A GEOMETRIC ROUTING SCHEME IN WORD-METRIC SPACES FOR DATA NETWORKS

Miguel Hernando Camelo Botero

Dipòsit legal: Gi. 1844-2014
http://hdl.handle.net/10803/283749

DOCTORAL THESIS

A GEOMETRIC ROUTING SCHEME IN WORD-METRIC SPACES FOR DATA NETWORKS

Miguel Hernando Camelo Botero

2014

DOCTORAL THESIS

A GEOMETRIC ROUTING SCHEME IN WORD-METRIC SPACES FOR DATA NETWORKS

Miguel Hernando Camelo Botero

2014

DOCTORAL PROGRAM IN TECHNOLOGY

Supervisors:
Ph.D. Lluís Fàbrega and Ph.D. Pere Vilà

This thesis is presented in fulfillment of the requirement for the conferral of the degree of Doctor of Philosophy by the University of Girona

Universitat de Girona

Los Dres. Lluís Fàbrega y Pere Vilà, del Departament d'Arquitectura i Tecnologia de Computadors de la Universitat de Girona,

DECLARAMOS:

Que el trabajo titulado A GEOMETRIC ROUTING SCHEME IN WORD-METRIC SPACES FOR DATA NETWORKS, que presenta Miguel Hernando Camelo Botero para la obtención del título de doctor/a, se ha realizado bajo nuestra dirección y que cumple los requisitos para poder optar a Mención Internacional.

Y para que así conste y tenga los efectos oportunos, firmamos el presente documento.

Lluís Fàbrega

Pere Vilà

Girona, septiembre de 2014

Acknowledgments

First at all, all the credit of this thesis is to God, who made all this possible. In the last years, many persons have influenced, encouraged, and helped me to the completion of this thesis. I would like to start my list of thanks with my supervisors Lluís Fàbrega and Pere Vilà for the guidance, encouragements, support, patience and also for having faith in me. I also would like to thank Dimitri Papadimitriou, who hosted me at Alcatel Lucent Bell labs (Antwerp, Belgium) and provided rigorous and valuable feedback during all the stages of my research.

I thank to Ramón Fabregat and José Luís Marzo, who gave me the opportunity to work in the Broadband Communication and Distributed Systems (BCDS) research group, and to Yezid Donoso for being always a guidance in all my academical processes. In this list, I can not forget my co-workers (and ex co-workers) at BCDS research group (Silvi, Carlitos, Fer, Nico, Ludy, Juan, Marc), who I had the opportunity of sharing good moments of friendship with them and interesting discussions and support about the work in this dissertation. I would like to include in this list to Oscar, Alejo, Fabis, Jito, Juanda and Isa who have been as my family during all this time in Girona and Alejo (kvi), Luis and Julian, my best friends who always have been there to support me.

I am grateful to Montse, Cristina y Anna for sorting out conference trips, research stays and all the paperwork needed during this research work. I appreciate the valuable feedback that the anonymous reviewers of my PhD thesis document and of our papers have provided throughout these years.

Finally, I would like to express the most special thank you to my family for all its unconditional love, which brings me the strength to achieve any personal goal, and to my lovely Christine, who now is an important part of my life and who give me the love, support and motivation to complete this process. This thesis is especially dedicated to them.

This research work is partially funded by the European Commission (EULER project, FP7 258307), the Spanish Government (RoGER project, TEC 2012-32336), the Generalitat of Catalonia (CSI project, SGR-1202), and the SUR of the Generalitat of Catalonia and the European Social Fund (PhD grant FI-DGR 2011).

List of Acronyms

AuS Automatic Structure

AS Autonomous System

FSA Finite State Automata

MWP minimum-length word problem

AG Automatic Group

SAS Shortlex Automatic Structure

CG Cayley Graph

ERT Equivalent Routing Table

IA Index Automaton

CRS Confluent Rewriting System

WA Word-Acceptor Automaton

GM General-Multiplier Automaton

WD Word-Differences Automaton

DC Data Center

CRP Compact Routing Problem

GGR Greedy Geometric Routing

CR Compact Routing

GR Geometric Routing

BFS Breadth-First Search

SPRT Shortest Path Routing in Trees

GPS Global Positioning System

GPSR Greedy Perimeter Stateless Routing

GOAFR Greedy Other Adaptive Face Routing

GPGF Gravity-Pressure Greedy Forwarding

GNP Global Network Positioning

SWDC Small-World Datacenter

APSP All Pairs Shortest Path

SP Shortest Path

MDST Minimum Diameter Spanning Tree

MdST Minimum Degree Spanning Tree

AMST Approximated Minimum Spanning Tree

DFS Depth-First Search

RT Routing Table

BGP Border Gateway Protocol

IR Internet Router

CIDR Classless Inter-Domain Routing

IoT Internet of Things

SPRT Shortest Path Routing on Trees

CDF Cumulative Distribution Function

BA Barabasi-Albert preferential attachment

WM Word-Metric

GRH2 Greedy Geometric Routing in $\mathbb{H}^2$

HK Holme-Kim preferential attachment

RTR Reducing Table Ratio

C-GGR Compact Greedy Geometric Routing

GF Greedy Forwarding

List of Tables

List of Figures

Related Author's Publications

Here we list the international journals and proceedings of international conferences where this work has been published:

[1] M. Camelo, D. Papadimitriou, P. Vilà, and L. Fàbrega. "Geometric Routing with Word-Metric Spaces". In: *IEEE Communications Letters (indexed in the Journal Citation Reports Science Edition 2012, Impact Factor 1.463, Q2 in Telecommunications)*. Accepted. 2014.

[2] M. Camelo, P. Vilà, L. Fàbrega, and D. Papadimitriou. "Cayley-Graph-based Data Centers and Space Requirements of a Routing Scheme using Automata". In: *2014 IEEE 34rd International Conference on Distributed Computing Systems Workshops*. Madrid, Spain: IEEE Computer Society, 2014, pp. 63–69. ISBN: 978-0-7695-5199-9.

[3] M. Camelo, D. Papadimitriou, L. Fàbregas, and P. Vilè. "Efficient Routing in Data Center with Underlying Cayley Graph". In: *Complex Networks V*. Ed. by P. Contucci, R. Menezes, A. Omicini, and J. Poncela-Casasnovas. Vol. 549. Studies in Computational Intelligence. Springer International Publishing, 2014, pp. 189–197. ISBN: 978-3-319-05400-1.

[4] M. Camelo et al. "Functional Model of a Routing System Architecture". In: *Proceedings of the 1st Workshop "Future Internet: Efficiency in high-speed networks" (W-FIERRO 2011)*. Ed. by P. P. Mariño. July 2011, pp. 19–26. ISBN: 978-84-96997-69-1.

Contents

Abstract

It is commonly recognized by the scientific and technical community of the Internet that its current routing system architecture suffers from a scalability problem, because the fast (exponential) growth in its number of nodes (routers and Autonomous Systems), as well as other factors, translates into a similar growth in the size of the Routing Table (RT). This makes Internet routers to require an excessive amount of storage to maintain the RT and a high processing to make routing decisions. A similar problem arises in Data Centers (DCs), where the emergence of new paradigms such as Cloud Computing, Smart Cities and the Internet of Things (IoT), have also increased exponentially their number of nodes (servers and network devices). This growth degrades the performance of the DC routing system and has a negative impact on energy consumption and environment. As a consequence, the design of scalable routing schemes for the Internet and DC is of major importance.

The problem known as Compact Routing (CR) consists in designing routing schemes that achieve scalable RT size with respect to the number of network nodes n, i.e., the RT size grows sub-linearly (or lower) in n, with vertex labels (i.e., node identifiers) of logarithmic size in n, and low stretch. The (multiplicative) stretch of the routing scheme is defined as the worst (highest) ratio between the length of the path produced by the routing scheme and the length of the shortest path (for the same source-destination pair), among all source-destination pairs (we consider that the length of the path is its number of hops).

As a potential solution to the Compact Routing Problem (CRP), Greedy Geometric Routing (GGR) has been proved to be both simple and heuristically effective (a GGR scheme that solves the CRP is called a Compact Greedy Geometric Routing (C-GGR) scheme). This family of routing schemes assigns some (virtual) coordinates in a metric space to each node through the process called embedding. By forwarding packets to the closest neighbor node (in this space) to the destination, they ensure a completely local process with the RT size bounded by the maximum vertex degree. However, the GGR schemes proposed so far experience one or more of the following problems: 1) they do not guarantee packet delivery, 2) they produce vertex labels of size linear (or higher) in n,

3) they can not be implemented in a distributed way, 4) they require a full knowledge of the network topology, or 5) they have unbounded stretch.

In this work we propose a novel and simple embedding of any connected finite graph into a Word-Metric space, i.e., a metric space generated by algebraic groups. By combining word processing in groups with graph search algorithms, we prove that any GGR scheme built on top of this embedding guarantees the packet delivery (the embedding is said to be "greedy"). Then, for any graph H with n nodes, m edges, maximum vertex degree Δ_H, and spanning tree T_H with diameter $D(T_H)$, we propose the following three GGR schemes:

- A GGR scheme for any kind of graph, with stretch of $O(D(T_H))$, $O(D(T_H) \cdot log(\Delta_H))$ bits per vertex label, RTs of size $O(\Delta_H \cdot D(T_H) \cdot log(\Delta_H))$ and routing decisions that take $O\left(\Delta_H^2 \cdot D(T_H)\right)$ steps.
- A C-GGR scheme for scale-free graphs (which include many real-world topologies such as Internet), with stretch of $O(log(n))$, $O(log^2(n))$ bits per vertex label, RTs of size $O(n^{1/2} \cdot log^2(n))$ and routing decisions that take $O(n \cdot log(n))$ steps.
- A C-GGR scheme for Cayley Graphs (which are used as a model for Data Center interconnection networks), with shortest paths, $O(log(n) \cdot log(log(n)))$ bits per vertex label, routing tables of size $O(log^2(n) \cdot log(log(n)))$, and routing decisions that take $O\left(log^3(n)\right)$ steps.

While the first GGR scheme works for any kind of graph and its complexity depends on the parameters $D(T_H)$ and Δ_H, the two C-GGR schemes are specialized, and their complexity only depends on n. In addition, these C-GGR schemes can be constructed in a distributed way in $O(log(n))$ steps, using $O(n + log(n) \cdot m)$ messages of size $O(log^2(n))$ and $O(\Delta_H \cdot log(n))$ bits of additional storage to build the RT, i.e., not only the RT is scalable but also the routing scheme itself.

Experimental evaluation through simulation of the C-GGR scheme for scale-free graphs and the C-GGR scheme for Cayley Graphs shows that the stretch, the vertex label and the RT size are well below the theoretical upper bounds, and that these results are better in comparison with other well-known routing schemes.

Resumen

Está ampliamente reconocido por la comunidad científica y técnica de Internet que la arquitectura actual del sistema de encaminamiento sufre un problema de escalabilidad, ya que el rápido (exponencial) crecimiento en su número de nodos (routers y sistemas autónomos), así como otros factores, se traslada a un crecimiento similar del tamaño de la tabla de encaminamiento (*Routing Table* o RT). Esto hace que los routers de Internet requieran una cantidad excesiva de espacio de almacenamiento para mantener estas tablas y que las decisiones de encaminamiento también requieran un elevado procesamiento. Un problema similar aparece también en los centros de datos (*Data Centers* o DC), donde la aparición de los nuevos paradigmas de computación en la nube, ciudades inteligentes y la Internet de las cosas, han provocado un crecimiento exponencial en su número de nodos (servidores y elementos de red). Este crecimiento degrada el rendimiento del sistema de encaminamiento y esto acaba teniendo un impacto negativo en el consumo de energía y en el medio ambiente. Como consecuencia, el diseño de esquemas de encaminamiento escalables para Internet y centros de datos es de una gran importancia.

El problema conocido como encaminamiento compacto (*Compact Routing* o CR) consiste en diseñar esquemas de encaminamiento que consigan que el tamaño de la tabla de encaminamiento sea escalable con respecto al número de nodos de la red n, es decir, que el tamaño de la tabla de encaminamiento crezca de un modo sub-linear (o inferior) a n, con unas etiquetas de nodos (o sea, identificadores de nodos) de tamaño logarítmico respecto a n, y manteniendo un *stretch* bajo. El *stretch* (multiplicativo) de un esquema de encaminamiento se define como la peor (o la más alta) ratio entre la longitud de los caminos producidos por el esquema de encaminamiento y la longitud del camino más corto (para el mismo par origen-destino), entre todos los posibles pares origen-destino (consideramos que la longitud de un camino es su número de saltos).

Como posible solución al problema del encaminamiento compacto (*Compact Routing Problem* o CRP), se ha propuesto el uso de esquemas de encaminamiento geométrico greedy (*Greedy Geometric Routing* o GGR), los cuales han demostrado ser esquemas simples y heurísticamente efectivos (un esquema GGR que resuelve el problema CRP es llamado un *Compact Greedy Geometric Routing* o C-GGR). Esta familia de esquemas

de encaminamiento asigna algún tipo de coordenadas (virtuales) de un espacio métrico a cada nodo a través de un proceso que se llama incrustación (*embedding*). Haciendo que los nodos retransmitan los paquetes al nodo vecino más cercano (en este espacio métrico) al destino, se consigue un proceso es completamente local y un tamaño de la tabla de encaminamiento limitada por el grado máximo de los nodos. Sin embargo, los esquemas GGR propuestos hasta hoy presentan uno o más de los siguientes problemas: 1) no garantizan la entrega de todos los paquetes, 2) producen etiquetas de los nodos de tamaño linear (o superior) respecto a n, 3) no pueden ser implementados de una forma distribuida, 4) necesitan una visión global de la topología de la red, o 5) presentan un *stretch* no limitado.

En este trabajo se propone un nuevo y simple método de incrustación de un grafo finito y conectado cualquira en un espacio métrico de palabras (*Word-Metric space*), es decir, un espacio métrico generado por grupos algebraicos. Combinando el procesamiento de palabras en grupos con algoritmos de búsqueda en grafos, hemos demostrado que cualquier esquema GGR construido sobre esta incrustación garantiza la entrega de todos los paquetes (es decir, que el incrustado es "*greedy*"). Entonces, para cualquier grafo H con n nodos, m enlaces, grado nodal máximo Δ_H, y un árbol recubridor (*spanning tree*) T_H con diámetro $D(T_H)$, proponemos los siguientes tres esquemas GGR:

- Un esquema GGR para cualquier tipo de grafo, con un *stretch* de $O(D(T_H))$, etiquetas con $O(D(T_H) \cdot log(\Delta_H))$ bits, tablas de encaminamiento de tamaño $O(\Delta_H \cdot D(T_H) \cdot log(\Delta_H))$ y donde las decisiones de encaminamiento se toman en $O\left(\Delta_H^2 \cdot D(T_H)\right)$ pasos.

- Un esquema C-GGR para grafos libres de escala (*scale-free*) (los cuales incluyen muchas topologías del mundo real como Internet) con un *stretch* de $O(log(n))$, etiquetas con $O(log^2(n))$ bits, tablas de encaminamiento de tamaño $O(n^{1/2} \cdot log^2(n))$ y donde las decisiones de encaminamiento se toman en $O(n \cdot log(n))$ pasos.

- Un esquema C-GGR para grafos de Cayley (los cuales se usan de modelo para las redes de interconexión en centros de datos), con *stretch* 1 (todos los caminos son los más cortos), etiquetas con $O(log(n) \cdot log(log(n)))$ bits, tabla de encaminamiento de tamaño $O(log^2(n) \cdot log(log(n)))$ y donde las decisiones de encaminamiento se toman en $O\left(log^3(n)\right)$ pasos.

Mientras que el primer esquema GGR propuesto funciona en cualquier tipo de grafo y su complejidad depende de los parámetros $D(T_H)$ y Δ_H, los dos esquemas C-GGR son especializados y sus complejidades dependen solo de n. Además, estos esquemas

C-GGR se pueden implementar de una manera distribuida en $O(log(n))$ pasos, usando $O(n+log(n)\cdot m)$ mensajes de tamaño $O(log^2(n))$ y usando un espacio de almacenamiento adicional de $O(\Delta_H \cdot log(n))$ bits para construir la tabla de encaminamiento, es decir, no solo los esquemas especializados propuestos presentan una tabla en encaminamiento escalable, sinó tambiém los mismo esquemas de enrutamiento.

Se ha realizado una evaluación experimental, a través de simulaciones, del esquema C-GGR para grafos de escala libre y del esquema C-GGR para grafos de Cayley, donde se ha comprobado que el *stretch*, el tamaño de las etiquetas y el tamaño de la tabla de encaminamiento están muy por debajo de las cotas superiores teóricas, y que dichos resultados son mejores en comparación con otros esquemas de encaminamiento conocidos.

Resum

Està àmpliament reconegut per la comunitat científica i tècnica d'Internet que l'arquitectura actual del sistema d'encaminament presenta un problema d'escalabilitat, ja que el ràpid (exponencial) creixement en el seu nombre de nodes (routers i sistemes autònoms), així com altres factors, es trasllada a un creixement similar de la longitud de la taula d'encaminament (*Routing Table* o RT). Això fa que els routers d'Internet requereixin d'una quantitat excessiva d'espai d'emmagatzemament per mantenir aquestes taules i que les decisions d'encaminament també requereixin d'un elevat processament. Un problema similar apareix en els centres de dades (*Data Centers* o DC), on l'aparició dels nous paradigmes de informàtica en núvol, ciutats intel·ligents i Internet de les coses, han provocat un creixement exponencial en el seu nombre de nodes (servidors i elements de xarxa). Aquest creixement degrada el rendiment del sistema d'encaminament i això acaba tenint un impacte negatiu en el consum d'energia i en el medi ambient. Com a conseqència, el disseny d'esquemes d'encaminament escalables per a Internet i centres de dades és d'una gran importància.

El problema conegut com encaminament compacte (*Compact Routing* o CR) consisteix en dissenyar esquemes d'encaminament que aconsegueixin que la longitud de la taula d'encaminament sigui escalable respecte del nombre de nodes de la xarxa n, és a dir, que la longitud de la taula d'encaminament creixi d'un mode sub-linear (o inferior) a n, amb unes etiquetes dels vèrtexs (o sigui, identificadors dels nodes) d'una longitud logarítmica respecte a n, i mantenint un *stretch* baix. L'*stretch* (multiplicatiu) d'un esquema d'encaminament es defineix com la pitjor (o la més alta) ràtio entre la longitud dels camins produïts per l'esquema d'encaminament i la longitud del camí més curt (per al mateix parell origen-destí), entre tots els possibles parells origen-destí (considerem que la longitud d'un camí és el seu nombre de salts).

Com a possible solució al problema de l'encaminament compacte (*Compact Routing Problem* o CRP), s'ha proposat l'ús d'esquemes d'encaminament geomètric greedy (*Greedy Geometric Routing* o GGR), els quals han demostrat ser esquemes simples i heurísticament efectius (un esquema GGR que resol un problema CRP s'anomena *Compact Greedy Geometric Routing* o C-GGR). Aquesta família d'esquemes

d'encaminament assigna algun tipus de coordenades (virtuals) d'un espai mètric a cada node a través d'un procés que s'anomena incrustaci (*embedding*). Fent que els nodes retransmetin els paquets al node veí més proper (en aquest espai mètric) al destí, s'aconsegueix un procés completament local i una longitud de la taula d'encaminament limitada per el grau màxim dels vèrtexs. No obstant, els esquemes GGR proposats fins avui presenten un o més dels segents problemes: 1) no garanteixen l'entrega de tots els paquets, 2) produeixen etiquetes dels vèrtexs d'una longitud linear (o superior) respecte a n, 3) no poden ser implementats de forma distribuïda, 4) necessiten una visió global de la topologia de la xarxa, o 5) presenten un *stretch* no limitat.

En aquest treball es proposa un nou i simple mètode d'incrustaci d'un graf finit connectat qualsevol en un espai mètric de paraules (*Word-Metric space*), és a dir, un espai mètric generat per grups algebraics. Combinant el processament de paraules en grups amb algoritmes de cerca en grafs, hem demostrat que qualsevol esquema GGR construït amb aquest incrustaci garanteix l'entrega de tots els paquets (és a dir, que l'incrustaci és "*greedy*"). Per tant, per a qualsevol graf H amb n nodes, m enllaços, grau màxim de vèrtex de Δ_H, i un arbre recobridor (*spanning tree*) T_H amb un diàmetre $D(T_H)$, proposem els següents tres esquemes GGR:

- Un esquema GGR per a qualsevol tipus de graf, amb un *stretch* de $O(D(T_H))$, etiquetes amb $O(D(T_H) \cdot log(\Delta_H))$ bits, taula d'encaminament de longitud $O(\Delta_H \cdot D(T_H) \cdot log(\Delta_H))$ i on les decisions d'encaminament es prenen en $O\left(\Delta_H^2 \cdot D(T_H)\right)$ passos.

- Un esquema C-GGR per a grafs lliures d'escala (*scale-free*) (que inclouen moltes topologies del món real com ara Internet) amb un *stretch* de $O(log(n))$, etiquetes amb $O(log^2(n))$ bits, taula d'encaminament de longitud $O(n^{1/2} \cdot log^2(n))$ i on les decisions d'encaminament es prenen en $O(n \cdot log(n))$ passos.

- Un esquema C-GGR per a grafs de Cayley (que s'utilitzen de model per a les xarxes d'interconnexió en centres de dades) amb un *stretch* 1 (tots els camins són els més curts), etiquetes amb $O(log(n) \cdot log(log(n)))$ bits, taula d'encaminament de longitud $O(log^2(n) \cdot log(log(n)))$ i on les decisions d'encaminament es prenen en $O\left(log^3(n)\right)$ passos.

Mentre que el primer esquema GGR proposat funciona en qualsevol tipus de grafs i la seva complexitat depèn dels paràmetres $D(T_H)$ i Δ_H, els dos esquemes C-GGR són especialitzats i les seves complexitats depenen només de n. A més a més, aquests esquemes C-GGR es poden implementar d'una manera distribuïda en $O(log(n))$ passos, utilizant $O(n + log(n) \cdot m)$ missatges de longitud $O(log^2(n))$ i usant un espai d'emmagatzemament addicional de $O(\Delta_H \cdot log(n))$ bits per a construir la taula d'encaminament, és a dir, no només els esquemes especialitzats proposats presenten una taula d'encaminament escalable sinó també ells mateixos esquemes d'encaminament.

S'ha realitzat una avaluació experimental, a través de simulacions, de l'esquema C-GGR per a grafs d'escala lliure i de l'esquema C-GGR per a grafs de Cayley, on s'ha comprovat que l'*stretch*, la longitud de les etiquetes i la longitud de la taula d'encaminament estan molt per sota de les fites superiors teòriques, i que aquests resultats són millors en comparació amb altres esquemes d'encaminament coneguts.

Chapter 1

Introduction

In this chapter we present the motivation for this research work and the desired objectives. We also describe the structure and contents of this document.

