges to node(s) after the arrow

If 0221 is known to be faulty, then avoiding either the edge 02 or 21 will never result in the node 0221 in use in any path. Hence we could then take 0→1→2 at the base graph level and thus get the path: 0110(10(0→1→2)21)2112

0110→1001→0112→1221→2112 thus avoids the faulty node. It is possible that we might not have to go down to the base graph since an intermediate level might have a direct edge and in which case the path is between non-diametric nodes.

D. *Routing Table for shortest path*

In networks, a routing table is a data structure in the form of a matrix, which lists the routes to all the destinations, next node hope information and other information. The routing table is generated out of the knowledge of the topology of the entire network which determines statically the next node, based on the current node and the final destination. These tables are used for packet forwarding.

Consider a typical table of n rows, one for each destination. To conserve memory, each node only keeps track of the next

hop information from itself, based on the final destination, along with some other information. If there is a need to store multiple paths, there could be multiple next node options for each row's final destination. There would hence be as many rows as there are nodes in the network. In contrast, the size of the routing table in this construction is only ($a_i d + a_{(i-1)}$), as described below.

base graph	EL(G)	EL(G)	EL(G)
a	ab	abbc	abbcbccd
b	bc	bccd	bccdcdde
c	cd	cdde	cddedeef
d	de	deef	deefeffg
e	ef	effg	*effgfggh*
f	fg	fggh	fgghghhi
g	gh	ghhi	ghhihiij
h	hi	hiij	hiijijjk
i	ij	ijjk	
j	jk		
k			

Fig. 5. Field of Influence and shortest path determination

Consider the shortest path between two nodes say, abbcbccd to hiijijjk in Fig. 5. A node 'y' below another 'x' along a column means the edge x→y exits at that EL(G) level, and each column shows the result of the EL(G) of the previous column. Only the shortest path between the rightmost predecessor of the source node 'd' and the leftmost predecessor of the sink node 'h' is required to be known in routing tables. These tables do not need to be of n rows, but of at most d^2-1 rows (maximum number of nodes in the base graph). From the routing table, once the shortest path (shaded in yellow) from the first column of the base graph is known to be d→e→f→g→h, then the full path is automatically known due to the node naming. In addition, this knowledge enables each node to know how other nodes will behave, thus helping in identifying nodes that are not behaving correctly.

If it is known that effgfggh is faulty, then to avoid this node, the path in the base graph from d →…→h can avoid at least one of the edges: ef, fg, gh and by doing so this regular expression will not be present in any of the nodes of the path in the final graph thus ensuring the shortest path is still used despite the faulty node.

To→ From↓	0	1	2
0	n/a	1,2	2,1
1	0,2	n/a	2,0
2	0,1	1,0	n/a

Fig 6. Sample routing table. Top row is the final destination, left column is the current node and the entries are prioritized next nodes.

Consider again the example graph of Fig 4 of 14 nodes. The routing table to be stored in each of these 14 nodes is shown in Fig. 6.

This information is sufficient for us to determine the shortest path from any node to any other node in the final graph of Fig 4. The reason is that the node naming allows us to determine the shortest path, and if the recursively going back on the EL(G) columns, we end up at the base graph, we use the above routing table. If we see a path in an intermediate level, we don't even need the routing table.

Notice also that although the final graph might contain any number of nodes, the routing table needs to have only ($a_i d + a_{(i-1)}$) rows and columns (at most d^2-1). So in the example of Fig. 4 and Fig. 6, the same routing table would suffice if the final graph had more EL(G) transformations, since the base graph and its shortest paths would be the same. Each row indicates the starting node, and the column number represents the final node of the base graph. Every entry contains the next node to take, to get the shortest path. In case of known faulty nodes, the routing table will allow avoiding specific edges of the base graph, based on the regular expression of the faulty node.

If the faulty nodes are having transient faults and the status at a given time is known to all the nodes, the routing can be done by avoiding these nodes. On the other hand, if this transient behavior is seen to be more permanent, then a decision can be made to make it as a permanent fault thus enabling actions to reconfigure the network if required.

V. SECURITY WITH 'WISH'

A. *Control plane security with 'What I See'*

Here we describe the 'What I See' part of the WISH method, which helps identify invalid routes due to mis-configuration since as the node naming gives some protection. Note two nodes can only be adjacent only if their names are such that the right half of the regular expression of the source is the same as the left half of the receiver's regular expression. Thus any router indicating that the next node with an incorrect name will raise an alarm. Similarly, the routing table of the base graph gives information on which nodes exist and what the paths would be. Each node also has a color associated with it based on it's the node name.

A source node sends the packet with the following information along with the packet:

1. Source node
2. Final Destination
3. Sum of all colors of the nodes along the evaluated shortest path (Intermediate nodes change it to the Running Total of path ahead).