1.1 Motivation

The Internet is a large, dynamic, heterogeneous collection of independently administered computer networks, where each one has its own administration, rules, and policies. There is no central authority overseeing the growth of this network-of-networks, where connections and computers are being added/deleted every day. The Internet topology can be viewed as an undirected graph, where a vertex represents either a router (Internet Router (IR) level topology) or an Autonomous System (AS) (AS level topology). Although both models represent the Internet at different levels, they present similar statistical and structural properties at large-scale (Chapter 5, [1]). Both models are characterized by heavy tailed vertex degree distributions following a power-law form, and the presence of shortcuts that connect far away parts of the network, thus reducing the average path length of the graph, one of the main characteristic of small-world networks, i.e., they are scale-free networks [2]. Figure 1.1 shows an example of the Internet topology at the level of IR and AS.

The current inter-AS routing scheme in the Internet is based on the Border Gateway Protocol (BGP) [3]. Each network node maintains a Routing Table (RT) that contains the next neighbor node per destination, information that is used for packet forwarding. In the early years of the Internet, the hierarchical topology together with the assignment of addresses based on topological location using Classless Inter-Domain Routing (CIDR), made the Internet to achieve high address aggregation and hence small RTs in routers [4]. However, the de-aggregation of addresses, traffic engineering, network dynamics and routing policies, among others, have broken this model, and even if the number of

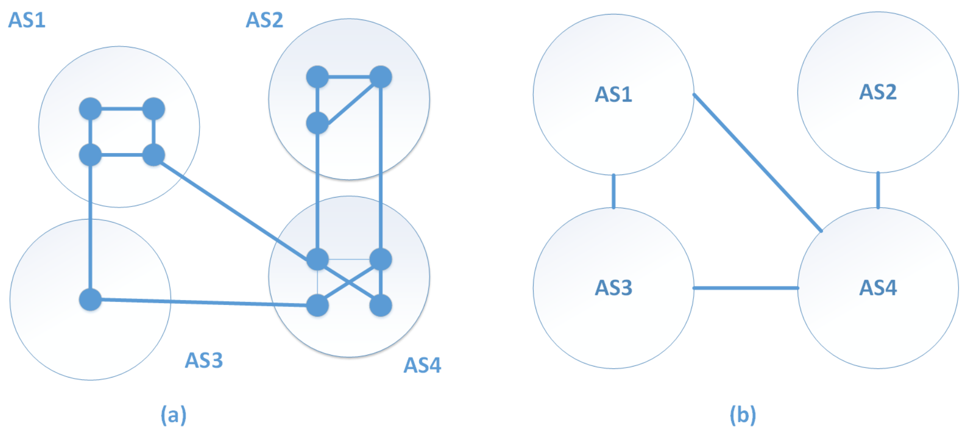


Figure 1.1: The Internet topology represented at different level: a) Internet Router and b) Autonomous Systems.

network nodes were constant, the size of the RTs would grow. Recent measurements have shown that the growth of the RT size is exponential (see Figure 1.2), and therefore routers require excessive amount of storage to maintain the RT, a high processing to make routing decisions, and high number of update messages of the routing protocol.

A lot of partial solutions (or patches) have been developed and deployed to solve the above problems, but the Internet Architecture Board recognizes that this patching methodology will not sustain the continuous growth of the Internet at an acceptable cost and speed. The fundamental problem is the poor scalability of the routing scheme based on BGP. BGP achieves the shortest path but it requires RT of size $\Omega(n \cdot log(n))$, assuming $log(n)$ bits per label, and being n the number of network nodes. In fact, it has been proved that this is the lowest bound of the RT size for any shortest path algorithm that works on any kind of network [5]. Instead of this super-linear growth (in terms of n), the desired routing system scalability would be supra-linear in n, and ideally proportional to $log(n)$ [4].

A similar problem arises in Data Centers (DCs), where the emergence of new paradigms such as Cloud Computing, Smart Cities, Bring your Own Device and the Internet of Things (IoT), have also increased exponentially their number of nodes (servers and network devices) in order to provide large-scale storage and computing for services in various domains such as e-commerce, health-care, smart-grids, and other. A DC (sometimes called a server farm) is a centralized repository for the storage, management, and dissemination of data and information. Figure 1.3 shows an example of a Cloud Computing infrastructure. Their interconnection networks are designed with topological properties such as high connectivity, node symmetry, hierarchical structure (allowing recursive construction) and small vertex degree in order to achieve low equipment cost, high and balanced throughput, easy expandability, low delay, scalable performance, and

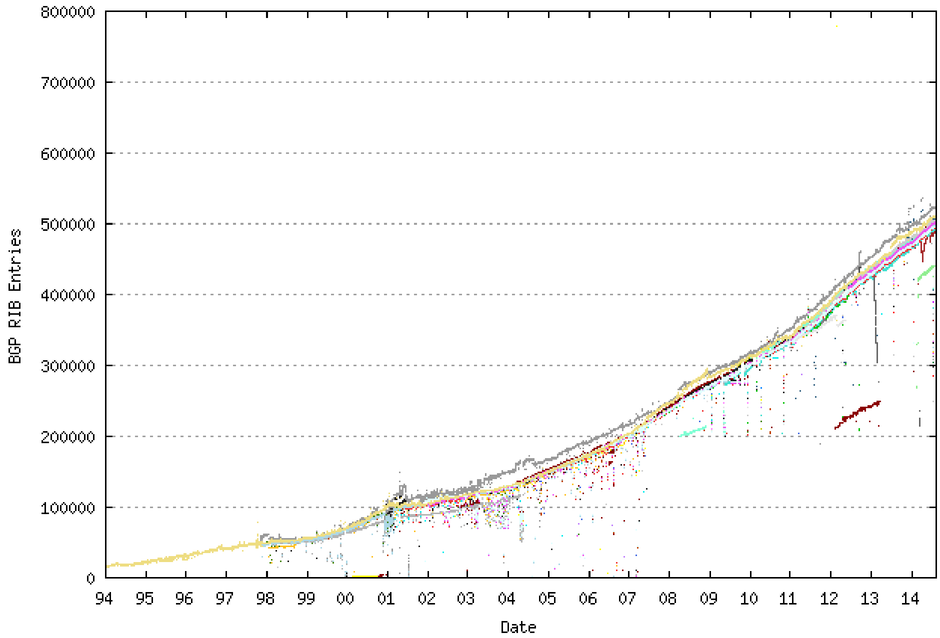


Figure 1.2: The growth of the RT between 1994 and 2014.
Source: http://bgp.potaroo.net.

robustness [6, 7]. Example of such topologies are Fat-Tree [8], BCube [9], Hypercube and Butterfly [6, 10], among others.

Modern DCs connect hundreds of thousands of computers and keep growing exponentially in number and size [11]. This information is recently supported by the Cisco Visual Networking Index 2013–2018 [12], where it is predicted 20.6 billion network devices in 2018 (from 12 billion in 2013) that will generate 1.6 zettabytes of traffic per year. This growth degrades the performance of the DC routing system and increases the cost of power and cooling systems (as part of the operation total cost) and its environment impacts [13]. As a consequence, the design of routing schemes in DCs that achieve good scalability plays a vital role in the system performance and energy efficiency [14].

In order to find sustainable solutions for the scalability problem in both the Internet and DCs topologies, the scientific community is trying to design routing schemes that have low time and space complexity, and that provide routes as close as possible to the shortest ones, by exploiting the statistical and structural properties of such networks. One of these research initiatives was the EULER project [15], an EU FP7 project in which this research work has been developed, whose aim was to investigate on new paradigms for distributed and dynamic routing schemes suitable for the Internet and its evolution.

The problem known as Compact Routing (CR) consists in designing routing schemes that achieve scalable RT size with respect to the number of network nodes n, i.e., the RT size grows sub-linearly (or lower) in n, with vertex labels (i.e., node identifiers) of

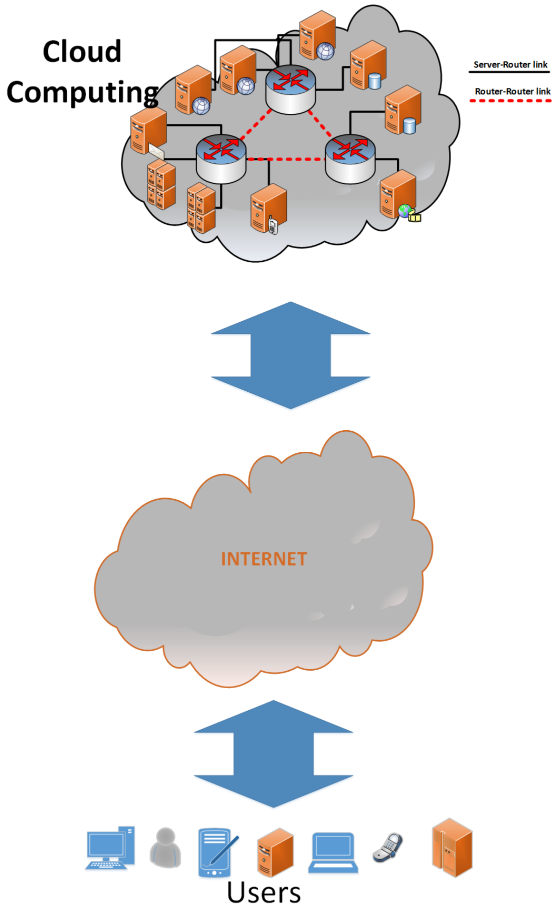


Figure 1.3: A Cloud Computing Infrastructure.

logarithmic size in n, and low stretch. The (multiplicative) stretch of the routing scheme is defined as the worst (highest) ratio between the length of the path produced by the routing scheme and the length of the shortest path (for the same source-destination pair), among all source-destination pairs (we consider that the length of the path is its number of hops). In other words, CR schemes can be seen as a trade-off between source routing (optimal RT size at the detriment of packet header size) and point-to-point routing (optimal packet header size at the detriment of RT size) by removing the requirement that packets are always routed on the shortest path.

As a potential solution to solve the Compact Routing Problem (CRP), *Geometric Routing (GR)* has been proved to be both simple and heuristically effective [16, 17]. Greedy Geometric Routing (GGR) schemes exploit the "geometric" dimension of graphs by assigning to each vertex (virtual) coordinates out of a metric space. These coordinates represent the relative position of the vertices as a function of their distances, such that they can be used to forward packets by selecting the closest neighbor (under some criteria in this space) to the destination requiring only local routing information. When the criteria to select the closest neighbor is the one whose distance to the destination is the minimum one among all the neighbors, the forwarding process is referred to as Greedy Forwarding (GF). The family of routing schemes that perform GF using coordinates of some metric space (i.e. Euclidean Hyperbolic, etc.) are called GGR schemes [18–20]. Without maintaining any routing state information per destination, the RT size in GGR is bounded by the maximum vertex degree with fast computation. GGR schemes that solve the CRP are called Compact Greedy Geometric Routing (C-GGR) schemes. However, the GGR schemes proposed so far experience one or more of the following problems: 1) they do not guarantee packet delivery, 2) they produce vertex labels of size linear (or higher) in n, 3) they can not be implemented in a distributed way, 4) they require a full knowledge of the network topology, or 5) they have unbounded stretch.

1.2 Objective of the thesis

This research work explores the use of the GGR schemes to solve the CRP in Internet-like networks and several families of DCs architectures. The main objective of this thesis is to design GGR schemes that have low time and space complexity, and that achieve routes as close as possible to the shortest ones (low stretch), suitable for general graphs and also specialized for these two types of networks.

1.3 Contributions

The main contribution of this dissertation is the proposal of a graph embedding together with the related mathematical tools, and the design of GGR schemes on top of this embedding that solve the CRP in both Internet-like networks and DCs. Specifically, the main contribution of this thesis are the following:

- An state-of-art of the scalability problem (CRP) in both Internet-like networks and DCs together with the most relevant works on GGR for these topologies.
- The mathematical tools to exploit the properties of metric spaces generated by algebraic groups and its application in GGR.
- Several novel and simple C-GGR schemes, with low time and space complexity and that achieve routes as close as possible to the shortest ones, for any finite connected graph, for Internet-like networks and for several families of DCs interconnection networks, by exploiting the statistical and structural properties of these networks.

1.4 Outline of the document

This document is organized into six chapters including this one, plus the bibliography at the end.

In Chapter 2 we present the fundamental concepts, terminology and notation on graph theory, metric spaces and group theory that are used in the rest of the document.

In Chapter 3 we present a survey of the most relevant works on GGR in several kinds of topologies.

In Chapter 4 we propose several GGR schemes, one GGR scheme for any finite connected graph and two specialized C-GGR schemes for Internet-like networks and for several families of DCs interconnection networks, and we analyze their performance theoretically.

In Chapter 5 we present the experimental evaluation through simulation of the two specialized C-GGR schemes, one in several scale-free networks and the other in several Cayley Graph (CG) interconnection networks, in terms of the RT size, the path length and others.

Finally, in Chapter 6 we summarize the main contributions of this work and propose possible directions for future work.

Chapter 2

Definitions, notation and terminology

In this chapter we present the fundamental concepts, terminology and notation that will be used in this document in the topics of graph theory, metric spaces and group theory. Although theorems, propositions, lemmas, and other, are presented without formal proofs, references to the source are provided to the reader.

2.1 Graph theory

In this section we establish definitions, terminology, notation, and important results in the area of graph theory. For more information on graph theory concepts and results, we refer the reader to the books [21–23].

2.1.1 Graphs and subgraphs

In graph theory, the term ***graph*** refers to the representation of a set of elements where some pairs of them are connected by links. Formally, a graph is defined as follows:

Definition 2.1. *A **graph** H consists of two sets V and E together with two maps $\theta_o : E \to V$ and $\theta_t : E \to V$.*

We write $H = (V,E)$. The set $V(H)$ is called ***the vertices, nodes or points*** and the set $E(H)$ is called ***the edges, lines or links***. If the context makes it clear, they can be simply denoted by V and E. The ***cardinality*** of V, that is the number of vertices, is denoted by $|V| = n$. The cardinality of E, that is the number of edges, is denoted by $|E| = m$. Given an edge $e \in E$, the vertex $\theta_o(e)$ is called the ***origin*** of e, and the vertex $\theta_t(e)$ is called the ***terminus*** of e. These two vertices are called the ***endpoints or endvertices*** of e. We say that $\bar{e}$ is the ***inverse*** of the edge e if $\theta_o(e) = \theta_t(\bar{e})$ and $\theta_t(e) = \theta_o(\bar{e})$. The pair of endpoints $x,y \in V$ associated with an edge $e \in E$ is simply denoted by $e = \{x,y\}$. When $x,y \in V$ are endpoints of an edge, we say that x and y are ***joined or connected by*** e, or that x and y are ***adjacent or neighbors***, or e is ***incident*** to both x and y. When $\theta_o(e) = \theta_t(e)$, the edge e is called a ***loop***. Note that this definition permits that two or more edges have the same endpoints x and y. In this case, we say that there are ***multiple edges*** connecting x and y. A graph is said to be ***simple*** if it contains neither loops nor multiple edges. A graph is said to be ***finite*** if both V and E are finite sets.

A graph H is called ***undirected*** if the elements of the set E have no orientation, i.e., each $e \in E$ is an unordered pair of vertices ($e = \bar{e}$). On the other hand, if the elements of the set E have orientation, i.e., E is composed by ordered elements, then H is said to be ***directed***. Let $H = (V,A)$ be a directed graph. Each ordered element $(x,y) \in A$ is called an ***arc*** or ***oriented edge***. Vertex y is called the ***head*** (or ***successor* of** x) and vertex x is called the ***tail*** (or ***predecessor*** of y). In addition, we say that the arc $e \in A$ ***comes out*** of the vertex x and that it ***enters into*** vertex y. Finally, associated to any graph H, there exists an undirected graph with the same set of vertices and two vertices are connected by an edge if, and only if, they are different and joined by at least one (oriented) edge in H. This graph is called the ***the underlying graph*** of H.

Definition 2.2. *Let $H = (V,E)$ and $K = (U,F)$ be two graphs. A* ***subgraph*** *K of H, denoted by $K \subseteq H$ or simply K_H, is a graph of the form $K = (U,F)$, where $U \subseteq V$ and $F \subseteq E$, such that any edge of F has its endpoints in U. If $V = U$, then the subgraph K_H is called a* ***spanning subgraph*** *of H.*

A subgraph K_H is said to be ***induced*** if $U \subseteq V$ and the endpoints of any edges are in both V and U. In other words, the graph K is an ***induced subgraph*** of H if it has all the edges that appear in H with the same vertex set. In general, since $U \subseteq V$, we say that K is the graph ***induced by*** the vertex set U. A ***supergraph*** of a graph H is a graph of which H arises as a subgraph. Finally, if a graph H does not contain K as an induced subgraph, then it is called ***K-free***. As example, those graphs that do not have a triangle graph as an induced subgraph are triangle-free graphs.

2.1.2 Graph isomorphism problem

Given two graphs H and K, an important question in graph theory is *Do H and K have an identical graph structure?* or simply *Are H and K the same graph?*. Formally, the terms *"same"* or *"identical"* graph are defined as follows:

Definition 2.3. *Let $H = (V,E)$ and $K = (U,F)$ be two graphs. An* ***isomorphism*** *of graphs H and K consist of two bijections: $\phi_0 : V \to U$ and $\phi_1 : E \to F$, such that for each $e \in E$ and $x,y \in V$, the edge $\phi_1(e)$ has for endpoints $\phi_0(x)$ and $\phi_0(y)$ in K, if and only if the edge* *e has x and y as endpoints in H. The graphs H and K are said to be* ***isomorphic*** *(written $H \cong K$) if such bijections exist.*

Note that these mappings preserve the incidence relation of the edges to the vertices. In fact, both graphs have the same graph structure and properties, but they differ in the ***names*** or ***labels*** of vertices and edges. Since it is the structural properties of the graph what we primarily are interested in, unlabeled graphs can be considered as a representative of an equivalence class of isomorphic graphs.

We conclude this subsection by introducing some special families of graphs. A graph that is simple and where any two vertices are connected by an edge is called a ***complete graph***. Up to isomorphism, there is just one complete graph on n vertices and it is denoted by K_n. If a graph can be drawn in a plane with the condition that two edges can not cross each other, except in their common endpoints, then it is called a ***planar graph***. A graph is called ***bipartite*** if its set of vertices can be split into two disjoint subsets of vertices $V = X \bigcup Y$ such that each edge has an endpoint in each subset. Such partition (X,Y) is called a ***bipartition*** of the graph. If the bipartite graph with bipartition (X,Y) is simple and each vertex of X is connected to each vertex of Y, then it is called a ***complete bipartite graph***. If $|X| = n_x$ and $|Y| = n_y$, then this graph is denoted by K_{n_x,n_y}. Finally, a graph is called ***sparse*** if m is much less than n^2, while it is called ***dense*** if m is closed to n^2.

2.1.3 Paths, cycles and vertex degree

For any pair $u,v \in V$, we define a ***walk*** as a sequence of the form $\{x_0,e_1,x_1,e_2,\ldots,e_k,x_k\}$, where k is an integer ≥ 0, $x_0 = u$, $x_k = v$, $x_i \in V$ and $e_i \in E$ such that for $i = 0,\ldots,k-1$, x_i and x_{i+1} are the endpoints of e_{i+1}. The vertex x_0 is called the ***start or source*** vertex, x_k the ***end or destination*** vertex, and both together are called the ***ends*** of the walk. The value of the integer k is the ***length*** of the walk. We also can define a walk by the sequence $\{x_o,x_1,\ldots,x_k\}$ or $\{e_1,e_2,\ldots,e_k\}$. If the edges of the walk are all distinct, then it is called a ***trail***. A walk is called a ***path*** if all its vertices are different, and it will be denoted by $p(u,v)$, where $u,v \in V$ is any source-destination pair of vertices. Note that a path itself is a trail. A ***cycle*** or ***closed walk*** is a walk of the form $\{x_o,e_1,x_1,e_2,\ldots,e_k,x_0\}$. We say that a walk is a ***simple cycle***, ***circuit*** or ***polygon*** if it is a closed walk and all its vertices are different, excepts its ends. A graph H is said to be ***connected*** if there is a path for any two vertices. The ***distance*** $d_H(x,y)$ in H between two vertices $x,y \in V$ is the length of a path that has the shortest length connecting them. The largest distance between any two vertices in H is called the ***diameter*** of H, and it is denoted by $D(H)$, or simply D if the context makes it clear.

The ***degree*** of a vertex x in an undirected graph is the number of incident edges to

x. This value is an integer and is denoted by $d(x)$. The ***minimum vertex degree*** of the graph H is the smallest degree among all its vertices and is denoted by δ_H. Likewise, the ***maximum vertex degree*** of the graph H is the largest degree among all its vertices and is denoted by Δ_H. A graph is called ***locally finite*** if $d(x)$ is finite for any vertex x in H. When a graph H has the same vertex degree k in all its vertices, then this graph is called ***k-regular graph***. In directed graphs, the ***in-degree*** (respectively ***out-degree***) of a vertex x is the number of arcs entering into (exiting from) x. In general, the concepts of vertex degree, walk, trail, cycle, circuit, path, etc., can be transposed directly from the undirected to the directed case by replacing the word *edge* by the word *arc*. However, there is an alternative approach which is to apply the concepts defined for undirected graphs to directed graphs by means of its underlying graph, e.g., we say that a directed graph is connected if its underlying graph is connected. In this document, we will follow the second approach.

Finally, the following proposition shows the relation between the number of edges and the vertex degrees of a graph.

Proposition 2.1 (Proposition 1.1 of [21]). *In a graph $H = (V,E)$, we have:*

$$\sum_{x\in V} d(x) = 2m.$$

Note that as a direct consequence of the above proposition, we can obtain

$$n \cdot \delta_H(H) \leq 2m \leq n \cdot \Delta_H(H).$$

2.1.4 Trees

One family of graphs that plays an important role in many applications of graph theory is the ***trees***. Formally, it can be defined as follows:

Definition 2.4. *A **tree** is an undirected and connected graph $T = (V,E)$ that has no cycles (acyclic).*

Theorem 2.1 (Theorem 2.1 of [21]). *The following conditions for a graph T are equivalent:*

1. *If T is finite, then $n = m + 1$ (Proposition 2.1 of [21]).*
2. *Any two vertices in T can be connected by a unique path. (Proposition 2.3 of [21]).*
3. *For every edge $e \in E$, removing e from T disconnects the graph. (Proposition 2.5 of [21]).*

A vertex $x \in V(T)$ is called ***internal*** if $d(x) \geq 2$. If $d(x) = 1$, then the vertex is called ***terminal*** or ***leaf***. A ***polytree*** or ***oriented tree*** is a directed acyclic graph whose underlying undirected graph is a tree, i.e., by replacing its arcs with edges, the resulting undirected graph is both connected and acyclic. A tree is said to be ***rooted*** if some arbitrary vertex $r \in V(T)$ is selected as the ***root*** of the tree. The ***tree order*** associated with T and r is a partial ordering on $V(T)$ such that for any $x, y \in V(T)$, $x \leq y$ if the unique path from the root to y passes through x. In a rooted tree, the ***parent*** of a vertex x is its adjacent vertex on the path to the root. Note that every vertex has a unique parent, except the root which does not. A ***child*** of a vertex x is a vertex of which x is the parent. A tree for which any vertex has at most k children is called ***k-ary tree***. A ***spanning tree*** T of H, represented by T_H, is a spanning subgraph of H that is a tree. The ***depth*** of a node is the length of the path from the root to the node, and the ***tree depth***, denote by $td(T)$ is the length of the path from the root to the deepest node. Finally, one of the more useful results of spanning trees is the following:

Proposition 2.2 (Proposition 2.6 of [21])**.** *Every connected graph H has (at least) one spanning tree.*

2.1.5 Representation of graphs

There are several ways to represent graphs. Let $H = (V, E)$ be a graph with vertex set $V = \{x_1, x_2, \ldots, x_n\}$. The ***adjacency matrix*** is the square matrix $M(H) = [m_{i,j}]$ of order n, where $m_{i,j}$ is the number of edges having x_i and x_j as endpoints in H. This representation requires a memory space of the order $O(n^2)$ and because the time to process the graph is at least the time to read its data, then the time complexity of any algorithm over a graph with this representation is at least $O(n^2)$. Another way of representing a graph is the ***adjacency list***, which consist of an array A of n lists, one for each vertex in V. For each $x \in V$, the adjacency list $A[x]$ contains all the vertices $y \in V$ such that there is an edge $\{x, y\} \in E$. In this representation, the space complexity is bounded by $O(n + m)$ and its processing time is linear.

The selection of such representation is very important because it may have a direct impact on the efficiency of the algorithms in terms of complexity (Chapter 1, [24]). For example, in adjacency list, listing the neighbors of each vertex is performed efficiently in time proportional to the degree of the vertex. However, the same operation in an adjacency matrix takes time proportional to the number of vertices in the graph, which may be significantly higher than the degree. In contrast, the adjacency matrix allows verifying if two vertices are adjacent in constant time, but in adjacency list this operation is slower [21, 25]. In terms of space complexity, the adjacent list is a compact way to represent

sparse graphs. However, if the graph is ***dense***, the adjacency matrix is the best option to run algorithms that require some specific operations. For example, determining whether there is an edge between two vertices, which is used in all-pairs shortest path algorithm, can be performed in constant time on graphs with adjacency matrix representation [25].

2.2 Metric spaces and embeddings

Given a set of elements X, the concept of ***distance*** is a numerical way to describe how far apart the elements are. Roughly speaking, if there is a ***distance function*** or ***metric***, which behaves according to a set of rules, such that the distances among all the members of the set are defined, then the function together with the set are called a ***metric space***. The metric on the set induces geometric properties that can be useful to solve problems on that set.

If a problem is defined over a ”difficult” metric, it could be reduced to a problem over an ”easier” metric. Here, the concept of embedding between metric space becomes important. For further information about metric spaces, we refer the reader to the book [26] and for graph embedding and topological graph theory to [27] .

Definition 2.5. *A* ***Metric Space*** (X, d_X) *consists of a set* X *and a distance function* $d_X : X \times X \to \mathbb{R}$ *such that for any* $x, y, z \in X$*:*

1. $d(x,y) \geq 0$ *(non-negative).*
2. $d(x,y) = 0$ *if and only if* $x = y$ *(identity of indiscernible).*
3. $d(x,y) = d(y,x)$ *(symmetry).*
4. $d(x,y) + d(y,z) \geq d(x,z)$ *(triangle inequality).*

An interesting class of metric spaces is the one obtained by defining the vertices of a graph as the set of elements, and defining the distance as the number of edges of a shortest path connecting them. Formally, it can be defined as follows:

Definition 2.6. *Let* $H(V,E)$ *be a graph. The graph* H *is a* ***metric graph*** *by assigning to each edge* $e \in E(H)$ *a metric of length 1: define* $X = V(H)$ *and for any pair of vertices* $x, y \in V(H)$*, define* $d_H(x,y)$ *as the length (number of edges) of a shortest path connecting them.*

In metric spaces, a ***geodesic*** is a curve which is everywhere locally a distance minimizer, i.e., the shortest curve connecting two points. A metric space where every pair of points can be connected by a geodesic is called a ***geodesic space***. In metric graphs, a geodesic between two vertices correspond to a shortest path between them. Therefore, any connected metric graph is itself a geodesic metric space.

Given two sets X and Y, X is said to be embedded in Y if there exists some injective and structure-preserving map $f : X \to Y$. Here, the meaning of structure-preserving depends on the mathematical structure that X and Y are. Therefore, an embedding between metric spaces can be defined as follows:

Definition 2.7. *A* ***metric embedding*** *(with distortion $c > 0$) from a* ***metric space*** *(X, d_X) (the source metric) to another (Y, d_Y) (the target metric) is an injective map $\phi : X \to Y$ such that:*

$$\forall x, y \in X,\ l \cdot d_X(x,y) \leq d_Y(\phi(x), \phi(y)) \leq c \cdot l \cdot d_X(x,y), \text{ for some constant } l > 0.$$

Definition 2.8. *A* ***contractive metric embedding*** *with distortion c from a metric space (X, d_X) to another (Y, d_Y) is an injective map $\phi : X \to Y$ such that:*

$$\forall x, y \in X, 1 \leq \frac{d_X(x,y)}{d_Y(\phi(x), \phi(y))} \leq c.$$

Definition 2.9. *An* ***expansive metric embedding*** *with distortion c from a metric space (X, d_X) to another (Y, d_Y) is an injective map $\phi : X \to Y$ such that:*

$$\forall x, y \in X, 1 \leq \frac{d_Y(\phi(x), \phi(y))}{d_X(x,y)} \leq c.$$

More generally, a ***metric embedding*** is a trade-off between the dimension and the fidelity (or distortion) of the embedding. Note that if $c = 1$, then the Definitions 2.9 and Definition 2.8 coincide with an exact preservation of all distances. This kind of embedding is called ***metric embedding with no distortion***, ***distance-preserving embedding***, or ***isometric embedding***:

Definition 2.10. *An* ***isometry*** *from one* ***metric space*** *(X, d_x) to another (Y, d_Y) is an injective map $\phi : X \to Y$ such that:*

$$\forall x, y \in X,\ d_X(x,y) = d_Y(\phi(x), \phi(y)).$$

If the function ϕ exists then (X, d_x) and (Y, d_Y) are said to be ***isometric****.*

Definition 2.11. *An* ***isometric embedding*** *from one* ***metric space*** *(X, d) to another (Y, d_Y) is a map $\phi : X \to Y$ such that ϕ is an isometry.*

In general, the problem of finding an isometric embedding between metric spaces is proved to be *NP-Hard* [28]. However, the value of the distances themselves is not essential in nearest-neighbor searching problems in contexts such as visualization, compression, routing, and clustering [29, 30]. In these cases, what is important is to preserve the relative order between pairs of distances (which pairs are larger or smaller), and not necessarily the values of the distances themselves, which is obtained by an ***ordinal embedding***.

Definition 2.12. *An* ***ordinal embedding*** *(with no relaxation) from a metric space* (X,d) *to another* (Y,d_Y) *is a mapping* $\phi : X \to Y$ *such that every comparison between pairs of distances has the same outcome:* $\forall w,x,y,z \in X, d_X(w,x) \leq d_X(y,z)$ *if, and only if,* $d_Y(\phi(w),\phi(x)) \leq d_Y(\phi(y),\phi(z))$.

Note that the map ϕ induces a ***monotone function*** $d(p,q) \to d'(\phi(p),\phi(q))$, i.e., the map preserves the given order between distances. Because the order of the distances captures enough information, then it is only necessary an embedding that ensures a monotone mapping of the distances into the target metric space. However, finding this kind of embedding still is *NP-hard* and the target metric space requires $\Omega(|X|)$ dimensions [29]. As an alternative, ***greedy embeddings*** [31], which is a kind of ordinal embedding, have been used to find low-dimensional target metric spaces in polynomial time [32, 33]. This embedding requires order preservation only among pairs of points of the form (x,z) and (y,z), where x and y must be connected or share an adjacent point in the source metric space. Formally, this embedding is defined as follows:

Definition 2.13. *A* ***greedy embedding*** *from a metric space* (X,d_X) *to another* (Y,d_Y) *is an injective map* $\phi : X \to Y$ *such that for every pair of elements* $x,z \in X$ *there exists an element* y *adjacent to* x *such that* $d_Y(\phi(y),\phi(z)) < d_Y(\phi(x),\phi(z))$.

It is proved in [34] that finding a monotone map for a given order on the pairwise distances between n points is not equivalent to finding a distance-preserving greedy embedding. On the other hand, a monotone map for a metric space is also a greedy embedding for this metric. When the source metric space is a finite and connected graph, the existence of a greedy embedding is a monotonic increasing graph property, i.e., adding an edge does not prevent a graph of greedy embedability [31]. The following lemma is a direct consequence of this property.

Lemma 2.1. *Let* (X,d_X) *be a metric space and* H *be a connected metric graph. If* K *is a spanning subgraph of* H*, then every greedy embedding* $\phi : V(K) \to X$ *is also a greedy embedding from* $V(H)$ *to* X.

Proof. Let $\phi : V(K) \to X$ be a greedy embedding. For any two vertices $x,y \in V(K)$, there exists a path $x,s_0,s_1,s_2,\ldots,s_k,y$ such that $d_X(\phi(s_i),\phi(y))$ is monotonically decreasing as i moves from 0 to k. Because $V(K) = V(H)$ and $E(K) \subseteq E(H)$ by Definition 2.2, then all the vertices and edges in the path $x,s_0,s_1,s_2,\ldots,s_k,y$ are also in H. Thus, x has a neighbor s_0 in $V(H)$ such that $d_X(\phi(s_0),\phi(y)) < d_X(\phi(x),\phi(y))$. □

2.3 Group theory

In the area of abstract algebra, group theory studies the mathematical structures known as groups. Besides studying groups as algebraic structures (the concept of group comes from the 19th century [35]), they can be studied from the point of view of geometry and theory of regular languages. In this section we present the main concepts and results about groups from these different points of view. For more information about combinatorial group theory, we refer the reader to [36, 37], about geometric group theory to [26, 38, 39], about word processing in groups to [40] and about computational group theory to [41].

2.3.1 Groups as algebraic objects

A group is defined as follows:

Definition 2.14. *An **(algebraic) group** $(G,\cdot)$ is a non-empty set G together with a binary operation $\cdot$ defined in G for which the following four conditions (also called the group axioms) are satisfied:*

1. *For any pair of elements $g,h \in G$, there exists a **uniquely determined element** $k \in G$ such that $g \cdot h = k$.*

2. *The operation $\cdot$ is **associative**, i.e., for any elements $g,h,k \in G$ we have $(g\cdot h)\cdot k = g\cdot(h\cdot k)$.*

3. *For any element $g \in G$, there exists an element $e \in G$, called the **identity element** of the group, for which $g \cdot e = e \cdot g = g$.*

4. *If g is any element in G, then there exists an element g^{-1}, called the **inverse** of g, such that $g\cdot g^{-1} = g^{-1}\cdot g = e$.*

A group is called an ***abelian group*** if the operation $\cdot$ is commutative, i.e.:

5. $g\cdot h = h\cdot g$, for all $g,h \in G$

In this document we will refer to the group $(G,\cdot)$ simply by G, and the operation symbol $\cdot$ will be omitted and write gh instead of $g\cdot h$. We also assume that the groups are ***multiplicative***, i.e., they have the multiplication as its group operation. Note that the following can be easily deduced from Definition 2.14: for all $g,h,k \in G$, $(gh)^{-1} = h^{-1}g^{-1}$; if $gh = gk$ then $h = k$, and if $hg = kg$ then $h = k$. If $n \in \mathbb{N}$ and $g \in G$, then g^n is defined inductively by $g^1 = g$ and $g^{n+1} = gg^n$ for $n \geq 1$. We also define $g^0 = e$. The ***order of an element*** $g \in G$, denoted by $|g|$, is the least $n > 0$ such that $g^n = e$, if such n

exists. If it does not exist, then we say that g has ***infinite order***, denoted by $|g| = \infty$, and the elements g^i are all different for any $i \in \mathbb{N}$. Finally, the ***order or rank of the group*** G, denoted by $|G|$, is its cardinality, i.e., the number of elements in the set G.

Definition 2.15. *Let $X = \{x_1, \dots, x_n\}$ be a set of elements in a group such that $X \subseteq G$. The group G is said to be* ***generated by*** *X if every element of G can be expressed as a product of elements from X and their inverses. The set X is called the* ***generating elements*** *of G.*

In computational group theory, the field of group theory that deals with the design and analysis of algorithms and data structures to compute information about groups, there are three methods commonly used to describe groups: ***group of permutations*** of a finite set, ***groups of matrices*** over a ring and groups defined by a ***finite presentation***. In this document, we will use the finite presentation of groups (see below in Definition 2.24) for two main reasons. Firstly, because they often provide the most compact and precise definition of the group, and secondly, because the algorithms used in this work require the finite presentation for computational purposes. However, there exist algorithms that allow transformations among these different representations (see [42] and both Chapters 3 and 5 of [41]).

2.3.2 Groups as algebraic objects: group elements and words

Because the notion of group is an abstract concept, we will introduce some ideas to represent a formal product of group elements by using symbols, in the same way that polynomial variables are used to represent algebraic combination of elements in an integer domain.

Definition 2.16. *Let $S = \{s_1, \dots, s_n\}$ be a set and let $S^{-1} = \{s_1^{-1}, \dots, s_n^{-1}\}$ be a set, disjoint from S, for which there is a bijection (map one-to-one) $S \to S^{-1}$ and denoted by $s_i \mapsto s_i^{-1}$, for all i. A* ***word*** *is a sequence $w = s_1 s_2 \dots$, where $s_i \in S \cup S^{-1}$, for all i.*

Definition 2.17. *The* ***length*** *of a word $w = s_1 s_2 \dots s_n$, denoted by $l_s(w)$, is the value of the integer n. The* ***empty word*** *is the word of length 0 and it is denoted by 1.*

It is customary to abbreviate a block of n consecutive symbols s_i and s_i^{-1} by s_i^n and s_i^{-n}, respectively.

Definition 2.18. *If $w = s_1 s_2 \dots s_n$ is a word, then its* ***inverse*** *is the word $w^{-1} = s_n^{-1} \dots s_2^{-1} s_1^{-1}$. The inverse of the empty word is itself.*

Definition 2.19. *A* ***subword*** *of $w = s_1 s_2 \dots s_n$ is either an empty word or a word of the form $v = s_i \dots s_j$, where $1 \le i \le j \le n$.*

Definition 2.20. *A word w is* ***reduced*** *if either w is empty or $w = s_1 s_2 \dots s_n$ does not contain any subword of the form $s_i s_i^{-1}$.*

Given any two words in the set $s_i \in S \cup S^{-1}$, we define the operation of ***juxtaposed product*** of elements as follows: if $w = s_1 s_2 \dots s_n$ and $v = s'_1 s'_2 \dots s'_m$ are two words, then their juxtaposed product is $wv = s_1 s_2 \dots s_n s'_1 s'_2 \dots s'_m$. Clearly $(wv)^{-1} = v^{-1} w^{-1}$ and $l_s(wv) = l_s(w) + l_s(v)$.

From the definitions above, one can define a map of symbols S into a group G.

Definition 2.21. *Let G be a group, S a set of symbols and S^* the set of words over the set S. By interpreting the juxtaposed product of words as an associative multiplication on G, we define a surjective group homomorphism $\pi : S^* \to G$. Because π is surjective, then $\pi(S)$ generates G as a group and the elements of S are called the* ***generating symbols*** *for G (under π).*

Note that under this map, $\pi(1) = e$, i.e., the empty word 1 defines the identity element e of G. In the same way, the generating elements of G can be obtained under the map $\pi(S)$. For the rest of the document, if the context makes it clear, both the generating symbols and the generating elements may be referred to as the ***generators*** of G. In addition, both the empty word and the identity of the group will be simply denoted by e. Note also that, it is possible to see that the juxtaposed product of two words w and v does not define a product on the set of the all reduced words on S because wv does not need to be reduced even if w and v are. By defining a new juxtaposed product of reduced words as the reduced word obtained from wv after cancellation, it is possible to construct the ***free group*** with basis S.

Definition 2.22. *If S is a set of generators for a group G and no reduced word w represents the identity element of G, then it is called the* ***free group with basis*** *S.*

The ***rank of the free group*** with basis S is the number of elements in S. We denote a free group of rank n by $F_n(S)$ or simply $F(S)$, if the set of generators and its cardinality is sufficiently clear from context. Two important results about free groups are related to the fact that 1) given a set S, it is always possible to define a free group on it and, 2) given two sets S and T where $|S| = |T| = n$, then the free groups $F_n(S)$ and $F_n(T)$ are isomorphic.

Theorem 2.2 (Theorem 11.1 of [37])**.** *Given a set S, there exists a free group $F(S)$ with basis S.*

Theorem 2.3 (Theorem 11.4 of [37])**.** *Let $F(S)$ and $G(T)$ be free groups with bases S and T, respectively. Then $F \cong G$ (isomorphic) if and only if $|S| = |T|$.*

Any word w that defines the identity element e in the group is called a ***relator***. In addition, the equation $w = v$ is called a ***relation*** if the word wv^{-1} is a relator, or equivalently if $\pi(w)$ and $\pi(v)$ define the same element in the group. Note that in any group G with generator set S, the empty word and all the words of the form $w = s_i s_i^{-1}$, for $s_i \in S \cup S^{-1}$, are always relators and they are called ***trivial relators***.

Definition 2.23. *Let $p, q, r, \ldots$ be relators of a group G. A word w is **derivable** from $p, q, r, \ldots$, if the following operations, applied a finite number of times, change w into the empty word:*

1. *Insertion of one of the words $p, p^{-1}, q, q^{-1}, r, r^{-1} \ldots$ or one of the trivial relators between any two consecutive subwords of w, or before w, or after w.*
2. *Deletion of one of the words $p, p^{-1}, q, q^{-1}, r, r^{-1} \ldots$ or one of the trivial relators, if it is a subword in w.*

It is clear that if w is derivable from the relators $p, q, r, \ldots$, then w is itself a relator because the application of the operations 1) and 2) on w does not change the element of the group defined by w, and because the empty word is obtained, then w must define the identity element of G. If every relator is derivable from the relators $p, q, r, \ldots$, then the set $p, q, r, \ldots$ is called ***set of defining relators*** or a ***complete set of relators*** for the group G on the generators S. We denote that set by R.

Definition 2.24. *Let G be a group with generator set S. The group G has a **presentation** $\langle S|R\rangle$ if and only if G can be described by a set S of generators and a set R of defining relations for these generators, i.e., a set of relations from which all others can be derived.*

Theorem 2.4 (Theorem 3.15 of [39])**.** *If G has presentation $\langle S|R\rangle$, $G \cong \langle S|R\rangle$.*

The group presentation is called ***finitely generated*** (***finitely related***) if the number of generators (relators) in it is finite. If a group presentation is finitely generated and finitely related, then the group is said to be ***finitely presented***.

2.3.3 Groups as geometric objects: the graph of the group

As we have seen, groups are abstract objects represented by a set of elements and a binary operation that satisfy a list of conditions (see Definition 2.14). However, groups arise in other contexts. Groups can also be studied by means of their actions on mathematical objects such as graphs, sets of numbers, regular polygons, etc. This section presents the geometric realization of a group: the so-called Cayley Graph.

The representation of the abstract concept of group as geometrical object such as a graph, requires the introduction of the concept known as group action on sets.

Definition 2.25. *Let $(G,*)$ and $(K,\cdot)$ be two algebraic groups. A **group homomorphism** from $(G,*)$ to $(K,\cdot)$ is a function $h : G \to K$ such that for all u and v in G it holds that $h(u*v) = h(u)\cdot h(v)$*

Definition 2.26. *A **permutation** ϕ on the set $X = \{1,...,n\}$ is a bijective map from X to X, and it is denoted by the images of the elements $(\phi(1),\ldots,\phi(n))$.*

Definition 2.27. *Let $Sym(X)$ be the group of all permutations on the set X. An **action** of a group G on a mathematical object X, denoted by $G \curvearrowright X$, is a group homomorphism from G to $Sym(X)$. In other words, it is a map from $G\times X \to X$ such that:*

1. *$e\cdot x = x$, $\forall x \in X$*
2. *$(gh)\cdot x = g\cdot(hx)$, $\forall x \in X$ and $\forall g,h \in G$*

The associated homomorphism of the group action $G \curvearrowright X$ is called a ***representation*** of G. When the map is injective, then the representation is ***faithful***. The group $G = Sym(X)$ is called the ***symmetric group of*** X being X a set. In other contexts, $Sym(X)$ receives different names. For example, if X is a group, then G is the set of all automorphisms of G and is denoted by $Aut(G)$. Given a metric space (Y,d), the group of all asymmetries from (Y,d) to itself is denoted by $Isom(Y)$. In general, the symmetric group denotes all bijections from X to X that preserve the mathematical structure of X.

An important result in group theory is the ***Cayley's Theorem*** where the notions of abstract group G and group of permutation are proved to be equivalents.

Theorem 2.5 (Theorem 1.5 of [39])**.** *Every group can be faithfully represented as a group of permutations.*

In other words, the Cayley's Theorem says that every group G is isomorphic to a subgroup of the symmetric group acting on G. One of its applications is the construction of a representation of G as a group of permutations of itself. However, it is not its unique application. An improvement of the Cayley's theorem allows studying an abstract group by means of its action on geometric objects such as graphs. Before presenting the extended version of the Cayley's Theorem, let us introduce the concept of symmetries of a graph:

Definition 2.28. *Let $H = (V,E)$ be a graph. A **symmetry** of H is a bijection β taking vertices to vertices and edges to edges such that if $e = \{x,y\}$, for $e \in E$ and $x,y \in V$, then $\beta(e) = \{\beta(x),\beta(y)\}$. The symmetric group of H is the collection of all its symmetries and is denoted by $Sym(H)$.*

Theorem 2.6 (Theorem 1.42 of [39])**.** *Every finitely presented group can be faithfully represented as a symmetric group of a connected, directed and locally finite graph.*

Definition 2.29. *Let G be a group with generating set S and let $\Gamma_{G,S}$ be a connected, directed and locally finite graph. The graph $\Gamma_{G,S}$ is called the **Cayley Graph (CG)** of G if it is the graph with vertex set $V(\Gamma_{G,S}) = \{g \mid g \in G\}$ and edge set $E(\Gamma_{G,S}) = \{(g, gs) \mid s \in S, g \in G\}$.*

For a given group G with generating set S, the proof of Theorem 2.6 shows that there exists an inclusion mapping from G to $Sym(\Gamma_{G,S})$. In the proof, it is constructed both the graph $\Gamma_{G,S}$ and the action of G on $\Gamma_{G,S}$ by multiplication on the left as follows: the element $g \in G$ defines a map $\phi_g : h \to gh$ that maps a vertex $h \in \Gamma_{G,S}$ to the vertex $gh \in \Gamma_{G,S}$ and the endpoints of the adjacent vertices of $h \in \Gamma_{G,S}$ go to the endpoints of the adjacent edges of gh preserving the direction and labeling on those edges. The graph $\Gamma_{G,S}$ is directed but it also can be considered undirected if we take an inverse-closed generating set, i.e., if $s \in S$ then $s^{-1} \in S$. Since every vertex has an edge for each generator and its inverse, then it is a $2|S|$-regular graph. If the graph $\Gamma_{G,S}$ has not auto-loops in every vertex, then the identity element of the group G does not belong to S. In addition, if the graph has no multiple edges, then $s_i \neq s_j, \forall s_i, s_j \in S$.