Each node when it receives the packet, sees the final destination in the packet and checks to see if per its calculations of the shortest path, if it should have received the packet from the predecessor. If it is not the case, it raises and alarm of control problems from the predecessor. It however, passes the packet onwards based on its shortest path calculations.

From Fig. 4 consider the shortest path:

0110→1002→0221→2112

If the node 1002 decides to forward the path to 0220 instead of 0221, then when 0220 gets the packet and knows the starting and final nodes, using only the nodes names and the routing table, it will know that 1002 has mis-forwarded the packet. This evaluation needs $\log_d n$ steps and hence is not computationally intense. There is no need for any key to be securely distributed either. The packet is forwarded on from here to the final destination with the alarm of the incorrect path from the previous node. In the remedial steps, the number of such issues is noted and beyond a predetermined threshold, it would be treated as a permanent failure and the node would be avoided in the shortest path determination as described in the previous sections.

B. 'What I Hear' issues.

Each node has a color assigned to it based on the node name. Each source node adds the color of all nodes along the path from it to the final destination, and includes it in the packet. Every subsequent node subtracts its color from the sum in the packet and passes it on to the next node. In addition, it finds the sum of the colors from it to the destination, and subtracts the sum obtained from its predecessor. This result should be the same as its own color. This test can catch some incorrectly routed packets.

To keep the sums small, it is possible to do the addition modulo some number also. Table 2 shows an example of how this is done. The starting node calculates the shortest path from S1 to S4, and adds the color of all intermediate nodes modulo 7 as the forward total. Each subsequent node, calculates the forward total from itself to the destination, and subtracts it from the incoming forward total, modulo 7 (note, only positive values). This difference should be the same as the color of the node.

Node	S1	S2	S3	S4
Color	3	5	4	6
Forward total	(5+4+6) mod 7=1	(4+6) mod 7 =3	6 mod 7=6	0
Incoming total minus Forward total = Color of node)		(1-3) mod 7 = 5	(3-6) mod 7 = 4	(6-0) mod 7 = 6

Table 2. "What I hear" test

The effort to do this calculation is to find the forward total from each node. This is the addition of at most $2\log_d n$ numbers at the starting node, and then adding one less number along each node in the path, which is again of the order of $\log^2 n$.

C. Remedial steps based on 'What I See and Hear' issues.

Periodically each node sends information to others on how many packets were received and if there were any issues seen on the control side. If packets are being dropped or incorrect paths are being traversed on a regular basis, then the knowledge of these problem nodes are passed on to other nodes. The number of nodes on the path is bounded by $2\log_d n$, and the number of steps required to test for the 'What I See' is $\log_d n$. Hence the total effort required would be of the order of $\log^2 n$.

Each node along the path would be doing these 'What I See' checks, and keeping track of the errors seen. It would be difficult for a malicious node to deliberately misroute, without altering the contents of the packet itself, like the sender or final destination node.

Any issues with the 'What I Hear' inaccuracies are another indication that there has been a problem in the shortest path in use.

In conjunction with the control and data plane checks, the 'What I See and Hear' will restrict the malicious nodes abilities to misroute packets.

VI. CONCLUSION

In this paper we described the security features associated with a class of uniform directed networks constructed based on Extended Line Graphs. The line graph's node naming method results in an upper bound on the routing table size, and is independent of the size of the network, which enables $O(\log_d n)$ steps to determine the shortest path in the network, with or without the presence of node faults. The shortest path determination problem is reduced to finding the shortest path in at most d^2-1 nodes independent of the number of nodes.

Any node routing a packet incorrectly will be identified by the next node due to the control feature of 'seeing' the proper shortest path and next nodes from the start to the destination node. Similarly 'hearing' the running color total and comparing with forward total results in a check in the data side. Future work will look into extending this research to help isolate multiple faulty nodes, and estimate the effort to reconfigure the network in the presence of permanent faults.

REFERENCES

[1] C.Berge, Graphs and Hypergraphs. Amsterdam, The Netherlands: North-Holland, 1973.

[2] A.J. Hoffman and R.R. Singleton, On Moore graphs with diameter 2 and 3, IBM J. Res. Develop. 4 (1960) 497–504.

[3] W. T. Tutte, A family of cubical graphs, Proceedings of the Cambridge Philosophical Society, 43 (1947) 459–474.

[4] J. R. Armstrong and F. G. Gray, "Fault diagnosis in Boolean n-cube array of microprocessors," IEEE Trans. Comput., vol. C-30, pp. 590-596, Aug. 1981.

[5] J. Kuhl and S. M. Reddy, "Distributed fault tolerance for large multiprocessor systems," in Proc. 7th Annual Symposium Computer Architecture, May 1980

[6] L. Bhuyan and D.P. Agrawal, "Generalized hypercube and hyperbus structure for a computer network," IEEE Trans Computers, vol. C-33, Apr. 1984

[7] M. L. Schlumberger, "DeBruijn communication networks," Ph.D. dissertation, Stanford Univ., Stanford, 1974.