Note that the action of a group G on a graph does not say anything specific about the group G itself. The following theorem indicates that all finitely generated groups can be seen as label and orientation preserving symmetries of a locally finite and directed graph.

Theorem 2.7 (Theorem 1.51 of [39])**.** *Let $\Gamma_{G,S}$ be the CG of a group G with generating set S. Consider $\Gamma_{G,S}$ as the directed graph with edge labels corresponding to the generating set S. Then $Sym(\Gamma_{G,S}) \cong G$.*

Thus, for any finite presentation of a group in terms of generators and defining relations, there exists an associated CG by Theorem 2.7, i.e., the geometry and structure of the $\Gamma_{G,S}$ is directly related to a group presentation and specifically to its generator set. Note that the CG is itself a graph according Definition 2.1, and it is a connected metric space by using Definition 2.6. In addition, it is also possible to define a metric in the group by using its algebraic structure rather than its geometric one as follows:

Definition 2.30. *The **length** of g, identified by $l_s(g)$, is the length of a shortest word in the Free Group $F(S)$ representing g, i.e., $l_s(g) = min\{l_s(w) \mid w \in F(S), \pi(w) = g\}$.*

Definition 2.31. *Let G be a group with generating set S. The corresponding **word-metric** (i.e., distance function) d_s is the metric on G satisfying $d(e,s) = d(e,s^{-1}) = 1$ for all $s \in S$, and $d(g,h) = min\{l_s(w) \mid w \in F(S), \pi(w) = g^{-1}h\}$, for all $g, h \in G$.*

The word-metric on G measures how efficient the difference $g^{-1}h$ can be expressed as a word in the generating set for that group. From the geometric point of view, the word-metric on a group $G = \langle S|R\rangle$ is a way to determine the length of a shortest path between any two elements of G in $\Gamma_{G,S}$. Therefore, the word-metric of a group $G = \langle S|R \rangle$ corresponds to the graph metric induced on its graph $\Gamma_{G,S}$. Finally, it is important to note that graph structure of the CG of a group depends on the choice of the generating set, i.e., different group presentations of the same group would result into completely different CG (from the point of viewpoint of graph theory).

2.3.4 Groups as geometric objects: words and paths

As we have seen, a group presentation $G = \langle S|R \rangle$ defines a unique group (up to isomorphism). However, it is difficult to derive characteristics of the group from its presentation, e.g. whether it is abelian, finite, and other. A way to determine whether a group is abelian is by verifying that for all $s_i, s_j \in S$, $s_i s_j s_i^{-1} s_j^{-1} = e$. In other words, if there was a procedure to decide whether or not a word defines the identity element in G, then it could be decided whether G is abelian. The problem of deciding whether a word in G represents the identity element (or, equivalently, whether two words represent the same element in G) is one of the three fundamental problems in Group Theory formulated by Max Dehn [43]. Formally, the so-called ***word problem*** can be formulated as follows:

Definition 2.32. *Let $G = \langle S|R \rangle$ be a group presentation. For an arbitrary word w in the set of generators, decide in a finite number of steps whether w defines the identity element of G or not.*

Although the word problem has been solved for many groups, in general the word problem for finitely presented groups is not solvable, that is, given two words in the group generator set, it might be that there was no algorithm able to decide whether the words represent the same element in the group [44]. A related problem is how to decide whether a given word w in the set of generators has minimum length. Equivalently, this can be formulated as follows:

Definition 2.33. *Let $G = \langle S|R \rangle$ be a group presentation. For an arbitrary word w in the generators, decide in a finite number of steps whether there exists a word v such that w and v define the same element in G and $l_s(v) < l_s(w)$.*

Definition 2.34. *Let X be a set. A **total ordering** is a binary relation (denoted by $\leq$) on a set X which is transitive, antisymmetric, and total. A set paired with a total word problem is called a **totally ordered** set. If X is totally ordered under $\leq$, then the following statements hold for all a, b and c in X:*

1. *If $a \leq b$ and $b \leq a$ then $a = b$ (antisymmetry).*
2. *If $a \leq b$ and $b \leq c$ then $a \leq c$ (transitivity).*

3. *$a \leq b$ or $b \leq a$ (totality).*

Definition 2.35. *Let X be a set. A* ***well-ordering*** *on S is a total order on X with the property that every non-empty subset of X has a least element in this ordering. A set paired with a well-ordering is called a* ***well-ordered*** *set.*

The above problem is called the ***minimum-length word problem (MWP)*** and it has been proved to be *NP-hard* [45]. Assume that the set S^* is given by a kind of ***normal or canonical*** form and it is a well-ordered set. Thus, deciding whether two elements in S^* are equivalent can be solved by testing their canonical forms for equality. In other words, if the map $\pi : S^* \to G$ (see Definition 2.21) is a bijection, then we can solve the MWP by reducing w to its canonical form w', and verifying that $w = w'$. Note that the last step can be done only if the word problem in G is solvable. As we will see in the next sub section, there are several groups where there exists a procedure to solve the word problem and to reduce words to normal forms, if these groups can be treated as a regular language. From a geometrical point of view, the MWP is equivalent to the problem of finding shortest paths between pairs of vertices in the CG of G. Given any word $w \in S \cup S^{-1}$, there is an associated edge path in the Cayley Graph $\Gamma_{G,S}$. The path starts at vertex corresponding to the identity and then traverses edges of $\Gamma_{G,S}$ as dictated by w. Conversely, every finite edge path in $\Gamma_{G,S}$ describes a word in the generators and their inverses: by reading off the labels of edges being traversed, and adding an inverse if they are traveling in the opposite direction of the orientation of the edge. Let g and h be two vertices in $\Gamma_{G,S}$ represented by the words w_g and w_h in the set $S \cup S^{-1}$. Using this relation between paths and words in the set S^*, we can define a generic path between vertices in the CG of G as follows:

Definition 2.36. *Let g and h be two vertices in $\Gamma_{G,S}$ represented by the words w_g and w_h in the set $S \cup S^{-1}$. A* ***generic path*** *between g and h is the one represented by $w_g^{-1} w_h$.*

The definition of this path is clear by the fact that w_g and w_h represent paths from the identity vertex to the vertices g and h, respectively. Thus, if we go back from g to e, following the path dictated by w_g^{-1}, and then go from e to h as dictated by w_h, the path $w_g^{-1} w_h$ is automatically obtained. If $w_g^{-1} w_h = s_1 s_2 ... s_t$, with $s_i \in S \cup S^{-1}$, $1 \leq i \leq t$, then $w = s_1 s_2 \dots s_t$ defines a path from vertex g to h with edges labeled by $\{s_1, s_2, \dots, s_t\}$ in $\Gamma_{G,S}$. Alternatively, finding a path from g to h is equivalent to finding a path from the vertex e to the vertex with label $w = w_g^{-1} w_h$. Therefore, finding a shortest path between a pair of vertices $g, h \in \Gamma_{G,S}$, which are represented by the words w_g and w_h, is equivalent to finding a word $w = s_1 s_2 \dots . s_t$ with minimum-length in the generators such that $w = w_g^{-1} w_h$. Conversely, given any word w_g representing an element $g \in \Gamma_{G,S}$, solving the MWP for w_g is equivalent to finding a shortest path between e and g.

2.3.5 Groups as languages

Besides the algebraic and geometric point of view of a group, groups can also be seen as languages. Let us define an ***alphabet*** A as a finite set of symbols. An element of A is called a ***letter***. A ***string*** on the alphabet A is a finite sequence of letters, that is, an integer $n \geq 0$ and a mapping $\{1,\ldots,n\} \to A$. If $n = 0$, there is a unique such mapping, called of ***nullstring*** and denote by ε. The set of all strings over the alphabet A is denoted by A^*. Let G be a group, A an alphabet and $\phi : A \to G$ a map, which needs not be injective. By interpreting concatenation as an associative multiplication on G, we define a group homomorphism $\pi : A^* \to G$, where A^* is the set of strings on the alphabet A. If w is a string over A, we say that $\pi(w)$ is the element of G represented by w. If the homomorphism is surjective, i.e., $\pi(A)$ generates G as a group, then A is the set of group generators for G. Note that the definitions in Subsection 2.3.1 are equivalent to the above ones by replacing the term “string” by “word”. In the rest of the document both terms will be used interchangeably.

Definition 2.37. *Let w, p, q and u be any (possibly null) strings over A such that $w = puq$. We say that the string p is a **prefix** of w, q is a **suffix** of w, and that u is a **sub-string** of w. For a integer $t \geq 0$, we denote by $w(t)$ the prefix of w of length t, or else w itself if t is greater than $l_A(w)$.*

Definition 2.38. *A **language** over A, denoted by L, is a subset of A^*, together with the alphabet A.*

Given a language L over A, it is possible to use π to denote the restriction of the map $\pi : A^* \to G$ to L. Since π is surjective, any element in the group G could be represented by at least one string in L. Therefore, we can define an equivalence relation between strings on A as follows:

Definition 2.39. *Let $\pi : L \to G$ be a group homomorphism and let u and v be any two strings in L. We say that u and v are **equivalent**, denoted by $u \sim v$, if $\pi(u) = \pi(v)$. Thus, the **equivalence class** of w, denoted by $[w]$, is the set $[w] = \{a \in L | \pi(w) = \pi(a)\}$.*

Note that an equivalence class of any element $w \in A^*$ is completely determined by any one of its representatives. Then, a new multiplication group arises from the equivalence classes of words as follows:

Theorem 2.8 (Theorem 1.1 of [36])**.** *Let $\pi : L \to G$ be a group homomorphism and L' be the set of equivalence classes of strings in L. By defining a multiplication between equivalence classes as $[w][v] = [wv]$, L' forms a group. In fact, the group homomorphism given by $\eta : L' \to G$ is bijective (i.e., $L' \cong G$).*

This group of equivalence classes is a way to define a ***normal or canonical*** form of every element in the group, i.e., each element of the group has a unique representation. In general, the existence of a regular language L (a language that can be defined by means of a regular expression) such that the group homomorphism given by $\eta : L \to G$ exists, allows solving the word problem for G.

Theorem 2.9 (Theorem 2.1.9 of [40])**.** *Let G be a finitely presented group with generator set A. Assume that G is regularly generated, which means that there is a regular language L over A such that $\pi : L \to G$ is surjective, and the inverse image of the identity under π is also a regular language in L. Then the word problem in G is solvable.*

A traditional way to classify languages is by means of the type of machine capable of recognizing them. A ***Finite State Automata (FSA)*** is a particularly simple type of machine, and it turns out that a language is regular if and only if it is recognized by some FSA [46].

Definition 2.40. *A* ***Finite State automaton*** *(or simply* ***automaton****) is a quintuple $M = (S, A, \mu, Y, s_0)$, where S is a finite set called the state set, A is a finite set called the* ***alphabet****, $\mu \colon S \times A \to S$ is a function called the* ***transition function****, Y is a subset of S called the* ***accepted states****, and $s_0 \in S$ is called the* ***start or initial state****.*

Roughly speaking, the main idea is that the automaton starts in s_0 and reads a string w over A, one letter at a time. After reading a letter, the state of the automaton changes according to its actual state, the letter read and the transition function μ. Once all the strings are read, if the state of the automaton is in Y, then the automaton answers Yes, and it is said that w is ***recognized by*** the automaton. Otherwise, it answers No. The language of strings recognized by the automaton M is denoted by $L(M)$. Finally, a FSA is often represented by a directed graph, where a node represents a state and an arc represents a letter that causes the transition from one state to another.

Definition 2.41. *Let G be a group, M be and FSA and $L(M)$ be the language of M. An* ***Automatic Structure (AuS)*** *on G consists of a set A of generators of G, a FSA WA over A, and a FSA M_a over (A,A), for $a \in (A \cup e)$, satisfying the following conditions:*

1. *The map $\pi : L(WA) \to G$ is surjective.*

2. *For $a \in (A \cup e)$, we have $(w_1, w_2) \in L(M_a)$ if and only if $\pi(w_1)a = \pi(w_2)$ and both $w_1, w_2 \in L(WA)$.*

In this definition, WA is called the ***Word-Acceptor Automaton***, M_e the ***Equality Recognizer***, and each M_a, for $a \in A$, a ***Multiplier Automaton*** for the AuS. The AuS is usually represented by $(A, L(WA))$. An ***Automatic Group (AG)*** is one that admits an AuS. If the Word-Acceptor Automaton (WA) accepts a unique word mapping onto each element of G, e.g. choosing the lexicographically least among the shortest words that map onto each element as the normal form representative of that element, we say that the WA has the ***uniqueness*** property. Note also that M_e recognizes equality in G between words in $L(\mathrm{WA})$.

Definition 2.42. *Let $<$ be a given well-ordering of a set A and let $u, v \in A$. Then, the associated **shortlex** ordering $<_A$ of A^* is defined by $u <_A v$ if either $l_A(u) < l_A(v)$, or if $l_A(u) = l_A(v)$ and $u <_A v$*

Definition 2.43. *Let $\leq_A$ be some total order on the alphabet A. An AuS is called **Shortlex Automatic Structure (SAS)** if $L(WA)$ consists of the shortlex representatives of each element $g \in G$; therefore the map $\pi : L(WA) \to G$ is bijective and all paths in $\Gamma_{G,A}$ according to the words of $L(WA)$ are the shortest ones. In other words, $L(WA) = \{w \in A^* \mid w \leq_A v, \forall w, v \in A^*, w =_G v\}$.*

Thus, given a group G with generator set A, a string $w \in L$ is called a ***geodesic*** if it has the minimal length among all strings representing the same element as w. Since the language of all geodesic strings maps finite-to-one onto G, a ***SAS*** is an AuS for G that contains a unique geodesic representative for each $g \in G$. In general, the problem to decide whether or not a group is automatic is undecidable, since the property of being automatic is a Markov property ([47], p. 192). However, if such structure exists, i.e., if it is verified that the group is automatic, then it is possible to perform the following tasks:

1. Using the WA with uniqueness, one can quickly enumerate unique representatives of all words up to a given length.

2. Determine the order of G, by counting the number of words in normal form.

However, the main feature of an AG is its ability to solve efficiently the word problem in quadratic time. Specifically, a given string can be put into normal form in quadratic time, and then the word problem can be solved by testing whether the normal form of two strings are equal using the Equality Recognizer. Formally, it is expressed as follows:

Theorem 2.10 (Theorem 2.3.10 of [40])**.** *Let G be an AG and $(A, L(WA))$ an AuS for G. For any word w over A, we can find a string in L representing the same element of G as w, in time proportional to the square of the length of w.*

The software packages KBMAG [48] and MAF [49] provide an implementation of a procedure for computing the AuS of finitely presented groups. The main objective of these packages is to construct a normal form for the elements of G in terms of its generators, together with a string reduction algorithm for calculating the normal form representative of an element in G, given as a string in the generators of G. The above mentioned packages offer two alternatives to achieve this objective. The first one is to apply the Knuth-Bendix algorithm [50] to the presentation of G, usually using the shortlex orderings on strings, and expecting that the algorithm completes with a finite ***Confluent Rewriting System (CRS)*** (see sub-section 2.3.6). Many infinite groups [46, 51] and all finite ones [52] have a CRS, although, the algorithm may take a long time to find it, or it may require more space than the available one. The second alternative is trying to compute directly the AuS of G. Again it uses the Knuth-Bendix procedure as one component of the algorithm, but it constructs several FSA rather than obtaining a finite CRS. Note that if a group is automatic it may not have a CRS. In fact, a finitely presented group with solvable word problem may not have a CRS [51] and conversely, a group with CRS does not imply that it is automatic (see example 6.2.2 of [40]).

2.3.6 Alternatives to solve the word problem in automatic groups

If the group G is automatic, then the word problem can be efficiently solved. In addition, during the process of finding an AuS for G, several intermediate structures are created, and they can also solve the same problem [46, 53]. A short description of these structures is the following:

- **Finite Confluent Rewriting System (CRS)**: This is a system of reductions (or directed equations) on strings of the form $v \to w$ such that any string that is not in normal form is composed by a sub-string that is the left-hand side of one of these equations. Therefore, any of these strings can be reduced to a unique normal form by replacing the left-hand side of one of the equations by its right-hand side in a finite number of steps. The Knuth-Bendix procedure [50] is an efficient method to make a rewriting system confluent. If the group has a CRS that is finite, then the word problem is solved in a way that is particularly easy to implement on a computer [51].

- **The Index Automaton (IA)**: This is a machine that recognizes the set of strings which are reducible with respect to a given CRS. In general, this automaton tells us if a specific left-hand in a CRS is a sub-string of a given string w. Since this automaton identifies sub-strings of w with left-hand sides in the CRS, we can replace that sub-string by the right-hand side of the equation. It is proved in [54] that this automaton improves the performance (in time) of a CRS when it is used as a string reducer. If the CRS is confluent with respect to a shortlex ordering, then the mapping

$\pi : L(\text{IA}) \to G$ is bijective. It implies that we can enumerate the elements of a group G in shortlex order by enumerating the language of the IA. Section 3.5 of [54] contains a detailed description of this automaton.

- **Word-Differences Automaton**: Any AG has the so-called ***fellow-traveler*** property, that is, the geodesics between any two vertices in the Cayley graph of the automatic group remain within a bounded distance of each other [40]. It implies that there exists a finite set of D (word differences) such that if $u, v \in L(WA)$ and either $u = v$ or $u^{-1}v = s$ for some $s \in S$, and $u(t)$ and $v(t)$ are prefixes of u and v having the same length, then $u(t)^{-1}v(t) \in D$. Therefore, the ***Word-Differences Automaton (WD)*** is the FSA that accepts (u, v) if and only if $u^{-1}v \in D$. In other words, WD can encode the set of rules from a (possibly infinite) Rewriting System. Thus, the set D can be seen as reduction rules for the group G and then, the WD can be used to reduce any word in the group to a normal form. Section 2 of [53] contains a detailed description of this automaton.

Software packages KBMAG and MAF also implement some procedures that attempt to construct the CRS, IA and WD of a finitely presented group.

Chapter 3

Solving the Compact Routing Problem with Greedy Geometric Routing

The Compact Routing Problem (CRP) consists of designing routing schemes (called Compact Routing schemes) that achieve scalable Routing Table (RT) size with respect to the number of network nodes n, i.e., the RT size grows sub-linearly (or lower) in n, with vertex labels of logarithmic size in n, and low stretch. A potential solution to this problem is the set of routing schemes known as Greedy Geometric Routing (GGR), where nodes are assigned some (virtual) coordinates in a metric space and the node along the routing path is the closest neighbor (in this space) to destination. In this chapter we describe the main GGR schemes that have been proposed and discuss its advantages and drawbacks.

3.1 The Compact Routing Problem

The main task of a network is to provide communication among its nodes. Given a source node and a destination node, there may be several *paths* or *routes* from the source to the destination and finding the best of such routes under some criteria is called the *routing problem*. A *routing scheme* is an algorithm that solves the routing problem.

In this work, we consider *distributed* routing schemes in data networks, that is, in each node there is a *routing process* that once a packet arrives, it decides (independently of the other nodes) whether the packet has just reached its destination, and otherwise, it forwards the packet to a neighbor node towards the destination. Each packet has a *header* that contains the destination address and (possibly) other information. The routing process in each node maintains a RT that together with the packet header is used to decide whether to pass the packet to the own node, i.e., the packet has reached its destination, or to forward the packet to one of its neighbors, otherwise. For example, a RTs may store an entry for each destination containing the next node in the path, IP prefixes, vertex labels of the neighbors, port numbers, or other.

The quality of a routing scheme can be measured by the following:

- ***Stretch***: Given a pair of vertices, the quality of a path computed by a routing scheme is measured as the ratio between the length of this path and the length of a shortest path between the same pair of nodes. This measure is called the path (route) stretch. The maximum path stretch among all the source-destination pairs is called the stretch of the routing scheme. In this work we consider that the length of the path is its number of hops.

- ***Vertex Label Complexity***: Any node of the network must be identified by a label. The vertex label complexity is the maximum size (in number of bits) of any label used by any node. Depending on the kind of scheme, the label can be a coordinate in some metric space, and IP address, a number, etc. If the size of a vertex label is polylogarithmic in n, then the scheme is said to be ***succinct***.

- ***Routing Table Size***: The output of a routing scheme is the RT. The size of the RT is the maximum number of bits that it stores at any node at any time.

- ***Memory/Space Complexity***: The space complexity of a routing scheme is measured as the maximum number of bits that are used during the construction the routing scheme together with the size of the RT. [a]

- ***Routing Decision Time***: Once a packet arrives to a node, this quantity measures the time that the routing scheme takes to decide the next hop toward the destination.

- ***Time Complexity***: This is the time that is needed to construct the routing scheme, i.e., the time after which each node is able to take routing decisions.

- ***Message Complexity***: In distributed routing schemes, it is the total number of messages exchanged among nodes to construct the routing scheme and the maximum size (in number of bits) of such messages.

- ***Routing Scheme Scalability***: The scalability of a RT is expressed in terms of the growth of its size with respect to n. The scalability of a routing scheme is expressed in terms of the growth of the space complexity with respect to n. A linear (or higher) growth rate is considered a poor scalability.[b]

[a] In literature on routing schemes, see e.g., Thorup et al. [5], the space complexity is the size of the RT.

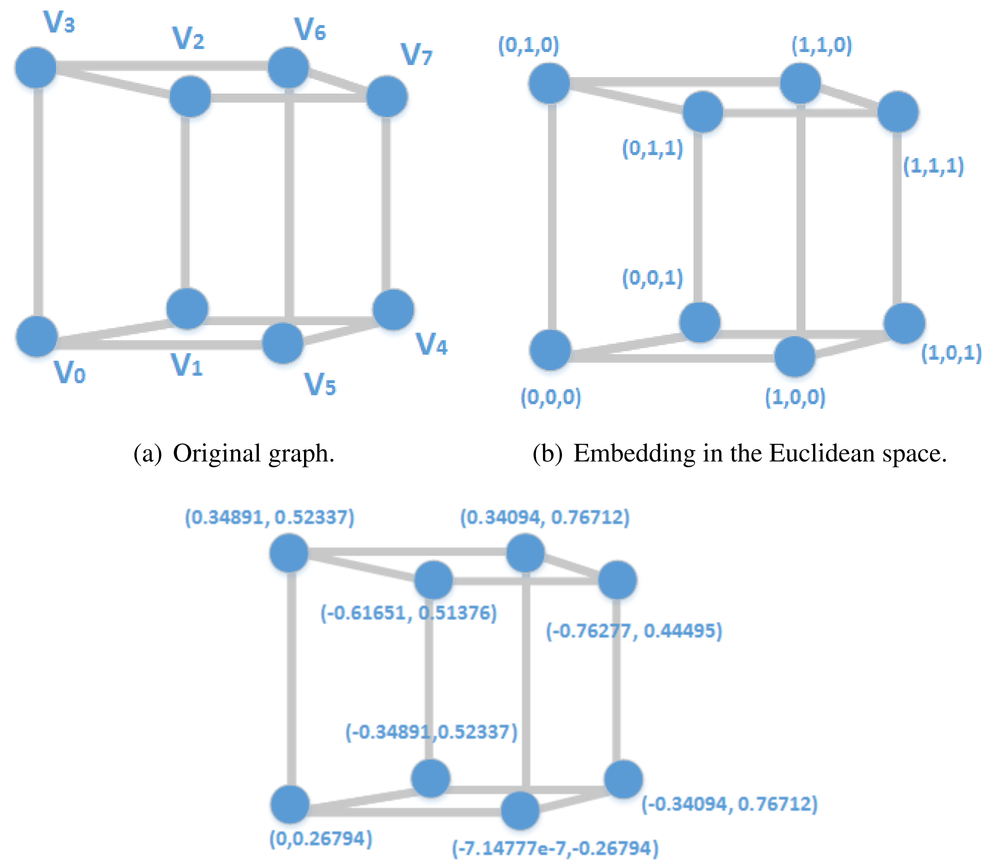
[b] In literature on routing schemes, see e.g., Krioukon et al. [4], the scalability of a routing scheme is the scalability of the RT.

There are two extreme solutions for the routing problem. The first one is to store a RT in each node having an entry for each destination, and each entry containing the output link (or port) through which the packets should be forwarded. With this approach, the packets can be routed through the shortest path following a *point-to-point* (also known as hop-by-hop) routing scheme (see e.g. [55, 56]). It is clear that, in the worst case, this solution requires storing in each node a RT of size $\Omega(n \cdot log(n))$ bits, assuming $log(n)$ bits per node identifier. The second extreme solution is the *source routing* scheme (see e.g. [57]), which includes into the packet header a complete description of the path through the packet must be routed. Again, the packet can be routed through the shortest path. However, the packet header should have a size of $\Omega(n \cdot log(n))$ bits assuming $log(n)$ bits per label identifier and worst-case path with n hops. Therefore, the RTs/packet headers makes both solutions not to scale well [5, 58]. The *Compact Routing Problem (CRP)* consists of designing routing schemes (called *Compact Routing (CR)* schemes) that achieve scalable RT size with respect to the number of network nodes n, i.e., the RT size grows sub-linearly (or lower) in n, with vertex labels of logarithmic size in n, and low stretch (usually constant or logarithmic in n). Usually Compact Routing schemes achieve this scalability at the cost of producing longer paths than the shortest ones [5], and therefore an important desired design goal is to achieve the lowest possible stretch.

CR schemes can be seen as a trade-off between source routing (optimal RT size at the detriment of packet header size) and point-to-point routing (optimal packet header size at the detriment of RT size) by abandoning the requirement that packets are always routed on a shortest path. A CR scheme is said to be *universal* if it works correctly and satisfies promised scaling bounds on any graph. If it does so only on some specific graph families, then it is called *specialized* [4]. Following [5, 58, 59], universal CR schemes with worst case stretch < 3 require at least $\Omega(n)$ bits per vertex for any graph. In other words, shortest path routing schemes are *incompressible*.

3.2 Greedy Geometric Routing schemes

As a potential solution to solve the CRP, *Geometric Routing (GR)* has been proved to be both simple and heuristically effective [17]. GGR schemes exploit the “geometric” dimension of graphs by assigning to each vertex (virtual) coordinates out of a metric space. These coordinates represent the relative position of the vertices as a function of their distances, such that they can be used to forward packets by selecting the closest neighbor (under some criteria in this space) to the destination requiring only local routing information. When the criteria to select the closest neighbor is the one whose distance to the destination is the minimum one among all the neighbors, the forwarding process



(a) Original graph. (b) Embedding in the Euclidean space.

(c) Embedding in the Hyperbolic space.

Figure 3.1: The embedding of the 3-cube graph in several metric spaces.

is referred to as *Greedy Forwarding (GF)*. The family of routing schemes that perform GF using coordinates of some metric space are called *Greedy Geometric Routing (GGR) schemes* [18–20]. In fact, GGR schemes are a type of *Distance Labeling Schemes* (see e.g. [60–63]), that is, routing schemes that rename (or label) the network nodes, such that a quick forwarding process is performed based only on the (approximate) distance to the destination (unique information in the packet header). In this work, we assume that the destination label is known by the source node and is written in the packet header. Figure 3.1 shows a 3-cube graph and its embedding in both the Euclidean space $\mathbb{R}^3$ and the Hyperbolic space $\mathbb{H}^2$ [c].

GGR schemes can be seen as composed by two procedures, the *vertex labeling* and the *packet forwarding*. The vertex labeling procedure computes and assigns coordinates (i.e., labels) to nodes by using some kind of *metric embedding* of the set of nodes (the source metric) into some target metric space (see Section 2.2). The packet forwarding

[c]In this example, a coordinate in the Hyperbolic space is a tuple representing a complex number. For visualization, the coordinates are truncated at the fifth digit right of the decimal point.

procedure uses the GF strategy to select the next hop to forward the packets. These two procedures can be directly mapped to the routing system architecture functional model that we proposed in [64], where the labeling procedure is part of the Process Topology and Routing Information function and the forwarding procedure is part of the Determining Routing Path function.

The potential of GGR schemes for solving the CRP relies on two main points: the embedding process and the topology itself. With respect of the topology, since the number of entries in a RT for a GGR scheme is bounded by $O(\Delta_H)$, topologies with $O(\Delta_H) << n$ can enjoy small RTs in size (assuming vertex label with succinct representation). With respect to the assignment of coordinates to nodes, we can see that it is equivalent to perform an embedding of the graph into a metric space. Therefore, the quality (in terms of stretch) of a GGR built on top of this embedding mainly depends on the quality of the embedding in terms of distortion, a metric that is closely related to the concept of embedding distortion, the dimensions of the metric space (i.e., the number of coordinates used to describe a point in the space), and the number of bits to describe each coordinate. This means that the potential of GGR schemes for solving the CRP relies on the definition of embeddings with low distortion, metric spaces with low dimension and a simple representation of coordinates.

3.3 GGR Schemes with no guarantee of packet delivery

One of the first GGR schemes developed for interconnection networks was presented in [65]. The author proposed a distributed procedure to solve the RT size limitations of the shortest path routing schemes in large scale networks. The nodes are labeled by a two-tuple $< location, id >$. The location represents the position of a node using some coordinate system, and the *id* is a unique identifier for that node. He assumes that the location of the nodes is known (e.g. latitude and longitude in the Earth's surface). However, this approach fails when the packet reaches a *local minimum*, i.e., it reaches a node such that none of its neighbors is closer to destination than itself. He partially solves the problem by searching a node that has a closer neighbor to destination, no farther than $(k-1)$ hops ($k \leq n$), from the node. Note that in the worst case, it is necessary a storage of $O(n \cdot log(n))$ bits to ensure packet delivery in all cases. Other proposals, such as Least Deviation Angle, also called Compass Routing [66], Nearest with Forwarding Progress (NFP) [67] and Most Forwarding progress within Radius (MFR) [68], also solve partially the local minimum problem and therefore do not guarantee the packet delivery.

In order to avoid the need of using either a Global Positioning System (GPS) or any similar device to obtain real (physical) location coordinates, Rao et al. [69] introduced the idea of assigning virtual coordinates (in some space) to the nodes, and then to forward packets using GF. Later, Caruso et al. [70] proposed a GGR scheme based on the computation of a virtual coordinate system based on the hop account metric, a metric that tends to be close to the one on $\mathbb{R}^3$ when the node density is high (the probabilistic analysis presented in [71] confirms it). However, there is still no guarantee of packet delivery.

3.4 GGR schemes with guarantee of packet delivery

To overcome the local minimum problem, other proposals such as Greedy Perimeter Stateless Routing (GPSR) [19] and Greedy Other Adaptive Face Routing (GOAFR) [72] (or its improvement GOAFR+ [16]) include mechanisms to guarantee the success of the routing process. These proposals assume that each node knows its own position, either from a GPS device or through other means, and also requires computing an underlying subgraph that needs to be planar. These algorithms are a combination of Greedy Forwarding and Face Routing (also called Compass Routing II [66]). When a local minimum is detected, the algorithms change the operation mode from Greedy to Face Routing. A practical improvement of Face Routing was presented in [73]. However, these proposals (and similar) have several drawbacks: first, the recovery mechanism increases the path length proportional to the square of the optimal ones; second, although the node coordinates correspond with points in the two-dimensional Euclidean Space, i.e., $\mathbb{R}^2$, the description complexity of each coordinate requires $O(n \cdot log(n))$ in the worst-case [18] (the same space complexity of the RT of a shortest path routing scheme); and finally, if the network topology violates the assumption of the unit disk model [74], then the planarization algorithms may produce graphs that are either non-planar or non-connected planarized in the real topology [75]. Some improvements in the computation to guarantee both a constant stretch and bounded degree for the underlying graph are proposed in [76, 77], and for mitigating the impact of the network model in [75].

The first theoretical work to guarantee packet delivery using GGR was presented in [31], where it is proved that any graph containing a 3-connected planar sub-graph has a greedy embedding in $\mathbb{R}^3$. In addition, they conjectured that every of such graph has a greedy embedding in $\mathbb{R}^2$. Recently, several works confirmed this conjecture [18, 78]. However, $O(n \cdot log(n))$-bits are required to described the coordinates. A more general approach with a distributed algorithm to construct the embedding was proposed in [32] for static graphs. The author proved that there always exists a greedy embedding of any connected finite graph into the two-dimensional Hyperbolic space $\mathbb{H}^2$. As a direct consequence of the greedy embedding, any spanning subgraph defines a greedy embedding of its supergraph yielding arbitrary but low stretch. An alternative algorithm for greedy embedding in dynamic graphs using $\mathbb{H}^2$, Gravity-Pressure Greedy Forwarding (GPGF), was presented in [79]. Although the last two approaches allow the embedding of any finite graph into a two-dimensional space with packet delivery guarantee, the coordinate description is still bounded by $O(n \cdot log(n))$ bits [18].

3.5 GGR schemes with succinct representation of the coordinates

Although most of the works related with graph embedding (from a pure mathematical view point) are impractical due to their high time and space complexity [34, 80–84], one of the most relevant results is that obtaining an isometric embedding of any graph into the Euclidean metric space requires at least $O(log(n))$ dimensions with succinct representation of coordinates. Specifically, in [34] the author proved that every tree has a greedy embedding in $\mathbb{H}^3$ and $\mathbb{R}^{log(n)}$ with coordinate description of $O(log^2(n))$. Recently, the works of Goodrich et al. [85] (for 3-connected graphs) and Eppstein et al. [20] (for any graph), which are based on the techniques used in [34], proposed a greedy embedding into $\mathbb{R}^2$ and $\mathbb{H}^2$, respectively, with succinct representation of $O(log(n))$ bits per coordinates. However, these embeddings are based on heavy path decomposition procedure, which requires a centralized algorithm that runs in $O(n \cdot log(n))$ and needs additional space of $O(n)$ bits [86].

3.6 GGR schemes in scale-free topologies

Many GGR schemes are usually designed without considering topological information about the structure of the network. However, if one could exploit the structure of the topology, then it would be possible to construct graph embeddings on which the GGR schemes built on top of them have good properties in terms of space and time complexity, label complexity and stretch. Real-world topologies such as the Internet, the World Wide Web, biological interactions, social and collaboration networks, among others, are part of the so-called *scale-free networks* [87]. Two of the most important properties of these graphs are the *power-law vertex degree distribution* [88] and that their diameter growth is proportional to $log(n)$ [89]. This property is called *the small-world effect* [90].

In 2002, the scheme called Global Network Positioning (GNP) [91] is built on top of an embedding of the Internet topology at the level of Autonomous System (AS) into the Euclidean space with seven dimensions $\mathbb{R}^7$. It tries to approximate the network distance between any pair of nodes by the Euclidean distance of their assigned coordinates. However, this scheme would incur in large stretch due to the triangle inequality violation in the Internet distances [92]. In fact, in [92] the authors proved that the Euclidean metric is a good estimation of the distance between nodes that are far away from each other, but it incurs in high distortion for closer nodes. In addition, they also proved that the embedding would suffer high distortion even if the dimensions of the embedding were increased. Shavitt and Tankel [93] showed experimentally that the Internet AS topology maps better (with low stretch) into a low-dimensional hyperbolic space than into a Euclidean space with similar dimensions.

Serrano et al. [94] found a geometrical interpretation of the structure and properties of the scale-free graphs, in which the nodes are located in some hidden metric spaces and the empirical results presented in [95] confirm it. In [95] the authors showed that GGR in scale-free networks with strong clustering (e.g. Internet-like topologies) can be performed in a very efficient way without global knowledge of the topology. In fact, the analysis presented in [96] proved that the scale-free network topologies are congruent with the hyperbolic metric space. They used the GPGF routing scheme proposed by [79] to show that even under highly dynamic networks, the congruence is holding. However, the stretch is unbounded and the coordinate description is not succinct. A procedure to perform the inverse process on real AS Internet topologies, i.e., to get coordinates in the hyperbolic metric space for the Internet topology nodes, was presented in [97]. This work provides important facts to understand the direct relationship between scale-free topologies and the hyperbolic metric spaces. However, a pure GGR scheme can not guarantee the packet delivery in all the cases, and again, the coordinate description is not succinct. As an alternative, in [98] the authors presented a greedy embedding with succinct coordinate description of $O(log^2(n))$ bits on scale-free topologies. However, the number of coordinates remains unbounded. Table 3.1 summarizes the main features of the routing schemes analyzed in this section.

Reference	**Advantages/Contributions**	**Packet delivery guarantee**	**Stretch upper bound**	**Other drawbacks**
[65]	It is independent of the coordinate system.	No	No	It requires $O(n \cdot log(n))$ bits of storage.
[66–68]	It solves partially the local minimum problem and introduces new forwarding techniques.	No	No	The input graph must be planar.
[69–71]	It introduces the idea of assigning virtual coordinates in some metric space.	No	No	It requires an input graph with high node density.
[16] [19] [66] [72, 73]	It overcomes the local minimum problem.	Yes	No	It implements recovery mechanisms that increase the path length. The vertex label size is $O(n \cdot log(n))$ bits. The graph must be planar.
[76, 77]	Both the stretch and the degree of the underlying graph are bounded. The time complexity is $O(\Delta \cdot log(\Delta))$	Yes	Yes	The input graph must be planar. It requires $O(n)$ messages.
[31]	First theoretical work that guarantees packet delivery. It introduces the concept of greedy embedding.	Yes	No	It requires vertex labels of $O(n \cdot log(n))$ bits. It only works on 3-connected planar subgraph.
[32, 79]	It works on any kind of graph. It can be implemented in distributed environments and its time and space complexity is low.	Yes	No	It requires vertex labels of $O(n \cdot log(n))$ bits.
[20, 85]	It guarantees a succinct representation of the coordinates.	Yes	No	It requires a graph pre-processing with a centralized algorithm that requires $O(n \cdot log(n))$ bits of storage.
[92]	It assigns Euclidean coordinates to Internet topologies at the AS level. It implements a hybrid model to reduce the stretch. This model can be extended to other metric spaces.	Yes	No	Euclidean spaces are not well suited to represent Internet nodes
[93]	By computing the curvature of the Internet, the embedding of the graph into a low dimensional hyperbolic space reaches a low stretch.	Yes	No	It requires a centralized algorithm to construct the embedding.
[96]	Theoretical proof of the negative curvature (hyperbolicity) of the Internet topology.	No	No	It requires vertex labels of $O(n \cdot log(n))$ bits.
[97]	It guarantees a succinct representation of the coordinates. It can be implemented in distributed environments with low time and space complexity.	Yes	Yes	The number of coordinates is unbounded

Table 3.1: State-of-art of GGR schemes for both general and scale-free graphs

3.7 GGR schemes in Data Centers

Nowadays Data Centers (DCs) are becoming huge facilities with hundreds of thousands of nodes, connected through a network. The design of such interconnection networks involves finding graph models that i) have good topological properties (e.g. high connectivity, small degree, etc.) to ensure good performance in terms of throughput, delay, robustness, etc., and ii) allow to have routing algorithms with both low stretch and low computational complexity even if the number of nodes in the network grows exponentially [6, 99]. In fact, the design of efficient routing algorithms with low stretch is one of the major challenges of the Information and communications technology sector with regard to the reduction of energy consumption [14]. Therefore, a way to face that challenge is by exploiting the geometric properties of the DC topologies in order to find shortest routes (or routes with low stretch) with small RTs. For a survey and taxonomy on DCs, we refer the readers to [100].

Many DCs topologies have been proposed in the literature. For instance, Fat-Tree [8], BCube [9], DCell [101] and FiConn [102] provide high performance with good topological properties. However, the previous proposals for mapping graphs (or at least a family of them) into some metric space can not be directly applied to this kind of topologies. The main reason is the difficulty of guarantee properties such as symmetry, fault tolerance, complete success packet delivery, bounded degree, logarithmic diameter, etc., under such maps. As an alternative, several DC architectures have been proposed such that their underlying topologies emerge from some metric space. In [103], the authors proposed a DC architecture based on tessellations of the hyperbolic metric space. In this topology, each node is identified with a coordinate in the hyperbolic plane. Because of the topology is congruent with its underlying hyperbolic geometry, any path computed by GGR is also a shortest path (even under network dynamics). A DC based on a 3D-Torus, called CamCube, was presented in [104]. This topology is regular, recursive and supports GGR schemes such as the one proposed in [105]. However, this topology suffers from large diameter (slow polynomial diameter growth). Y. Shin et al. proposed in [106] a DC topology based on the d-dimensional torus lattice, called Small-World Datacenter (SWDC), following the small-world model of [107]. This topology improved the performance of the CamCube by including random connectivity among nodes. However, the use of a regular lattice underlying the topology limits the incremental expansion of the network.

Another kind of topologies that have been proposed for DCs architecture are those ones based on Cayley Graphs (CGs) because they meet the desired properties i) and ii) mentioned above [108, 109]. Note that the definition of a CG implies that the vertices are elements of some group but it does not imply any specific group. This flexibility allows finding the graph that better meets the desired requirements on diameter, vertex degree, number of nodes, etc [110]. Moreover, it has been demonstrated that CGs can also be used as models of deterministic small world networks [111]. In general, the routing schemes for CG have been designed specifically for specific topologies in order to take advantage of their algebraic structure. Many of these schemes are based on permutation representation that admits greedy forwarding techniques (because the shortest path problem is equivalent to find an optimal sorting of integers in such topologies). Examples of this kind of topologies are the hypercube and butterfly graphs [10], the toroid graph [112], alternating-group graph [113], complete transposition, bubble-sort and star graphs [114, 115]. However, there are other CG-based topologies that either do not admit a simple greedy forwarding strategy based on permutation representation or can not ensure shortest paths. Examples of these CGs are the pancake [108] and the borel graphs [116, 117].

To our best knowledge, the only shortest path routing algorithm for any finite CGs was presented in [116]. In this work, the authors proved that all finite CGs can be represented by Generalized Chordal Rings (GCR) and based on this transformation, they proposed an iterative routing algorithm based on table look-up (which is constructed using a shortest path algorithm such as Dijkstra or Bellman Ford). However, the proposed scheme is not scalable. In general, all the schemes previously mentioned do not exploit the intrinsic geometric structure of any CGs when they are treated as metric spaces (see Section 2.3.3). Recently, Ji-Yong Shin presented in [118] a routing scheme over a specific CG that exploits the geometric structure of the underlying topology, and which uses a two-level shortest path GGR scheme based on the one proposed in [105]. Table 3.2 summarizes the main features of the routing schemes analyzed in this section.

3.8 Summary and conclusions

In this chapter we have described the main GGR schemes that have been proposed for several types of topologies and we have discussed its advantages and drawbacks. We have seen that these works experience one or more of the following problems: 1) they produce a non-greedy embedding, 2) they are not succinct, 3) they can not be implemented in distributed environments, 4) they require a full knowledge of the topology, or 5) they have unbounded stretch.

Several GGR schemes have been designed to exploit the topological structure of scale-free graphs. In fact, it has been found a geometrical interpretation of the structure and properties of these graphs using Hyperbolic geometry. With respect to DCs, the routing schemes are mostly designed only for specific graphs, and GGR schemes for any CG (i.e., a general purpose scheme) with low complexity have not been proposed. Just a few works exploit the geometric structure of some DCs topologies. Moreover, the family of CG-based DCs do not have (in the literature) any general-purpose GGR scheme that guarantees the shortest paths and that exploits its intrinsic Word-Metrics (WMs) spaces.

Reference	**Advantages/Contributions**	**Packet delivery guarantee**	**Stretch upper bound**	**Other drawbacks**
[103]	A DC architecture that is congruent with the hyperbolic space. The computed paths are the shortest ones.	Yes	Yes	It requires vertex labels of $O(n \cdot log(n))$ bits.
[104]	A CG DCs architecture that supports GGR.	Yes	No	It has large diameter
[106]	A hybrid DC architecture that combines a high regular and recursive underlying network topology with some features of small-world graphs.	Yes	No	It uses a regular lattice as underlying the topology that limits the incremental expansion of the network.
[10] [112–115]	A CG DCs that support greedy forwarding based on permutation representation of groups.	Yes	Yes	Although all of them are CG, each topology has its own routing scheme that can not work on any other kind of CG DC
[116, 117]	Regular and recursive DC architecture with high fault tolerance.	Yes	Yes	The routing scheme is based on a shortest path routing algorithm, and therefore it requires $O(n \cdot log(n))$ bits of storage.
[118]	A completely wireless CG DCs that support greedy forwarding with high aggregate bandwidth, lower latency, and high fault tolerance.	Yes	No	The scalability is worst than in the traditional (wired) DC.