[8] D.K. Pradhan and S.M. Reddy, "A fault tolerant communication architecture for distributed systems" IEEE Transactions Computers vol. C-31, Sept. 1982.

[9] D. K. Pradhan, Z. Hanquan, and M. L. Schlumberger, "Fault tolerant multibus architecture for multiprocessors," in Proc. 14th Int. Conference on Fault-Tolerant Computers, 1984, pp. 400-408.

[10] M. Imase, T. Soneoka, and K. Okada, "Connectivity of regular directed graphs with small diameters" IEEE Transactions on Computers, vol. C-34, March 1985.

[11] U. Schumacher, "An algorithm for k-connected graph with minimum number of edges and quasiminimal diameter," Networks, vol. 14, 1984.

[12] A.Sengupta, P. D. Joshi and S. Bandyopadhyay,"A Synthesis Approach to Design Optimally Fault Tolerant Network Architecture", IEEE Transactions on Computers, vol.40, January 1991.

[13] Daniela Ferrero and Carles Padro, "Connectivity and fault-tolerance of hyperdigraphs", Discrete Applied Mathematics, 2002.

[14] M. A. Fiol, I. Alegre, and J. L. A. Yebra, "Line digraph iterations and the (d . k) problem for directed graphs," in Proceedings of the 10th International Sympmosium on Computer Architecture, Stockholm, Sweden, 1983.

[15] M. A. Fiol, A. S. Llado, and J. L. Villar, "Digraphs on alphabets and the (d.N) digraph problem," Ars Combinatoria, vol. 25C, pp. 105-122, 1988.

[16] M. A. Fiol, J. L. A. Yebra, and I. Alegre, "Line digraph iterations and the (d . k) digraph problem," IEEE Transactions on Computers, vol C-33, pp. 400-403, May 1984.

[17] S.M. Reddy, J.G. Kuhl, and S.H. Hosseini, "On digraphs with minimum diameter and maximum connectivity," in the Proceedings of the 20th Annual Allerton Confernce, Oct 1982.

[18] D.K. Pradhan, "Fault tolerant multiprocessor link and bus network architecture," IEEE Transactions on Computers, vol. C-34, Jan. 1985

[19] Liu, S. Trajanovski, P. Van Mieghem, "Reverse Line Graph Construction: The Matrix Relabeling Algorithm MARINLINGA Versus Roussopoulos's Algorithm", Delft University of Technology submission to xrXiv.org. October 2010.

[20] J. Naor and M. B. Novick, "An efficient reconstruction of a graph from its line graph in parallel." J. of Algorithms, 11(1): 132-143, 1990.

[21] J. Suurballe and R. Tarjan, "A quick method for finding shortest pairs of disjoint paths," Networks, vol. 14, pp. 325–336, 1984

[22] Dahai Xu, Yang Chen, Yizhi Xiong, Chunming Qiao, Xin He, "On the Complexity of and algorithms for finding the shortest path with disjoint counterpart", IEEE/ACM Transactions on Networking, vol. 14, No. 1, February 2006.

[23] Prashant D. Joshi, Said Hamdioui, "Modified uniform line digraphs with optimal connectivity and small diameteres", Forty-Fifth Southeastern International Conference on Combinatorics, Graph Theory and Computing, 2014.

[24] Amitabh Trehan, "Algorithms for Self-Healing Networks", Ph.D. dissertation, University of New Mexico., New Mexico, 2010.

[25] Alper Mizrak,Yu-Chung Cheng, Keith Marzullo, Stefan Savage, "Fatih: detecting and isolating malicious routers", in the Proc. of The International Conference on Dependable Systems and Networks, 2005.

[26] Robert Poor, Charlotte Auburn and Cliff Bowman,"Self Healing Networks", ACM Queue, May 2003.

[27] Mihaela Enachescu, Mei Wang, Ashish Goel, "Reducing Maximum Stretch in Compact Routing", in the Proc. of IEEE INFOCOM, 2008.

[28] Dmitri Krioukov, Kevin Fall, Xiaowei Yang, "Compact Routing on Internet-Like Graphs", in the Proc. of IEEE INFOCOM, 2004

[29] Aifei Zhong, Srehari Nelakuditi, Yinzhe Yu, Sanghwan Lee, Junling Wang, Chen-Nee Chuah, "Faulure inferencing based fast rerouting for handling transient link and node failures", in the Proc. of IEEE INFOCOM, 2005.

[30] Saia, J., Trehan, A., "Picking up the pieces: self-healing in reconfigurable networks" IEEE International Symposium on Paralleland Distributed Processing, 2008.

[31] R. Mahajan, D. Wetherall, T Anderson, "Understanding BGP misconfigurations" Proc of ACM SIGCOMM, Aug 2002.

[32] X. Zhao, et al, "An analysis of BGP multiple origin AS (MOAS) conflicts" ACM SIGCOMMIMW, 2001.

[33] L. Subramanian, V. Roth, I. Stoica, S. Shenker, R. H. Katz, "Listen and Whisper: Security Mechanisms for BGP", USENIX NSDI, 2004.

[34] Z. Mao, J. Rexford, J. Wang, R. H. Katz, "Towards an accurate AS-level tracerouter tool", ACM SIGCOMM, 2003.

[35] D. Zhu, M. Gritter, D. Cheriton, "Feedback based routing", Proceedings of Hotnets, 2002.

[36] S. Kent, C. Lynn, K. Seo, "Secure Border Gateway Protocol *Secure-BGP)", IEEE Journal on Selected Areas of Communication, April 2000

5

Conclusion

This chapter summarizes the overall achievements of this dissertation and highlights some future research directions. Section 5.1 presents a summary of the main conclusions presented in this dissertation. Thereafter, Section 5.2 recommends future research directions.

5.1. Summary

This research work is focused on describing the mathematical procedure to design optimally fault tolerant regular directed networks with no restriction on the number of nodes or the degree of each node. The procedure introduced a new topology called a '**torculant**' which was the merger of a torus and a circulant, to serve as a seed graph. This seed graph when tranformed recursively using a new procedure called the '**Extended Line Graph**' resulted in the diameter being $2\log_d n$, where *d* is the degree and *n* is the number of nodes, which is the best known in literature without such a restriction. The routing tables were shown to be a function of *d* and not *n* which makes them orders of magnitude smaller enabling a large number of beneficial properties. These properties include finding the shortest paths very quickly and efficiently with and without the presence of node faults, thus enabling self-healing in the presence of known faults. In addition, the degradation of the diameter in the presence of up to (*d* - 1) node faults is only one, which is an extremely tight bound which shows the efficacy of this method.

Subsequently, this research explored the concept of topological region based connectivity on this family of networks. It was shown that these networks can withstand up to (*d* - 1) topological region faults, and an upper bound on the size of each region was shown to be at most $2(\frac{d^{r+1}-1}{d-1})$–1 nodes, where *r* is the radius of the region. A new concept called '**region based containers**' was introduced to study the degradation of the diameter in the presence of topological region failures. The robustness of this method was underscored by showing that the diameter of the network is degraded by one despite these region failures.

It is important to emphasize that this new topology shows a lot of promise in comparison with the existing supercomputer network topologies. It has important scientific and technical implications to society, in that the performance of supercomputers can be further improved without detriment to the complexity or power consumption. This thesis has shown that it is possible to increase the number of processing elements by orders of magnitudes for the same constraints as allowed by some implementations of some of the fastest supercomputers of today. On the other hand, for the same number of processing elements, the performance would go up many fold.

Chapter 1, "Introduction", briefly introduced the topologies and issues around the topologies of the supercomputers of the last decade. Thereafter, it showed that the impressive exponential gain in the performance over the last few decades has primariliy been due to the increase in the number of processing elements that have been networked together using good topologies. This then formed the motivation of the thesis and the focus was on reliability, robustness and security. The thesis focused on a new topology that is obtained by a series of recursive steps performed on a new topology called the 'torculant'.

Chapter 2, "Reliability", focused on the design of very high performance reliable topologies with graceful degradation which can be designed into the specifications of the system. It showed how networks can be designed with extremely low delays without increasing the degree of the networks. In addition it was shown that these

networks can be designed with specifications that let the degradation be very tightly bound in the presense of a new metric dealing with region faults in addition to point faults.

Chapter 3, "Robustness", focused on the ability of these networks to function in spite of faults. The ability of 'self-healing' when specific types of faults are detected, enable the system to work outside of its specifications, thus making it robust. Such rerouting on the fly, taking into possible faults that are region based gave rise to a new paradigm for network analysis. It was shown that on this new metric, the proposed robust networks outperform the existing supercomputing topologies.

Chapter 4, "Security", proposed a new security protocol (WISH), to identify misdirected messages with or without the presense of faults. This was possible because the controls for tracking the message paths were dependent on a very small routing table, orders of magnitude smaller than what would be otherwise required.

5.2. Future Research Directions

Several recommendations can be suggested to further improve the state-of-the-art. They are organized by the different aspects of the research topics as listed below.

- Reliability

It might be desireable in the specification of a network topology to define among other constraints discussed, the average delays with and without faults instead of the maximum delays. Such analysis and comparison with existing topologies will help give a more realistic performance expectations instead of only bounding the worst. It might also be instructive to look at some real network loads and do some simulations with the various topologies.

Another interesting extension to the reliability specifications deals with the delays on broadcast of messages from a node to all others. This would require more study of the *w*-star diameters of the existing networks as well as the proposed network.