Table 3.2: State-of-art of CG DC architectures and their routing scheme properties

Chapter 4

Greedy Geometric Routing schemes with Word-Metric spaces

In this chapter we describe the main contributions of our work. First we propose a novel and simple greedy embedding of any finite connected graph into a metric space generated by algebraic groups. Then we present three Greedy Geometric Routing schemes built on top of this embedding: the first one for any kind of graph, the second one for scale-free graphs and the third one for Cayley Graphs. We also prove that these last two schemes can be considered as specialized Compact Routing Schemes.

4.1 A greedy embedding in Word-Metric spaces

In this work we propose the use of Word-Metric (WM) spaces, i.e., metric spaces generated by algebraic groups, to build Greedy Geometric Routing (GGR) schemes. Our first contribution is to prove that there exists an embedding of any graph into this metric space and that this embedding is greedy. We state our main theorem.

Theorem 4.1. *Every connected and finite graph $H = (V, E)$ has a greedy embedding in the WM space of an algebraic group.*

The proof of Theorem 4.1 requires finding a distance-decreasing monotone map from vertices of the graph to points (or elements) of a word-metric space generated by an algebraic group and verifying the correctness of Algorithm 1 that performs such mapping. In order to prove Theorem 4.1, firstly, we prove that the algebraic group that generates the target Word-Metric space is the free group of rank $|2 \cdot \Delta_H|$.

Proposition 4.1 (Theorem 3.20 of [39])**.** *A group is free if, and only if, it acts freely on a tree.*

Algorithm 1 Greedy embedding in WM spaces.

1: Choose an arbitrary vertex $r \in V(H)$.
2: Use a distributed algorithm to compute a spanning tree T_H rooted at r.
3: Compute the maximum degree Δ_H of the graph.
4: For each $v \in V(H)$, enumerate its children and assign them a unique integer $i \in \{1,\dots,\Delta_H\}$.
5: Assign to each vertex a unique word (label) representing a group element from a specific algebraic group G.

Proposition 4.2. *Let $\Gamma_{G,S}$ be the Cayley Graph (CG) of a group G generated by S. Given an element $g \in G$, the left multiplication $\pi_g : G \to G$ defined by $\pi_g(h) = g \cdot h$ is an automorphism of its $\Gamma_{G,S}$.*

Proof. Let h and k be any two adjacent vertices in $\Gamma_{G,S}$, i.e., $\{h,k\} \in E(\Gamma_{G,S})$. Because h and k are connected by an edge, there exists an element $s \in S$ such that $k = h \cdot s$ by Definition 2.29. In fact, $\{h,k\} = \{h, h \cdot s\}$ is an edge with label s. Now, assume that π_g does not define an automorphism, i.e., the vertices $\pi_g(h) = g \cdot h$ and $\pi_g(k) = g \cdot k$ are not adjacent under π_g. If we replace k by $h \cdot s$, then we get $\pi_g(k) = \pi_g(h \cdot s) = g \cdot h \cdot s$. Let p be the element of G such that $p = g \cdot h$, then $\pi_g(h) = p$, $\pi_g(k) = \pi_g(h) \cdot s$, and therefore $\pi_g(h)$ and $\pi_g(k)$ are connected with an edge with label s by Definition 2.29, but this is a contradiction to the assumption that π_g does not define an automorphism in $\Gamma_{G,S}$. □

Proposition 4.3. *Let $\Gamma_{G,S}$ be the CG of a group G generated by S. Then $\Gamma_{G,S}$ is the infinite connected $2|S|$-regular tree if, and only if, G is a free group with basis S. In fact, if $G = F(S)$, then $\Gamma_{G,S}$ is the word-metric space generated by $F(S)$.*

Proof. Firstly, we prove that if G is a free group with basis S, then $\Gamma_{G,S}$ is a tree. Note that if G is generated by S, then the graph $\Gamma_{G,S}$ is a connected and $2|S|$-regular graph by Definition 2.29. Note that if G is free, there is no freely reduced word in G representing the identity, and according to the correspondence between words in the group G and paths in its CG, the graph $\Gamma_{G,S}$ has no cycles. Therefore $\Gamma_{G,S}$ is a tree by Definition 2.4. Secondly, we prove that if $\Gamma_{G,S}$ is a tree, then G is a free group with basis S. If $\Gamma_{G,S}$ is a tree, then it is connected and generated by S. Given an element $g \in G$, left multiplication $\pi_g : G \to G$ defined by $\pi_g(h) = g \cdot h$ is an automorphism by Proposition 4.2. In fact, it is also an isometry by Definition 2.10, i.e., $d_s(\pi_g(x), \pi_g(y)) = d(x,y), \forall x,y \in V(\Gamma_{G,S})$. Therefore, the action of G on its CG is by isometries (Theorem 2.7). In addition, this action is free, i.e., for any $g,h \in G$, the existence of an $x \in V(\Gamma_{G,S})$ with $g \cdot x = h \cdot x$ implies $g = h$, otherwise, the path defined by $g \cdot h^{-1}$ is a cycle of length $l_s(g \cdot h^{-1}) \geq 2$, which contradicts the assumption that $\Gamma_{G,S}$ is a tree. Since a group is free if, and only if, it acts freely on a tree by Proposition 4.1, then $G = F(S)$. Finally, considering each edge of $\Gamma_{G,S}$ to be the isometric image of $[0,1]$, the word-metric of the group corresponds to the graph metric induced on $\Gamma_{G,S}$ (Definition 2.31), and the proposition follows. □

At this point, we have proved that the algebraic group G mentioned in Algorithm 1 is the free group of rank $|2 \cdot \Delta_H|$. The next step for proving Theorem 4.3 consists in defining an isometric embedding of a spanning tree of H into $\Gamma_{G,S}$ and then proving that it is also greedy.

Select an arbitrary vertex $r \in V(H)$. Construct a spanning tree rooted in r. Compute the maximum vertex degree Δ_H. Define an ancestor/descendant relationship between vertices: for some $v \in V(T_H)$, choose an ordering of the children of v and denote the *i-th* child of v as $c_i(v)$, for $i \in \{1,\dots,d\}$. Define the mapping function $f : T \to F(S)$ as follows: $f(r) = e$ and recursively for some $v \in V(T_H)$, let $f(c_i(v)) = f(v) \cdot s_i$ for all the children of v. Label each vertex $v \in V(T_H)$ with the word $f(v)$.

Proposition 4.4. *Let $F(S)$ be the free group with $|S| = \Delta_H$. Then the function $f : T_H \to F(S)$ defined above is a greedy embedding of T_H into $F(S)$.*

Proof. We claim that for every pair $s,t \in V(T_H)$ there is a vertex $u \in V(T_H)$ adjacent to s such that $d(f(u), f(t)) < d(f(s), f(t))$. Firstly, $f(c_i(v))$ must exist since there are at least as many group generators as children. Let $f(s) = w_s$, $f(t) = w_t$ and $f(u) = w_u$ be the assigned words in $F(S)$ to the vertices $s,t,u \in V(T_H)$, respectively. Depending on the relative position of s and t with respect to r, there are 3 cases to analyze:

1. If ***t* is an ancestor of *s***, that is, t lies in the path from s to r, then take u as the parent of s. Let s be the *i-th* child of u. Since t is an ancestor of s, we have $f(u) = w_u = w_t \cdot w$, where $w \in F(S)$ and $l_s(w) \geq 0$. So, $d(f(u), f(t)) = l_s(w_u^{-1} \cdot w_t) = ls(w^{-1} \cdot w_t^{-1} \cdot w_t) = l_s(w^{-1})$. Note that $f(s) = w_s = w_u \cdot s_i$, and then $d(f(s), f(t)) = l_s(w_s^{-1} \cdot w_t) = l_s(s_i^{-1} \cdot w_u^{-1} \cdot w_t) = l_s(s_i^{-1} \cdot w^{-1} \cdot w_t^{-1} \cdot w_t) = l_s(s_i^{-1} \cdot w^{-1})$. Therefore, we have $l_s(w^{-1}) < l_s(s_i^{-1} \cdot w^{-1}) \Rightarrow d(f(u), f(t)) < d(f(s), f(t))$.

2. If ***t* is a descendant of *s***, then w_t can be written as $w_t = w_s \cdot s_i \cdot w$, for some $w \in F(S)$, $s_i \in S$ and $l_s(w) \geq 0$. Take u as the single child of s such that $f(c_i(s)) = f(u) = w_s \cdot s_i$, and the proof follows.

3. If ***t* and *s* are not ancestor of each other**, then the path from s to t goes through the parent of s. Take u as the parent of s, and the rest follows.

Finally, in order to establish the correctness of Algorithm 1, we must prove that if $v,u \in V(T_H)$ are adjacent, then $d(f(v), f(u)) = 1$. By contradiction, assume that $d(f(v), f(u)) \neq 1$. Suppose that u is the *i-th* child of v, and let $f(v) = w_v$ be the assigned word in $F(S)$ to the vertex $v \in V(T_H)$ by Algorithm 1. Since $f(u) = f(v) \cdot s_i = w_v \cdot s_i$, then we have that $d(f(v), f(u)) = l_s(w_v^{-1} \cdot w_v \cdot s_i) = l_s(s_i) = d_s(e, s_i) = 1$ by Definition 2.31, but this is a contradiction. The same applies if v is the *i-th* child of u. □

At this point we have proved that the embedding of a spanning tree T_H into $F(S)$ (with $|S| = \Delta_H$) is greedy. Combining this results with Lemma 2.1, we are able to prove our main Theorem 4.1.

Proof. (Theorem 4.1): We need to prove that if T_H is a spanning tree of H and $f : T_H \to F(S)$ is a greedy embedding, then $f : H \to F(S)$ is also a greedy embedding. Because H is connected, then it is always possible to construct a spanning tree T_H by Proposition 2.2. Note that the map $f : T_H \to F(S)$ is a greedy embedding of T_H into the metric space generated by $F(S)$ by Proposition 4.4, but the greedy embedding of any spanning subgraph of H is also a greedy embedding of itself by Lemma 2.1. Therefore $f : H \to F(S)$ is also a greedy embedding of H in the word-metric space generated by the group $F(S)$. □

4.2 A GGR scheme in WM spaces for any kind of graph

In this section we propose a Greedy Geometric Routing (GGR) scheme for any connected and finite graph on top of the previous embedding. We prove its vertex label size, Routing Table (RT) size, stretch, and routing decisions time upper bounds in terms of Δ_H and $D(T_H)$. A simple application case and the worst-case graph analysis is also presented.

4.2.1 The GGR scheme and its complexity analysis

In Section 3.2 we showed that GGR schemes can be seen as composed by two procedures, the vertex labeling and the packet forwarding. The first procedure is performed by Algorithm 1, which gives to each vertex a unique label representing an element of the free group. The second procedure uses the Greedy Forwarding (GF) strategy to select the next hop to forward the packets. This procedure requires a mechanism to determine the distance between vertices using their labels in the WM space, that is, for any pair of vertices $u, v \in H(V)$ with labels $w_u, w_v \in F(S)$, we need a procedure to compute $d_s(f(u), f(v)) = min\{l_s(w) \mid w \in F(S), w = w_u^{-1} \cdot w_v\}$. As solving this problem is equivalent to solve the minimum-length word problem (MWP) in groups (See Definition 2.33), then we propose to compute the distances between vertices using the Confluent Rewriting System (CRS) of $F(S)$. Given a graph H, the complexity analysis of this GGR scheme is summarized in the following four theorems:

Theorem 4.2. *Any GGR scheme built on top of the proposed greedy embedding produces labels of size $O(D(T_H) \cdot log(\Delta_H))$.*

Proof. For any vertex, the number of generators in its label is bounded by the tree depth of the spanning tree $td(T_H)$. However, it is well-known that $td(T_H) \leq D(T_H)$. Since each generator can be represented by $O(log(\Delta_H))$ bits, then the *label vertex complexity* is bounded by $O(D(T_H) \cdot log(\Delta_H))$. □

Theorem 4.3. *Any GGR scheme built on top of the proposed greedy embedding produces RT of size $O(\Delta_H \cdot D(T_H) \cdot log(\Delta_H))$.*

Proof. Each vertex stores a RT containing, for all neighbors, the label and the corresponding output identifier. Note that this information is for all neighbors because not only the edges in the spanning tree are used to forward traffic but all edges Because there is at most Δ_H neighbors and each one of them has a label of size $O(D(T_H) \cdot log(\Delta_H))$, the RT size is bounded by $O(\Delta_H \cdot D(T_H) \cdot log(\Delta_H))$. □

Theorem 4.4. *Any GGR scheme built on top of the proposed greedy embedding has a stretch bounded by $O(D(T_H))$.*

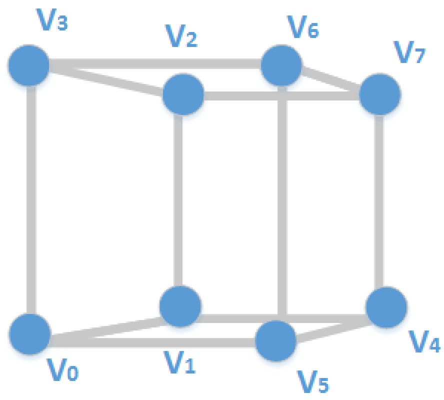
Proof. if the vertices u and v are at distance at least two each other in H, and if their only common parent in T_H is the root vertex, then $d_{T_H}(u,v) \leq 2 \cdot D(T_H)$. For this reason, the *stretch* upper bound of our scheme is $O\left(\frac{d_{T_H}(u,v)}{d_H(u,v)}\right) = O(D(T_H))$. □

Theorem 4.5. *Any GGR scheme built on top of the proposed greedy embedding takes routing decisions in $O\left(\Delta_H^2(H) \cdot D(T_H)\right)$ steps.*

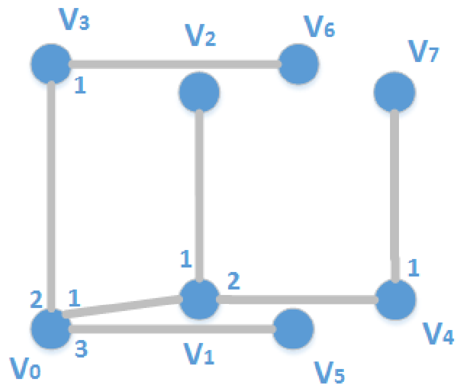
Proof. For free groups, the set of rules of the form $s \cdot s^{-1} \to e$ and $s^{-1} \cdot s \to e$, for all $s \in S$, form a CRS for $F(S)$ (Section 2.3, [54]). Due to any element $s \in S$ can be represented by $O(log(\Delta_H))$ bits, then the required space to store the CRS is bounded by $O(|S| \cdot log(\Delta_H))$. The CRS can reduce any word $w \in F(S)$ to a unique minimal length word in $O(|S| \cdot l_s(w))$ steps by using a simple replacement process of strings (Section 2.4, [54]). Note that the GGR scheme uses at most $D(T_H)$ generators in any vertex label, and therefore $l_s(w) = l_s(w_u^{-1} \cdot w_v) \leq 2 \cdot D(T_H)$. Since $|S| = \Delta_H$ and any vertex has at most Δ_H neighbors, the routing decisions take $O(\Delta_H^2(H) \cdot D(T_H))$ in each vertex while the required space for this computation is $O(\Delta_H \cdot log(\Delta_H))$. Therefore, the distance computation does not increase the space complexity of the proposed embedding. □

4.2.2 Using the GGR scheme in WM spaces: a simple case on a 3-cube graph

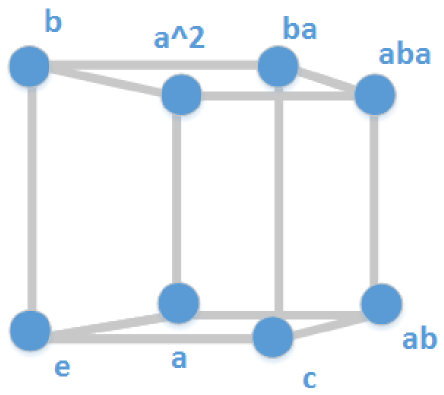
We present a simple example of how our greedy embedding would work on a 3-cube graph modeling a 8-node network. Figure 4.1 (a) shows the topology of the input graph. Let v_0



(a) Original graph



(b) The computed BFS spanning tree rooted at v_0



(c) The resulting embedding of the 3-cube in the WM space generated by the free group of rank 6

Figure 4.1: An example of graph embedding in the WM metric space for GGR.

be the vertex selected by Algorithm 1 to be the vertex r. Then we construct a spanning tree rooted on v_0 (e.g. using a Breadth-First Search (BFS) algorithm) and determine the value of Δ_H. For each $v \in V(H)$, we enumerate its children and assign them a unique integer $i \in \{1,\ldots,\Delta_H\}$. The resulting spanning tree and the enumeration of the vertices is shown in Figure 4.1 (b). Let S be an alphabet with letters a,b,c, let $S^{-1} = \{a^{-1},b^{-1},c^{-1}\}$ be the inverse of S and $|S| = \Delta_H$. Defining the mapping function $f : T \to F(S)$ as $f(r) = e$ and recursively for some $v \in V(T_H)$, $f(c_i(v)) = f(v)\cdot s_i$ for all the children of v, each vertex $v \in V(T_H)$ is labelled with the word $f(v) \in F(S)$). Figure 4.1 (c) shows the resulting mapping of the 3-cube graph in the WM space generated by $F(S)$.

Assume that vertex v_5 wants to send a message to vertex v_7 (labeled as aba). Firstly, v_5 uses the labels of its neighbors v_0, v_4 and v_6 to create the generic paths between them and v_7 (see Definition 2.36): $(e)^{-1}(aba)$, $(ab)^{-1}(aba)$ and $(ba)^{-1}(aba)$, for v_0, v_4 and v_6, respectively. Secondly, v_5 computes the length of the shortlex word that represents each one of the generic paths using the CRS for $F(S)$: $(e)^{-1}(abc) = abc$, $(ab)^{-1}(aba) = a$ and $(ba)^{-1}(aba) = b^{-1}a^{-1}aba$. Because the length of the reduced word that represents the generic path connecting v_4 and v_7 is the shortest one, v_5 sends the message to v_4. Vertex v_4 does the same process with the labels of its neighbors v_1 and v_7. The reduced words that represent $(a)^{-1}(abc)$ and $(abc)^{-1}(abc)$ are bc and e, respectively. Since e is the empty word, i.e., the word of length 0, v_4 sends the message to v_7, the final destination. Note that although the labeling process is based on a rooted spanning tree, the algorithm has found the a path (the shortest one)between v_5 and v_7 using the whole graph and not only the computed spanning tree.

4.2.3 Worst-case graph analysis

As one can deduce from Subsection 4.2.1, given a graph H, the complexity of this GGR scheme depends on two topological parameters: the maximum degree of the graph Δ_H and the diameter of the computed spanning tree $D(T_H)$. If we consider the worst-case graphs, the complexity of this GGR scheme (in terms of the number of nodes n) is presented in the following proposition.

Proposition 4.5. *For any GGR scheme built on top of the proposed greedy embedding, there exists a connected graph such that the worst-case of Algorithm 1 is:*

1. *The vertex label is $O(n \cdot log(n))$, or*
2. *The RT size is $O\left(n \cdot log^2(n)\right)$, or*
3. *The stretch is $O(n)$.*

Proof. Note that for any graph H, the maximum degree is at most n. Therefore, we need at most $O(log(n))$ bits to describe any generator. Now, for the case 1, assume that H is a path graph of n vertices. The diameter of this graph is $n-1$ and therefore there exists a node with label $O(n \cdot log(n))$ by Theorem 4.2. For the case 2, let H be a star graph of n vertices. In this graph there is a vertex with $\Delta_H = O(n)$, and therefore its RT is bounded by $O\left(n \cdot log^2(n)\right)$ following Theorem 4.3. Note that in both cases, the graph H is also a tree and its embedding in the word-metric space is an isometry, and therefore the resulting stretch is 1. Finally, let H be a cycle graph of n vertices. Any spanning tree in it has a minimum diameter of $\lceil n/2 \rceil$ vertices, and then the resulting embedding will have a stretch bounded by $O(n)$ by Theorem 4.4. □

Therefore, for general graphs the complexity of the vertex label and the RT size of the GGR scheme has a worst-case similar to either the point-to-point or the source routing scheme with shortest paths, while the stretch grows linearly in n.

4.3 A Compact GGR scheme in WM spaces for scale-free graphs

As we have just proved, the worst case of any GGR scheme built on top of the proposed greedy implies a severe limitation in its performance. As an alternative for solving this limitation and ensuring scalability in large-scale graphs, we present a specialized Compact Routing (CR) scheme built on top of the proposed embedding, which exploits the structural and statistical properties of scale-free graphs. In order to build this scheme, we will select a spanning tree such that its diameter is bounded by some function in terms of either the diameter or the number of nodes of the graph.

4.3.1 Spanning tree selection

Given a graph H, we showed in Section 4.2 that the selection of the spanning tree T_H plays a fundamental role in the complexity and performance of the constructed GGR scheme. In general, the following three main issues should be studied for the spanning tree computation:

1. How the selection of the vertex root will affect the structure of the resulting spanning tree.
2. How the maximum vertex degree will affect the complexity of the label description and RT size.
3. How the exploration strategy to create such spanning tree, specifically the diameter of the resulting spanning tree, will affect the stretch of the resulting routing scheme.

According to [119] and [120], the selection of the root has minimum impact in the structure of the computed spanning tree, and the resulting stretch of greedy embedding in tree metrics is almost independent of the degree of such root. On the other hand, the maximum vertex degree, which affects the number of generators for the CRS and the maximum number of entries of the RT, does not affect (as a factor) neither the label description complexity nor the stretch value. However the number of entries in the RT will be always limited for this value since our scheme is greedy. Therefore, the main parameter to be considered for selecting a spanning tree would be its depth, which affects the stretch and the label description. Table 4.1 presents some alternatives to compute a rooted spanning tree and their computational complexity in a distributed environment.

Spanning Tree	Time	Space	Message	
			Size	Number
Approximated Minimum Spanning Tree (AMST) [121]	$O(D+log^2(n))$	$O(\Delta_H \cdot log(n))$	$O(log(n))$	$O(m \cdot log^2(n))$
BFS [122]	$O(D_H)$	$O(\Delta_H)$	$O(1)$	$O(D \cdot m)$
Depth-First Search (DFS) [123]	$O(n)$	$O(\Delta_H \cdot log(n))$	$O(n)$	$O(n)$
Minimum Diameter Spanning Tree (MDST) [124]	$O(n)$	$O(n \cdot log(n))$	$O(log(n))$	$O(n \cdot m)$
Minimum Degree Spanning Tree (MdST) [125]	$O(n)$	$O(\Delta_H \cdot log(n))$	$O(log(n))$	$O(n \cdot m)$

Table 4.1: Strategies for distributed computing of a rooted spanning tree from an undirected, unweigthed and connected graph.