- Robustness

An extention to robustness would involve the physical placement of the nodes in a real server environment. This will help extend the analysis of robustness to not just topological but also geometric region connectivity of the system.

- Security

Security is by far one of the fastest growing concerns in networks as of late. One of the important extensions to this work in the area of security would be to design routing algorithms which will help identify faulty nodes, malicious or otherwise, to isolate and remove them from the network automatically. The framework for such methods is available with the WISH protocol proposed, and it would be illustrative to implement real networks to test such new algorithms.

References

1. **A. C. Aljundi, J. Dekeyser, T. Kechadi, and Isaac Scherson**, *A Study of an Evaluation Methodology for Unbuffered Multistage Interconnection Networks*, Proceedings of the 17th International Parallel and Distributed Processing Symposium, 2003.

2. **M. Feldman and A. Snell**, A New High Performance Computing Fabric for HPC, Intersect360 Research, May 2016.

3. **J. Duato, S. Yalamanchili, and N. Lionel**, *Interconnection Networks: An Engineering Approach*, Morgan Kaufmann Publishers Inc., 2003.

4. **W. Dally**, *Performance Analysis of k-ary n-cube Interconnection Networks*, IEEE Transactions on Computers, 39(6):775–785, June 1990.

5. **C. E. Leiserson**, *Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing*, IEEE Transactions on Computers, Vol. C-34, No 10, Oct 1985.

6. **A. Sengupta, A. Sen and S. Bandyopadhyay**, *Fault-Tolerant Distributed System Design*, IEEE Transactions on Circuits and Systems, Vol.35, No. 2, pp. 168-172, February 1988.

7. **M. Besta and T. Hoefler**, *Slim Fly: A Cost Effective Low-Diameter Network Topology*, Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, November 2014.

8. **J. Kim, W. Dally, and D. Abts**, *Flattened Butterfly: A Cost Efficient Topology for High-Radix Networks*, Proceedings of the 34th Annual International Symposium on Computer Architecture, pages 126– 137. ACM, 2007.

9. **G. Ostrouchov**, *Parallel computing on a hypercube: an overview of the architecture and some applications*, 19th Symposium on the Interface of Computer Science and Statistics, March 1987.

10. **U. Gulzari , M. Sajid M, S. Anjum, S. Agha and F. Torres**, *A New Cross-By-Pass-Torus Architecture Based on CBP-Mesh and Torus Interconnection for On-Chip Communication*, PLoS ONE11(12): e0167590, 2016.

11. **Top500 supercomputers** Top 500 The List

12. **H. Fu, J. Liao, J. Yang, et a**l, *The Sunway TaihuLight Supercomputer: System and Applications*, Science China Information Sciences 2016.

13. **H. Lin, X. Tang, B. Yu, et al**, *Scalable Graph Traversal on Sunway TaihuLight with Ten Million Cores*, IEEE International Parallel and Distributed Processing Symposium 2017.

14. **P. D. Joshi and S. Hamdioui**, *Shortest Path Reduction in a Class of Uniform Fault Tolerant Networks*, IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, October 2014.

15. **P. D. Joshi, D. Frank Hsu, A. Sen, S. Hamdioui and K. Bertels**, *Tight Bounds in Message Delays Despite Faults in a Class of Line Digraph Networks*, 14th International Symposium on Pervasive Systems, Algorithms and Networks, June 2017.

16. **P. D. Joshi, D. Frank Hsu, A Sen, S. Hamdioui and K. Bertels**, *Region Based Containers – A new paradigm for the analysis of Fault Tolerant Networks*, IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, October 2017.

17. **C. Minkenberg**, *Interconnection Network Architectures for High-Performance Computing*, Advanced Computer Networks, Guest Lecture, IBM Research, Zurich, May 2013.

18. **A. Sen, I. Zhou and B. Hao**, *Fault-tolerance in sensor networks: A new evaluation metric*, 25th Conference on Computer Communications, INFOCOM April 2006.

19. **P. D. Joshi, A. Sen, S. Hamdioui and K. Bertels**, *Region Disjoin Paths in a Class of Optimal Line Graph Networks*, 13th International Symposium on Pervasive Systems, Algorithms and Networks, December 2014.

20. **Y. Ajima et al.**, *(2014) Tofu Interconnect 2: System-on-Chip Integration of High-Performance Interconnect.* **In:J. M. Kunkel, T. Ludwig, H. W. Meuer(eds) Supercomputing. ISC 2014.** Lecture Notes in Computer Science, vol 8488. Springer, Cham

21. **P. D. Joshi and S. Hamdioui**, *Line Graph Based Fast Rerouting and Reconfiguration for Handling Transient and Permanent Node Failures*, IEEE 15th International Conference on High Performance Switching and Routing, December 2014.