Note that the only two algorithms that guarantee an upper bound for the tree-depth (in terms of the input graph diameter) are both the BFS and the MDST, for any selected root (see Proposition 4.6 below). Although it is clear that the minimum stretch of any of these spanning trees is obtained by the MDST algorithm, the computation of the MDST requires to run an All Pairs Shortest Path (APSP) algorithm and it implies $O(n \cdot log(n))$ bits in addition to the RTs (the MDST problem is NP-hard). Therefore, the best trade-off between complexity and the stretch is reached by the distributed version of the BFS proposed by [122].

Proposition 4.6. *Given a graph $H = (V,E)$ and a spanning tree T_H constructed by the BFS algorithm, $D(T_H) \leq D(H)$.*

Proof. By Theorem 22.5 of [25], a BFS tree rooted at vertex r is a subgraph of H such that for any $v \in V(H)$, $d_{T_H}(r,v) = d_H(r,v)$. Assume that $D(T_H) > D(H)$. Therefore, there exists a vertex $u \in V(H)$ such that $d_T(r,u) > d_H(r,u)$, but it is a contradiction. □

As we discussed before, the worst-case of a GGR scheme for any kind of graph is independent of the selected spanning tree: some graphs could have $D = O(n)$, e.g. take the cycle graph of order n, and therefore any computed spanning tree will have a tree-depth bounded by $O(n)$. So, in addition to the selection of the spanning tree, it is important to exploit the structural and statistical properties of some families of graphs such that our proposal can be seen as a specialized and distributed Compact Greedy Geometric Routing (C-GGR) scheme using word-metric spaces, when these properties are met.

4.3.2 Exploiting the topological structure of scale-free graphs

Modeling a system composed of individual elements and their relationships by means of networks allows the study of aspects such as the nature of each element, their connections or the regularity of those connections. Many real-world topologies such as the Internet, the World Wide Web, citation networks, biological networks, social and collaboration networks, among others, can be related by a number of common recurring patterns or characteristics in their structures. Two of the most important of such characteristics are *the small-world affect* [90] in the path length between pairs of elements, and the *power-law degree distribution* [88]. Graphs with power-law distribution are also known as *scale-free graphs*. The following results present the relationship between the number of nodes of a scale-free graph and its expected diameter and maximum vertex degree.

Proposition 4.7 (Theorem 22.5 of [25]). *Given a scale-free graph $H = (V,E)$, the expected diameter of H is $O(log(n))$.*

Proposition 4.8 (Theorem 3.1 of [126]). *Given a scale-free graph $H = (V,E)$, the expected maximum degree of H is $O(\sqrt{n})$.*

For the Internet, these values are also true and it allows the development of new communication protocols with optimal performance [1]. In fact, wherever it be at the level of either Internet Router (IR) or Autonomous Systems (ASs), both models are characterized by heavy tailed vertex degree distributions following a power-law form, and the presence of shortcuts that connect far away parts of the network, thus reducing the average path length of the graph, one of the main characteristic of small-world networks, i.e., they are scale-free networks [2]. Figure 1.1 shows the actual Internet topology at the level of ASs.

Combining the properties of scale-free graphs with the low-complexity of the BFS algorithm for computing spanning trees of connected graphs, we claim that a GGR scheme built on top of the proposed embedding is a scalable and distributed C-GGR scheme for this kind of networks. The complexity analysis of this GGR scheme, which proves our claim, is summarized in the following five theorems:

Theorem 4.6. *Given a scale-free graph $H = (V,E)$, a GGR scheme built on top of the proposed greedy embedding produces vertex labels of size $O(log^2(n))$.*

Proof. For any vertex, the number of generators in its label is bounded by tree-depth of the computed spanning tree. Since we are using the BFS algorithm, then the tree-depth of T_H is at most $O(D(H))$ by Proposition 4.6. In addition, we have $O(D(T_H)) \leq O(D(H)) \leq O(log(n))$ by Proposition 4.7. On the other hand, Δ_H is bounded by $O(n^{1/2})$ following Proposition 4.8, and therefore each generator can be represented by at most $O(log(n))$ bits. By replacing these values in Theorem 4.2, the label complexity is bounded by $O(log^2(n))$. □

Theorem 4.7. *Given a scale-free graph $H = (V,E)$, a GGR scheme built on top of the proposed greedy embedding produces RTs of size $O(n^{1/2} \cdot log^2(n))$.*

Proof. By Theorem 4.6, any vertex label in H will require $O(log^2(n))$. Since the expected maximum vertex degree is $O(n^{1/2})$ by Proposition 4.8, then any vertex will require at most $O(n^{1/2} \cdot log^2(n))$ bits to store its RT. □

Theorem 4.8. *Given a scale-free graph $H = (V,E)$, a GGR scheme built on top of the proposed greedy embedding has a stretch bounded by $O(log(n))$.*

Proof. Note that $O(D(H)) \leq O(log(n))$ and therefore $O(D(T_H)) \leq O(log(n))$ by Proposition 4.7. By replacing this result in Theorem 4.4, we obtain $O(log(n))$. □

Theorem 4.9. *Given a scale-free graph $H = (V,E)$, a GGR scheme built on top of the proposed greedy embedding takes routing decisions in $O(n \cdot log(n))$ steps.*

Proof. By replacing the values of $O(D(T_H))$ by $O(log(n))$ (Proposition 4.7) and Δ_H by $O(n^{1/2})$ (Proposition 4.8) in Theorem 4.5, the result follows. □

Theorem 4.10. *Given a scale-free graph $H = (V,E)$, Algorithm 1 performs a greedy embedding $f : H \to F(S)$ in $O(log(n))$ steps, it uses $O(n + log(n) \cdot m)$ messages of size $O(log^2(n))$ and requires and additional space of $O(n^{1/2} \cdot log(n))$ bits.*

Proof. Using the distributed BFS algorithm proposed by [122], we have that $O(D(H) \cdot m)$ messages of size $O(1)$ are needed to construct the BFS tree, and it takes only $O(D(H))$ steps (See Table 4.1). Replacing $O(D(H))$ by $O(log(n))$ (Proposition 4.7) and Δ_H by $O(n^{1/2})$ (Proposition 4.8), we obtain $O(log(n) \cdot m)$ messages to build the BFS spanning tree in $O(log(n))$ steps. The maximum degree computation and labeling process on T_H can be performed using an asynchronous broadcast and convergecast algorithm, which requires $O(n)$ messages and $O(D(H))$ steps (Section 15.3, [127]). The size of the message in each case is $O(log(n))$ (the maximum vertex degree) and $O(log^2(n))$ (the maximum size of any vertex label), respectively. Combining the space, message and time complexity of the BFS construction together with the maximum degree computation and labeling process, the result follows. Finally, in the proof of Theorem 4.5 we showed that the required space to store the CRS is $O(\Delta_H \cdot log(\Delta_H))$, and therefore, the complexity to store the CRS is bounded by $O(n^{1/2} \cdot log(n))$, if the input graph is a scale-free graph. □

In summary, our GGR scheme for scale-free graphs is scalable (because our auxiliary structures do not require more space than the RT itself), distributed and time optimal [122], and solves the Compact Routing Problem (CRP) for scale-free graphs according to the definition given in Section 3.1.

4.4 A Compact GGR scheme in WM spaces for Cayley Graphs

In general, the stretch of a graph embedding in tree metrics depends on the tree-like structure of the input graph [128]. Note that the proposed scheme has stretch 1 if the input graph is a tree. This is because the word-metric space generated by the free group is a tree metric, and the embedding is an isometry. But, is it possible to perform and isometric embedding of any graph into a word-metric space? The answer is yes.

Qin et. al. proved in [129] that any connected graph $H = (V,E)$ with n vertices and m edges has an isometric embedding in the Cayley Graph of an elementary abelian 2-group G of order 2^n. However, the number of generators $S \subset G$ is $\Omega(m)$ and the CRS for the resulting abelian sub-group has $\Omega(m)$ reduction rules [130]. In other words, the CRS requires at least the same space complexity as any routing scheme based on a shortest path algorithm. Although for general graphs the isometric embedding in word-metric spaces is too expensive, many families of Cayley Graphs, which are a word-metric space themselves, have a short CRS. Since many Cayley Graphs are used as underlying graphs for Data Center topologies, a shortest path GGR scheme can be built on top of the proposed embedding.

Theorem 4.11. *Given a finite and connected graph $H = (V,E)$ with underlying Cayley Graph $\Gamma_{G,S}$, Algorithm 1 (using a BFS spanning tree) performs an isometric embedding $f : H \to G$.*

Proof. Note that the vertex exploration strategy of a BFS algorithm gives us an ordering $<_{BFS}$ on V (Theorem 2, [131]). This ordering is called a BFS ordering. Let L be set of strings over S which are the representatives of the equivalent classes of elements in G under the shortlext ordering $<_s$ (see Definition 2.42). Therefore, the map $\pi : L \to G$ is a bijection by Theorem 2.8. However the enumeration of elements of G using a shortlext ordering is in fact a BFS ordering (Section 13.1.2, [41]). Therefore, the enumeration of V and L using a BFS strategy gives a natural mapping $\omega : V \to L$ with $\omega(v_i) = w_i$, for all $v \in V$ and $w \in L$. Because $G \cong \Gamma_{G,S} \cong \langle S|R \rangle$ by Theorems 2.7 and 2.4, and $G \cong L$ by Theorem 2.8, then $\Gamma_{G,S} \cong L$. The last isomorphism implies that the mapping $\omega : V \to L$ is an isometry, i.e., $d(v,u) = d(\omega(v), \omega(u))$, for all $v,u \in V$. □

Among the many existing models for CGs (e.g.,[6, 132–134]), we have selected the following six well-known families for our theoretical analysis: Hypercube, Butterfly, Transposition, Bubble-Sort, Star and Pancake. Table 4.2 summarizes the main topological properties of them (see Subsection 5.2.1 for a more detailed description).

Graph Family	Nodes (n)	Degree	Diameter
Hypercube $H(k)$	2^k	k	k
Butterfly $BF(k)$	$k \cdot 2^k$	4	$\lfloor 3(k)/2 \rfloor$
Transposition $TP(k)$	k!	$\binom{k}{2}$	$k-1$
Bubble-Sort $BS(k)$	k!	$k-1$	$\binom{k}{2}$
Star $ST(k)$	k!	$k-1$	$\lfloor 3(k-1)/2 \rfloor$
Pancake $P(k)$	k!	$k-1$	$17k/16 \leq Diameter \leq (5k+5)/3$

Table 4.2: Main properties of some well-known DC topologies based on Cayley Graphs.

The isometric embedding performed by Algorithm 1 together with the CRS of the algebraic group representing the CG define a GGR scheme for any topology with underlying CG. In fact, this is a general-purpose GGR scheme that guarantees the shortest path on any CG topology. With respect to the complexity of the scheme, it will depend on each graph itself. For the previous six CG-based DCs we have anayzed the complexity and the result is the following:

Theorem 4.12. *Given any graph from Table 4.2, a GGR scheme built on top of the proposed greedy embedding (using a BFS spanning tree) has a vertex label complexity of* $O(log(n) \cdot log(log(n)))$ *bits, RTs of size* $O(log^2(n) \cdot log(log(n)))$ *and the routing decisions take* $O\left(log^3(n)\right)$ *steps. In addition, such embedding can be constructed in a distributed way in* $O(log(n))$ *steps with* $O(n + log(n) \cdot m)$ *messages of size* $O(log^2(n))$.

Proof. By using the Stirling's approximation [135], we obtain that the diameters and maximum (regular) degree of all these graphs are $O(log(n))$. Therefore, by replacing $D(T_H) \leq D(H) = log(n)$ and $\Delta_H = O(log(n))$ in Theorem 4.2, Theorem 4.3 and Theorem 4.5, the first part of the result follows. On the other hand, following a procedure similar to the one described in the proof of Theorem 4.10, we can obtain the last part of the result. □

Unlike free groups, which have a very short CRS of size $O(|S|)$ equations, finite groups can have up to $O(n)$ equations. However, there exist many finite groups that have a short CRS, i.e., the number of rules is bounded by $O(\sqrt{|G|})$[136]. In the next chapter we will evaluate through simulation the size of the CRS and the size of other structures that can be used to solve the MWP (which is the same as the shortest path problem) for the graphs presented in Table 4.2.

4.5 Summary and conclusions

In this chapter we have presented a simple and novel embedding of any finite connected graph into a WM space, i.e., a metric space generated by algebraic groups, and we have proved that any GGR scheme built on top of this embedding guarantees the packet delivery (the embedding is greedy). Then, we have proposed the following three GGR schemes:

- A GGR scheme for any kind of graph, with stretch of $O(D(T_H))$, $O(D(T_H) \cdot log(\Delta_H))$ bits per vertex label, RTs of size $O(\Delta_H \cdot D(T_H) \cdot log(\Delta_H))$ and routing decisions that take $O\left(\Delta_H^2 \cdot D(T_H)\right)$ steps.
- A C-GGR scheme for scale-free graphs (which include many real-world topologies such as Internet), with stretch of $O(log(n))$, $O(log^2(n))$ bits per vertex label, RTs of size $O(n^{1/2} \cdot log^2(n))$ and routing decisions that take $O(n \cdot log(n))$ steps.
- A C-GGR scheme for Cayley Graphs (which are used as a model for Data Center interconnection networks), with shortest paths, $O(log(n) \cdot log(log(n)))$ bits per vertex label, routing tables of size $O(log^2(n) \cdot log(log(n)))$, and routing decisions that take $O\left(log^3(n)\right)$ steps.

While the first GGR scheme works for any kind of graph and its complexity depends on the parameters $D(T_H)$ and Δ_H, the two C-GGR schemes are specialized, and their complexity only depends on n. In addition, these C-GGR schemes can be constructed in a distributed way in $O(log(n))$ steps, using $O(n + log(n) \cdot m)$ messages of size $O(log^2(n))$ and $O(\Delta_H \cdot log(n))$ bits of additional storage to build the RT, i.e., not only the RT is scalable but also the routing scheme itself following the definition given in Section 3.1.

Chapter 5

Experimental Evaluation Through Simulation

In the previous chapter we proposed a greedy embedding of graphs into a Word-Metric space together with three Greedy Geometric Routing (GGR) schemes (one for any kind of graph and two specialized Compact Routing ones) built on top of this embedding, and we proved their algorithm complexities. This chapter presents and discusses the experimental evaluation through simulation of our specialized schemes, firstly the GGR scheme for scale-free networks and secondly the GGR scheme for Cayley Graph (CG) topologies. The goal is to assess how far the experimental results are from the theoretical upper bounds for our schemes and other related works.

5.1 Evaluation of the C-GGR scheme for scale-free network topologies

In Section 4.3 we have designed a GGR scheme built on top of the proposed embedding for scale-free graphs into the Word-Metric (WM) space generated by the Free Group $F(S)$ (with $|S| = \Delta_H$). The embedding process is performed by Algorithm 1 using a Breadth-First Search (BFS) spanning tree, while the distance computation is performed by the Confluent Rewriting System (CRS) of the $F(S)$. In this section, we evaluate through simulation the main performance metrics of a routing scheme in order to verify that this is a Compact Routing (CR) scheme for scale-free networks.

5.1.1 Measured parameters and metrics

The theoretical complexity upper bounds for the proposed GGR scheme for scale-free networks have shown that this is a specialized CR scheme that exploits the structural and statistical properties of this family of graphs. Table 5.1 summarizes the complexity upper bounds of the routing scheme (see Section 3.1 for the description of the metrics). In order to evaluate and compare our Compact Greedy Geometric Routing (C-GGR) scheme in WM spaces for scale-free graphs, we have implemented the Greedy Geometric Routing in $\mathbb{H}^2$ (GRH2) [32] and the traditional Shortest Path Routing on Trees (SPRT). We have selected these schemes for the following reasons: both algorithms are well-known routing schemes and they can be easily implemented and verified in simulators; their time and space complexity are closely related to the computation of a spanning tree of the input graph, and therefore the efficiency of these algorithms can be easily compared; finally, GRH2 is one of the most efficient GGR algorithms (in terms of space and time complexity), and the drawbacks with respect to its vertex label size and stretch upper bound can be mitigated by limiting the size of the evaluated topologies and its diameter.

To ensure a fair comparison among schemes, the same spanning tree constructed by Algorithm 1 is used in the other two schemes. The computed BFS tree is rooted in the vertex with the highest degree in the topology. Because we want to verify (by experimental) that our scheme remains below the upper bound of a specialized CR scheme, we will use as the comparison metrics the vertex label complexity, Routing Table (RT) size and stretch. On the other hand, the other metrics are not used since we are using the same spanning tree for the three evaluated schemes. Finally, for the experimental vs theoretical comparison of the WM algorithm upper bounds, the value of the logarithmic function is taken as $log_2(n) \subseteq O(log(n))$ and for the square root is $2 \cdot n^{1/2} \subseteq O(log(n))$.

	Upper Bounds
Vertex Label Complexity	$O\left(log^2(n)\right)$
Routing Table Size	$O\left(n^{1/2} \cdot log^2(n)\right)$
Stretch	$O(log(n))$
Memory Space Complexity	$O\left(n^{1/2} \cdot log^2(n)\right)$
Routing Decision Time	$O(n \cdot log(n))$
Time Complexity	$O(log(n))$
Message Complexity	$O\left(log^2(n)\right)$

Table 5.1: Complexity upper bounds of the proposed C-GGR for scale-free graphs.

5.1.2 Network topologies

We have selected 3 families of topologies with an incremental number of nodes. Two of them are synthetic ones, while the third one is obtained from a data set of real-world scale-free topologies. Each family of the synthetic topologies consists of graphs of size 25, 50, 75, 100, 250, 500, 750 and 1000 nodes. For each size, we have generated 10 different graphs using different seeds, and then we have run the routing algorithms on these topologies. We have used the mean values of the measurements and we have verified that their Confidence Interval (CI) is low in order to ensure the reliability of them. The description of the selected synthetic topologies and the statistics to validate the simulations runs are presented below.

Barabasi-Albert preferential attachment (BA)-based topologies

This a model widely used to successfully describe the scale-free nature of many networks. The BA model [88] starts with a connected graph with m_0 vertices, then a graph of n vertices is grown by attaching at each step, a new vertex with m_0 edges. These m_0 edges are attached to existing vertices, preferentially to those with high degree. All the topologies have been created by setting $m_0 = 2$. In Table 5.2 we show the main topological characteristics of the generated graphs, together with the diameter and depth of the spanning tree computed by Algorithm 1 using BFS. Table 5.3 shows the statistical values of both the mean and 95% CI of the stochastic topological parameters. By understanding that a large value of CI implies more uncertainty about the evaluated parameter (and vice versa), the 95% CI remains below 15% of the mean value. In other words, if we generate other BA topologies, the expected value of the stochastic topological parameters will remain very similar among all the generated graphs with the same number of vertices.

			Degree			BFS Tree	
Vertices	**Edges**	**Diameter**	**Maximum**	**Minimum**	**Average**	**Diameter**	**Depth**
25	46	4	14	1	3.68	4	3
50	96	5	12	1	3.84	6	5
75	146	6	34	2	3.89	6	5
100	196	6	31	2	3.92	6	6
250	496	6	35	2	3.97	8	5
500	996	7	57	2	3.98	8	6
750	1496	7	70	2	3.99	8	5
1000	1996	7	98	2	3.99	8	6

Table 5.2: Properties of the BA graphs and the diameter of the computed BFS tree.

	Diameter		Vertex Degree				BFS Tree			
			Maximum		Minimum		Diameter		Depth	
Vertices	mean	CI (+/-)	mean	CI (+/-)	mean	CI (+/-)	mean	CI (+/-)	mean	CI (+/-)
25.0	4.0	0.2	14.0	1.7	1.0	0.1	4.0	0.4	3.0	0.3
50.0	5.0	0.2	12.0	1.1	1.0	0.0	6.0	0.7	5.0	0.1
75.0	6.0	0.1	34.0	2.4	2.0	0.1	6.0	0.3	5.0	0.3
100.0	6.0	0.4	31.0	1.6	2.0	0.1	6.0	0.4	6.0	0.4
250.0	6.0	0.1	35.0	4.4	2.0	0.0	8.0	0.7	5.0	0.4
500.0	7.0	0.4	57.0	5.5	2.0	0.2	8.0	0.5	6.0	0.5
750.0	7.0	0.9	70.0	4.9	2.0	0.1	8.0	0.3	5.0	0.3
1000.0	7.0	0.7	98.0	11.0	2.0	0.2	8.0	0.2	6.0	0.1

Table 5.3: Mean and 95% CI of the stochastic topological parameters of the BA topologies.

Holme-Kim preferential attachment (HK) with clustering based topologies

Although the BA model generates graphs with small-world effect and power-law degree distribution, it can not provide an important feature of real-world topologies: the presence of clusters, i.e., a set of nodes more connected to each other than to other nodes. This feature is usually measured by the clustering coefficient of a node, which is computed as the ratio of edges among its neighbors which are actually realized compared with the number of all possible edges. Holme and Kim presented in [137] a model for scale-free networks with high clustering. It is essentially the BA model with an extra step where each added node has a probability p of making an edge to one of its neighbors too. We use $m_0 = 2$ and $p = 0.3$. The main properties of the created graphs and both the diameter and the depth of the computed BFS tree ($td(H)$) by Algorithm 1 are presented in Table 5.4. Table 5.5 shows the statistical values of both the mean and 95% CI of the stochastic topological parameters. As the BA model, the 95% CI remains below 10% of the mean, i.e., the stochastic topological parameters of the generated graphs remain very stable, among all the generated graphs with the same number of vertices. Therefore, the number of simulation runs and the stochastic topological models used in this section provide enough statistical robustness to verify the performance of the schemes.

			Degree			BFS Tree	
Vertices	**Edges**	**Diameter**	**Maximum**	**Minimum**	**Average**	**Diameter**	**Depth**
25	46	4	11	2	3.68	5	4
50	96	5	12	2	3.84	6	4
75	146	6	23	2	3.89	7	5
100	196	5	27	2	3.92	7	4
250	496	7	45	2	3.97	9	6
500	996	8	51	2	3.98	10	8
750	1496	7	68	2	3.99	9	6
1000	1996	8	108	2	3.99	8	7

Table 5.4: Properties of the HK graphs and the diameter of the computed BFS tree.