22. **P. D. Joshi and S. Hamdioui**, *Security Methods in Fault Tolerant Modified Line Graph based Networks*, IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, October 2014.

23. **P. D. Joshi and S. Hamdioui**, *Short Containers in Modified Line Digraphs*, Society for Industrial and Applied Mathematics (SIAM) Discrete Mathematics, SIAM-DM 2016.

24. **P. D. Joshi and S. Hamdioui**, *Modified Uniform Line Digraphs with Optimal Connectivity and Small Diameters*, 45th International SE Conference on Combinatorics, Graph Theory and Computing 2014.

25. **P. D. Joshi and S. Hamdioui**, *Modified Regular Line Digraphs for Optimal Connectivity and Small Diameters*, Society for Industrial and Applied Mathematics (SIAM) Discrete Mathematics, SIAM-DM 2014.

26. **W. Quattrociocchi, G. Caldarelli and A. Scala, Self-Healing Networks: Redundancy and Structure, PLOS ONE, 2014**.

27. **M. Bharadwaj**, *C2 Torus New Interconnection Network Topology Based on 2D Torus*, American Journal of Networks and Communication, Special Issue: Ad Hoc Networks, Vol. 4, No. 3-1, 2015.

28. **H. Gu, K. Chen and Y. Yang**, *MRONoC: A Low Latency and Energy Efficient On-Chip Optical Interconnect Architecture*, IEEE Photonics Journal, Vol: 9, Issue 1, February 2017.

29. **S. Faralli, F. Gambini, P. Pintus et al.**, *Bidirectional Transmission in an Optical Network on Chip with Bus and Ring Topologies*, IEEE Photonics Journal, Vol. 8, Number 1, February 2016.

30. **A. Sen, S. Murthy and S. Banerjee**, *Region-Based Connectivity – A New Paradigm for Design of Fault-tolerant Networks*, IEEE 10th International Conference on High Performance Switching and Routing, 2009.

31. **S. Trajanovski, F. A. Kuipers, P. Van Mieghem, A.Ilic and J. Crowcroft**, *Critical regions and region-disjoint paths in a network*, IFIP Networking Conference, 2013.

32. **Y. Kobayashi andO. Kensuke**, *Max-flow min-cut theorem and faster algorithms in a circular disk failure model*, INFOCOM 2014.

33. **L. Gewali, H. Selvaraj and D. Mazzella**, *Constrained Disjoint Paths in Geometric Networks*, International Conference on Computational Intelligence and Multimedia Applications 2007.

34. **S. Neumayer, A. Efrat and E. Modiano**, *Geographic Max-Flow and Min-Cut Under a Circular Disk Failure Model*, INFOCOM 2012.

35. **M. A. Fiol, I. Alegre and J. L. A. Yebra**, *Line digraph iterations and the (d,k) problem for directed graphs* Proceedings of the 10th International Symposium on Computer Architecture, 1983.

36. **M. A. Fiol, A. S. Llando and J. L. Villar**, *Digraphs on alphabets and the (d,N) digraph problem*, Ars Combinatoria, Vol. 25C, pp. 105-122, 1988.

37. **A. J. Hoffman and R. R. Singleton**, *On Moore graphs with diameter 2 and 3*, IBM J. Res. Development 4 (1960) 497-504.

38. **W. T. Tutte**, *A family of cubical graphs*, Proceedings of the Cambridge Philosophical Society, 43 (1947) 459-474.

39. **J. R. Armstrong and F. G. Gray**, *Fault diagnosis in Boolean n-cube array of microprocessors*, IEEE Transactions of Computers, Vol C-30, August 1981.

40. **J. Kuhl and S. M. Reddy**, *Distributed fault tolerance for large multiprocessor systems*, Proceedings of the 7th Annual Symposium on Computer Architecture, May 1980.

41. **L. Bhuyan and D. P. Agrawal**, *Generalized hypercube and hyperbus structure for computer network*, IEEE Transactions on Compuers, Vol. C-33, April 1984.

42. **M. L. Schlumberger**, *DeBruijn communication networks*, PhD. Dissertation, Stanford University, Stanford 1974.

43. **D. K. Pradhan and S. M. Reddy**, *A fault tolerant communication architecture for distributed systems*, IEEE Transactions on Computers, Vol. C-31, September 1982.

44. **D. K. Pradhan, Z. Hanquan and M. L. Schlumberger**, *Fault tolerant multibus architecture for multiprocessor*, Proceedings of the 14th International Conference on Fault Tolerant Computers, 1984.

45. **M. Imase, T. Soeoka and K. Okada**, *Connectivity of regular directed graphs with small diameters*, IEEE Transactions on Computers, Vol. C-34, March 1985.

46. **U. Schumacher**, *An algorithm for k-connected graph with minimum number of edges and quasiminimal diameter*, Networks, Vol. 14, 1984.

47. **A. Sengupta, P. D. Joshi and S. Bandyopadhyay**, *A Synthesis Approach to Design Optimally Fault Tolerant Network Architecture*, IEEE Transactions on Computers, Vol 40. January 1991.

48. **D. Ferrero and C. Padro**, *Connectivity and fault-tolerance of hyperdigraphs*, Discrete Applied Mathematics, 2002.

49. **S. M. Reddy, J.G. Kuhl and S. H. Hosseini**, *On digraphs with minimum diameter and maximum connectivity*, Proceedings of the 20th Annual Allerton Conference, Octorber 1982.

50. **D. K. Pradhan**, *Fault tolerant multiprocessor link and bus network architecture*, IEEE Transactions on Computers, Vol C-34, January 1985.

51. **J. Naor and M. B. Novick**, *An efficient reconstruction of a graph from its line graph in parallel*, Journal of Algorithms, 1990.

52. **J. Suurballe and R. Tarjan**, *A quick method for finding the shortest pairs of disjoint paths*, Netowrks, Vol 14. 1984.

53. **D. Xu, Y. Chen, Y. Xiong, C. Qiao and X. He**, *On the Complexity of an algorithm for finding the shortest path with disjoint counterpart*, IEEE/ACM Transactions on Networking, Vol 14. No. 1, February 2006.

54. **A. Trehan**, *Algorithms for Self-Healing Networks*, Ph.D. Dissertation, University of New Mexico 2010.

55. **A. Mizrak, Y. C. Cheng, K. Marzullo and S. Savage**, *Fatih: Detecting and isolating malicious routers*, Proceedings of the International Conference on Dependable Systems and Networks, 2005.

56. **R. Poor, C. Auburn and C. Bowman**, *Self Healing Networks*, ACM Queue, May 2003.

57. **M. Enachescu, M Wang and A. Goel**, *Reducing Maximum Stretch in Compact Routing*, INFOCOM 2008.

58. **D. Krioukov, K Fall, and X. Yang**, *Compact Routing on Internet-Like Graphs*, INFOCOM 2004.

59. **A. Zhong, S. Nelakuditi, Y. Yu et al.**, *Failure inferencing based fast rerouting for handling transient link and node failures*, INFOCOM 2005.

60. **J. Saia and A. Trehan**, *Picking up the pieces: Self healing in reconfigurable networks*, IEEE International Symposium on Parallel and Distributed Processing, 2008.

61. **D. F. Hsu**, *On Container Width and Length in Graphs, Groups and Networks*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 1994.

62. **S. Gao and D. F. Hsu**, *Short Containers in Cayley Graphs*, Discrete Applied Mathematics, 2009.

63. **D. Ferrero, S. Marcos, M. K, Menon and K. A. Vijayakumar**, *Containers and Wide Diameters of P3(G)*, Mathematica Bohemica 2012.

64. **M. S. Krishnamoorthy and B. Krishnamurthy**, *Fault diameter of interconnection networks*, Computers and Mathematics with Applications Vol I3, 1987.

65. BlueGene/L Architecture, Advanced Simulation and Computing, Lawrence Livermore National Laboratory.

66. **B. Barney**, Using the Sequoia and Vulcan BG/Q Systems, Lawrence Livermore National Laboratory.

67. **T. Inoue**, The 6D Mesh/Torus Interconnect of K Computer, Fujitsu Limited.

68. **X. Wang, et al.**, *Modeling region-based interconnection for interdependent networks*, Physical review. E 94 4-1, 2016.

69. **E. W. Weisstein**, Circulant Graph, MathWorld–A Wolfram Web Resource.

70. **F. Harary and R. Z. Norman**, *Some properties of line digraphs*, Rendiconti del Circolo Matematico di Palermo, 9 (2), 1960.

71. **H. Whitney**, *Congruent graphs and the connectivity of graphs*, American Journal of Mathematics, 54 (1), 1932.

72. **F. Harary, and C. St. J. A. Nash-Williams**, *On eulerian and hamiltonian graphs and line graphs*, Canad. Math. Bull. 8, 1965.

73. **L. Xiong, Z. Liu**, *Hamiltonian iterated line graphs*, Discrete Mathematics 256, 2002.

74. **P.A. Catlin, Iqbalunnisa, T.N. Janakiraman, N. Srinivasan**, *Hamilton cycles and closed trails in iterated line graphs*, Journal of Graph Theory 14, 1990.

75. **G. Chartrand**, *On hamiltonian line graphs*, Trans. Amer. Math. Soc. 134, 1968.

76. **S. Han, H. Q. Ngo, L. Ruan, D.-Z. Du**, *Transmission Fault-Tolerance of Iterated Line Digraphs*, TR 00-058 Technical report, Department of Computer Science and Engineering University of Minnesota, 2000.

77. **F. Cao, D.-Z. Du, S. Han, D. Kim, and T. Yu**, *Line digraph iterations and diameter vulnerability*, Taiwanese Journal of Mathematics 1999

78. **D. Bauer, R. Tindell**, *Graphs with prescribed connectivity and line graph connectivity*, J. Graph Theory, 1979.

79. **Y. Mao**, *Path connectivity of line graphs*, arXiv:1603.03995 , 2016

80. **S. Trajanovski, F. A. Kuipers, A. Ilić, J. Crowcroft, and P. Van Mieghem**, *Finding Critical Regions and Region-Disjoint Paths in a Network*, IEEE/ACM Transactions on Networking 23(3), May 2015.

81. **A. Xie, X. Wang, and S. Lu**, *Risk Minimization Routing Against Geographically Correlated Failures*, IEEE Access PP(99):1-1, May 2019

82. **X. Wang, M. Chen, and S. Lu**, *Modeling Geographically Correlated Failures to Assess Network Vulnerability*, IEEE Transactions on Communications PP(99):1-1, August 2018

83. **M. Conran**, NETWORK STRETCH, Network, Security, and Cloud, 2014.

List of Publications

1. **Prashant D. Joshi** and S. Hamdioui, *Shortest Path Reduction in a Class of Uniform Fault Tolerant Networks*, IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, October 2014.

2. **Prashant D. Joshi**, D. Frank Hsu, A. Sen, S. Hamdioui and K. Bertels, *Tight Bounds in Message Delays Despite Faults in a Class of Line Digraph Networks*, 14th International Symposium on Pervasive Systems, Algorithms and Networks, June 2017. **(Best Paper Award)**.

3. **Prashant D. Joshi**, D. Frank Hsu, A Sen, S. Hamdioui and K. Bertels, *Region Based Containers – A new paradigm for the analysis of Fault Tolerant Networks*, IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, October 2017.

4. **Prashant D. Joshi**, A. Sen, S. Hamdioui and K. Bertels, *Region Disjoin Paths in a Class of Optimal Line Graph Networks*, 13th International Symposium on Pervasive Systems, Algorithms and Networks, December 2014.

5. **Prashant D. Joshi** and S. Hamdioui, *Line Graph Based Fast Rerouting and Reconfiguration for Handling Transient and Permanent Node Failures*, IEEE 15th International Conference on High Performance Switching and Routing, December 2014.

6. **Prashant D. Joshi** and S. Hamdiou, *Security Methods in Fault Tolerant Modified Line Graph based Networks*, IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems, October 2014.

7. **Prashant D. Joshi** and S. Hamdioui, *Short Containers in Modified Line Digraphs*, Society for Industrial and Applied Mathematics (SIAM) Discrete Mathematics, SIAM-DM 2016.

8. **Prashant D. Joshi** and S. Hamdioui, *Modified Uniform Line Digraphs with Optimal Connectivity and Small Diameters*, 45th International SE Conference on Combinatorics, Graph Theory and Computing 2014.

9. **Prashant D. Joshi** and S. Hamdioui, *Modified Regular Line Digraphs for Optimal Connectivity and Small Diameters*, Society for Industrial and Applied Mathematics (SIAM) Discrete Mathematics, SIAM-DM 2014.

Curriculum Vitæ

Prashant D. Joshi

Prashant D. Joshi was born in Pune, India in 1964. He received his Bachelor of Technology degree in Computer Science and Engineering from the Indian Institute of Technology, Bombay in 1985, followed by his Masters in Computer Science from the University of South Carolina, Columbia S.C. in 1987.

Prashant started his career in the industry at Intel Corporation working on microprocessor designs. He has worked on the designs of its flagship processors like the Intel i486SL, Pentium II®, Pentium® 4 and the Intel Atom®, and many proliferations thereof. His work encompassed many areas of design, including validation, transistor level custom circuit design, cell based design, static timing anlaysis, noise anlysis, clock distribution, memory designs etc. on Intel's many leading products. He has also worked in Broadcom Corporation and IBM Corporation working in areas like synthesis, place and route, full chip timing and noise. At Cadence Design Systems he enabled marquee customers on their next generation design issues. Currently he is a Senior Engineering Manager at Intel working on Atom® designs.

Prashant has published about 25 journal and conference papers in the area of low power, high performance digital design and fault tolerant networks. He is active in IEEE activities and was the Chair of the IEEE DFT in 2011 and 2012, and has been its Industry Liaison since then. He has served on the program committees of several conferences and journals, and was a guest editor of The Journal of Electronic Testing, Theory and Applications - Special Issue on Defect and Fault Tolerance (Springer, 2012). He holds a patent in the area of modeling full and half cycle clock variability. He has given invited talks at conferences, including keynote demos, and at the 'Distinguished Speaker Series' at Cadence, and lectures at The University of Texas at Austin. His research interests include areas of low power circuit designs and fault tolerant network designs.