			Vertex Degree				BFS Tree			
	Diameter		**Maximum**		**Minimum**		**Diameter**		**Depth**	
Vertices	**mean**	**CI (+/-)**	**mean**	**CI (+/-)**	**mean**	**CI (+/-)**	**mean**	**CI (+/-)**	**mean**	**CI (+/-)**
25.0	4.0	0.3	11.0	0.6	2.0	0.2	5.0	0.3	4.0	0.0
50.0	5.0	0.2	12.0	0.6	2.0	0.0	6.0	0.4	4.0	0.1
75.0	6.0	0.2	23.0	1.3	2.0	0.2	7.0	0.3	5.0	0.1
100.0	5.0	0.4	27.0	1.9	2.0	0.2	7.0	0.3	4.0	0.3
250.0	7.0	0.2	45.0	4.5	2.0	0.0	9.0	0.8	6.0	0.0
500.0	8.0	0.5	51.0	3.3	2.0	0.2	10.0	0.9	8.0	0.5
750.0	7.0	0.6	68.0	5.8	2.0	0.0	9.0	0.2	6.0	0.2
1000.0	8.0	0.5	108.0	9.1	2.0	0.2	8.0	0.4	7.0	0.6

Table 5.5: Mean and 95% CI of the stochastic topological parameters of the HK topologies.

Internet Router (IR)-level topologies

Internet topology can be viewed as an undirected graph, where a vertex represents either a router (IR level topology) or an Autonomous System (AS) (AS level topology). Although both models are different, their statistical and structural properties at large-scale make that both models are scale-free [2, 138]. In this experimental evaluation we have selected 4 IR level topologies from the Rocketfuel data set [139]. The properties of each IR level topology and both the diameter and the tree depth of the computed **HS!** (**HS!**) tree by Algorithm 1 are presented in Table 5.6.

				Degree			BFS Tree	
AS	**Vertices**	**Edges**	**Diameter**	**Maximum**	**Minimum**	**Average**	**Diameter**	**Depth**
1755	957	1409	14	36	1	2.94	18	12
3257	1223	1550	16	90	1	2.53	16	11
3967	1480	2688	15	101	1	3.63	21	13
6461	2720	3824	12	162	1	2.81	15	13

Table 5.6: Graph properties of the evaluated AS topologies.

5.1.3 Results in BA-based topologies

Stretch

In Figure 5.1 we show the Cumulative Distribution Function (CDF) of the resulting path stretch for our WM scheme and for GRH2 and SPRT schemes running on a BA graph with 1000 nodes. As in the case of GRH2, the routes chosen by our WM scheme also follow many non-tree edges (shortcuts), thus reducing the stretch with respect to the SPRT. Note that for our WM scheme, around 70% of the routes have an strecht 1, which improves over the other two schemes, and around 92% have an stretch ≤ 1.3. The stretch of our WM scheme is only 4, the same for GRH2, while for SPRT is 8. In general our WM scheme achieves a better performance than the other two ones.

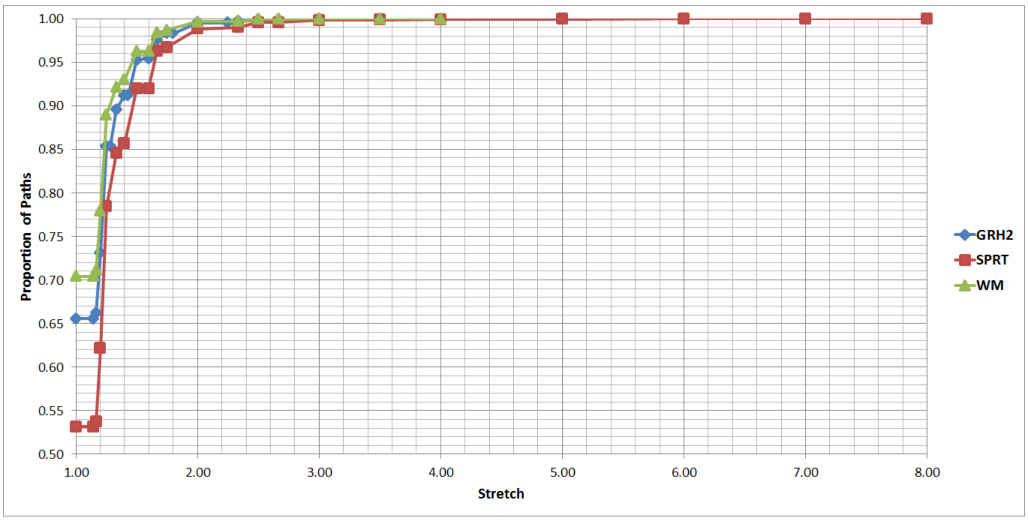


Figure 5.1: Stretch CDF for WM, GRH2 and SPRT on a BA graph with 1000 nodes.

Figure 5.2 and Figure 5.3 show the dependence of the stretch and the average path stretch values with respect to the number of network nodes in BA topologies for the three schemes. Table 5.7 shows the mean and 95% CI of the stretch. Note that in all cases the CI remains below 15% of the mean, i.e., the reliability of the computed stretch is very high.

The resulting stretch is well below the theoretical upper bound of $log_2(n) \subseteq O(log(n))$, while it also scales with the size of the network. In addition, we can see in Figure 5.4 that the 90% of the routes have stretch ≤ 1.5 and over the 98% of them have an stretch less than 2, in all the evaluated network sizes. Again, our routing scheme has a better performance than the other two schemes in almost all the network sizes.

	GRH2		**WM**		**SPRT**	
Vertices	**mean**	**CI (+/-)**	**mean**	**CI (+/-)**	**mean**	**CI (+/-)**
25.0	2.0	0.23	1.5	0.20	4.0	0.22
50.0	3.0	0.33	3.0	0.26	6.0	0.59
75.0	2.5	0.25	2.5	0.28	5.0	0.38
100.0	3.0	0.23	3.5	0.37	6.0	0.44
250.0	3.5	0.36	3.0	0.27	7.0	0.70
500.0	4.0	0.36	4.0	0.37	8.0	0.85
750.0	4.5	0.26	3.5	0.29	8.0	0.86
1000.0	4.0	0.49	4.0	0.38	8.0	0.99

Table 5.7: Mean and 95% CI of stretch in the BA topologies.

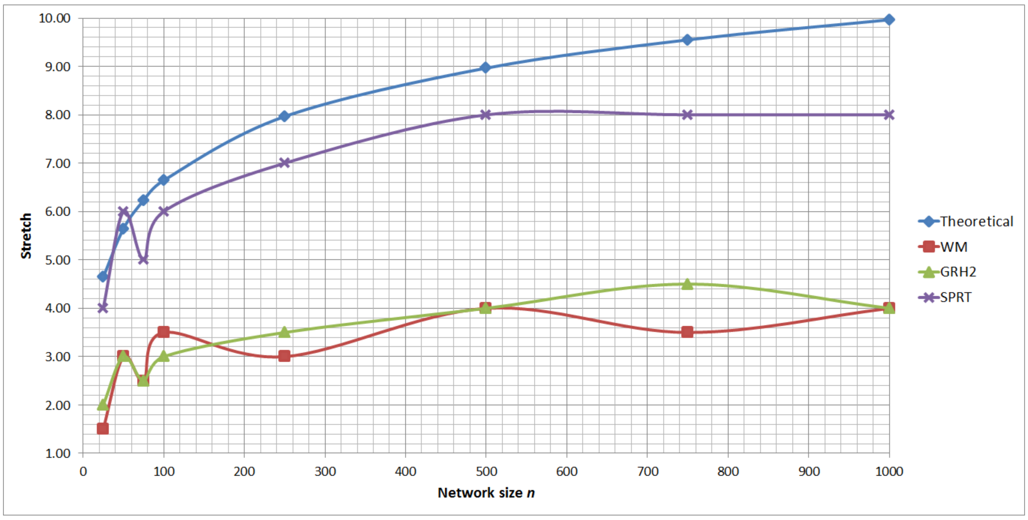


Figure 5.2: Stretch as a function of the number of nodes on BA graphs.

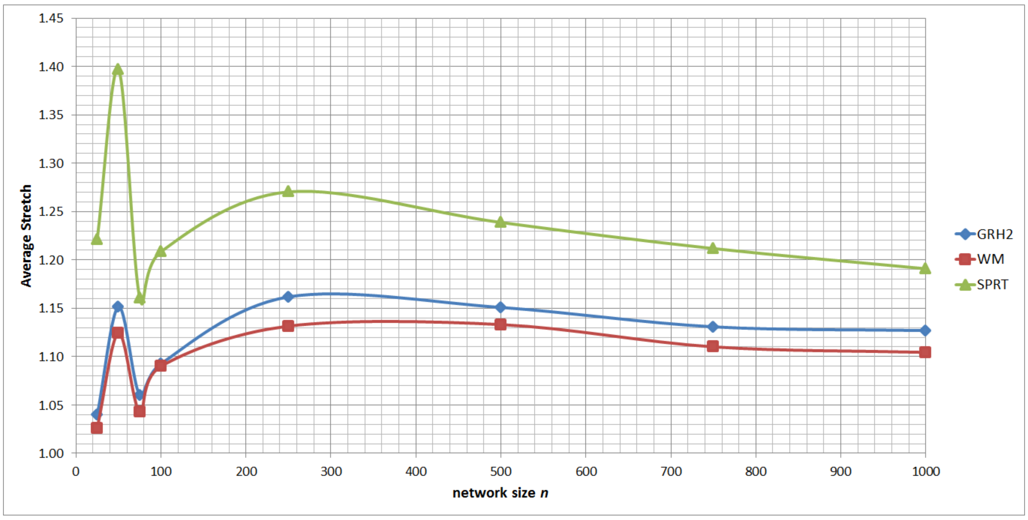


Figure 5.3: Average path stretch as a function of the number of nodes on BA graphs.

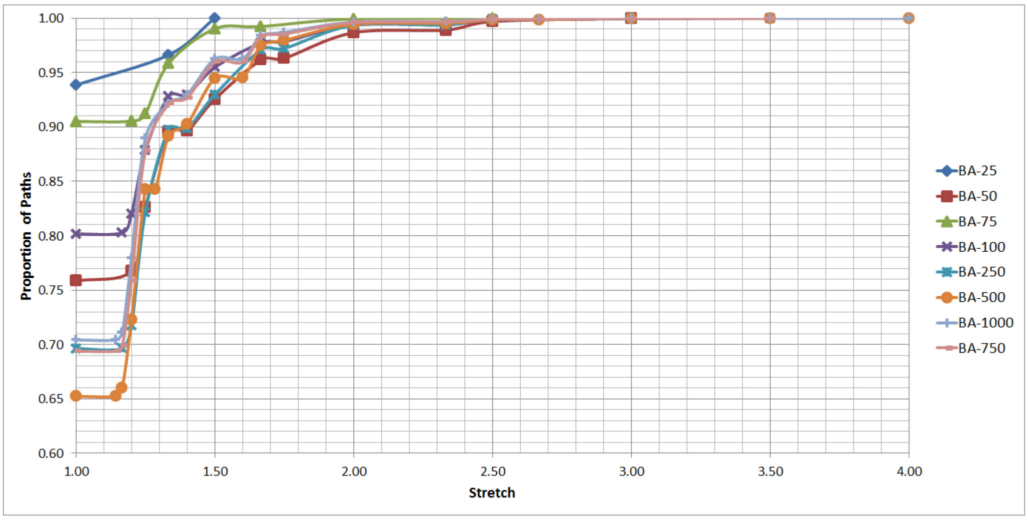


Figure 5.4: Stretch CDF for WM GGR scheme in several BA graphs.

.

Vertex label size and RT size

We have also analyzed the dependence of both the vertex label size and RT with respect to the number of network nodes in BA graphs for our WM scheme. In Figure 5.5 we show the maximum length (in bits) of the vertex labels obtained in the simulations compared with the theoretical upper bound. Note that Algorithm 1 produces labels of size $O(D(T_H) \cdot log(\Delta_H))$ by Theorem 4.2. In fact, the exact value of the vertex label size can be computed by $td(T_H) \cdot log(\Delta_H)$, where $td(T_H)$ is the tree depth of the computed spanning tree. The results show that the experimental values remain well below the theoretical ones in all the range of number of network nodes. Moreover our WM scheme achieves a succinct representation of labels since $log_2^2(n) \subseteq O\left(log^2(n)\right)$, i.e., the label vertex size grows poly-logarithmically in n. In addition, we show in Figure 5.6 that the RT grows sub-linearly in n (recall that a RT in a Shortest Path (SP) scheme grows linearly in n), while it remains below the theoretical upper bounds.

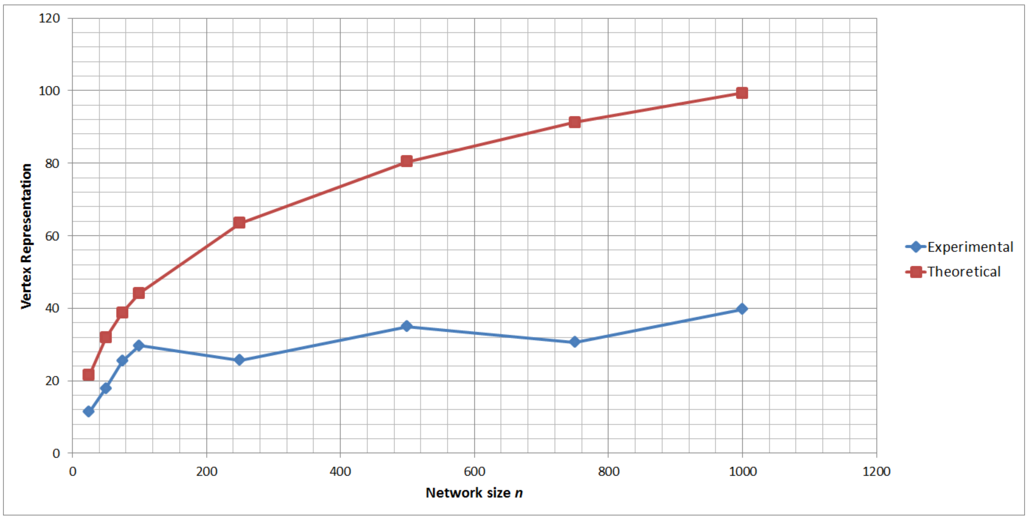


Figure 5.5: Theoretical upper bound vs experimental vertex label size on BA graphs.

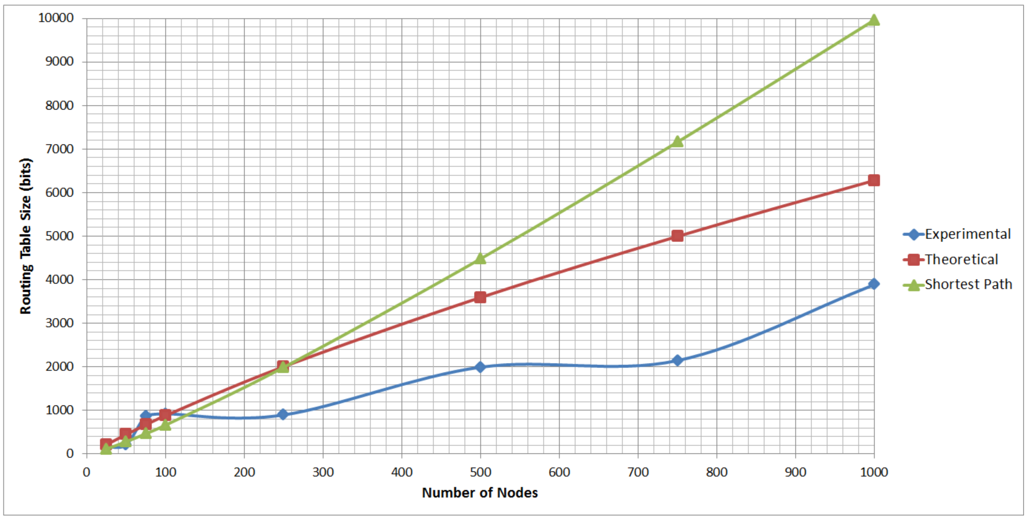


Figure 5.6: Theoretical upper bound vs experimental vs shortest path RT size on BA graphs.

5.1.4 Results in HK-based topologies

Stretch

In Figure 5.7 we show that the resulting CDF of the stretch for our WM scheme has a better performance than the other two evaluated schemes on a HK graph with 1000 nodes. The number of routes with stretch 1 is over the 70% in our scheme, which improves over the results of the other ones, and over the 91% of the routes have stretch ≤ 1.25. In fact, more than 99% of the computed routes have a stretch at most 2 in our scheme. The lowest stretch 3.5 was obtained by WM, while GRH2 and SPRT reached a stretch of 4.5 and 8, respectively.

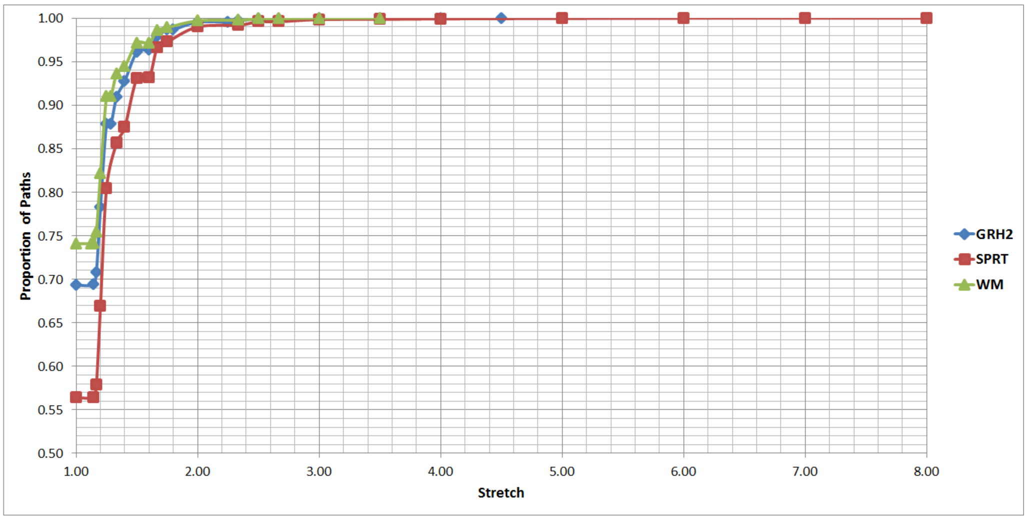


Figure 5.7: Stretch CDF for WM, GRH2 and SPRT on a HK graph with 1000 nodes.

Figure 5.8 and Figure 5.9 show the dependence of the stretch and the average path stretch values with respect to the number of network nodes in HK topologies for the three schemes. Table 5.8 shows that the 95% CI of the stretch remains below 15% of the mean in all cases, which ensures a high reliability of the computed stretch in this model topology. In both cases, the stretch and the average path stretch values are the lowest ones in WM. Compared with the theoretical upper bound, our scheme remains below it. In addition, we see that both values tend to be constant (or grow very slowly) when the number of vertices is incremented. In Figure 5.10 we show that the 90% of the routes obtained by our scheme in all the evaluated topologies have stretch ≤ 1.3 and over the 99% is less than 2.

	GRH2		WM		SPRT	
Vertices	**mean**	**CI (+/-)**	**mean**	**CI (+/-)**	**mean**	**CI (+/-)**
25.0	2.00	0.29	2.50	0.36	5.00	0.50
50.0	2.50	0.38	2.50	0.20	6.00	0.40
75.0	3.50	0.23	3.50	0.28	7.00	0.31
100.0	3.50	0.26	3.50	0.19	7.00	0.89
250.0	3.50	0.23	3.50	0.24	7.00	0.81
500.0	5.00	0.31	4.00	0.35	10.00	0.93
750.0	4.00	0.36	3.50	0.24	8.00	0.84
1000.0	4.50	0.21	3.50	0.24	8.00	0.94

Table 5.8: Mean and 95% CI of stretch in the HK topologies.

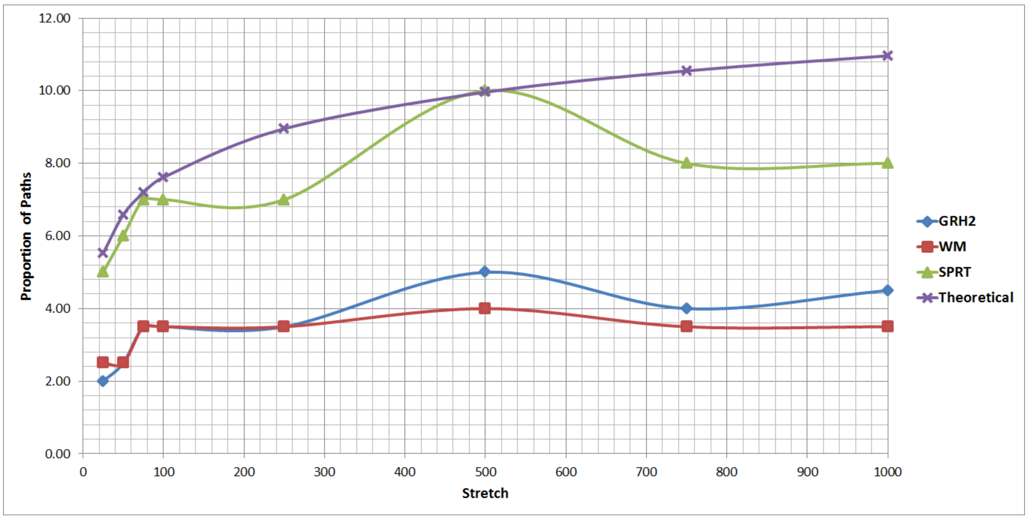


Figure 5.8: Stretch as a function of the number of nodes on HK graphs.

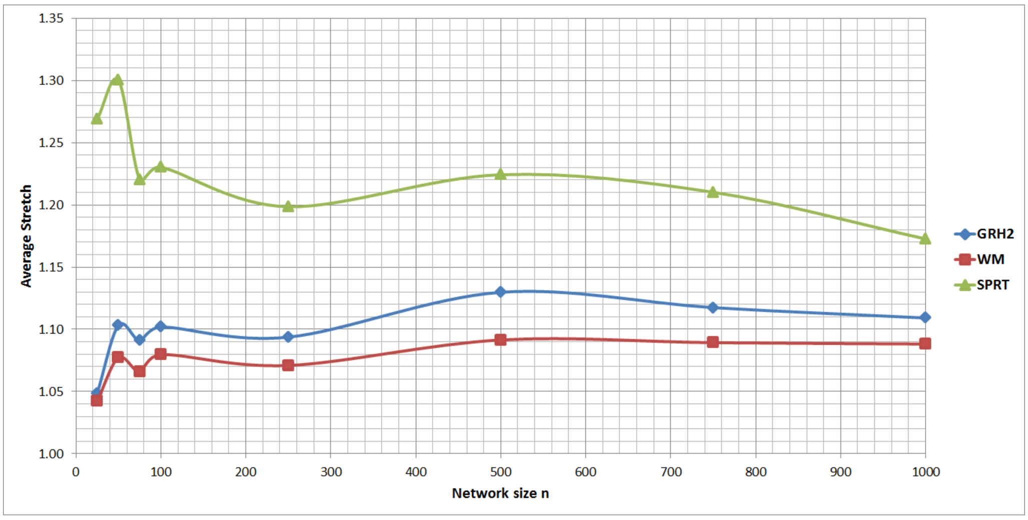


Figure 5.9: Average path stretch as a function of the number of nodes on HK graphs.

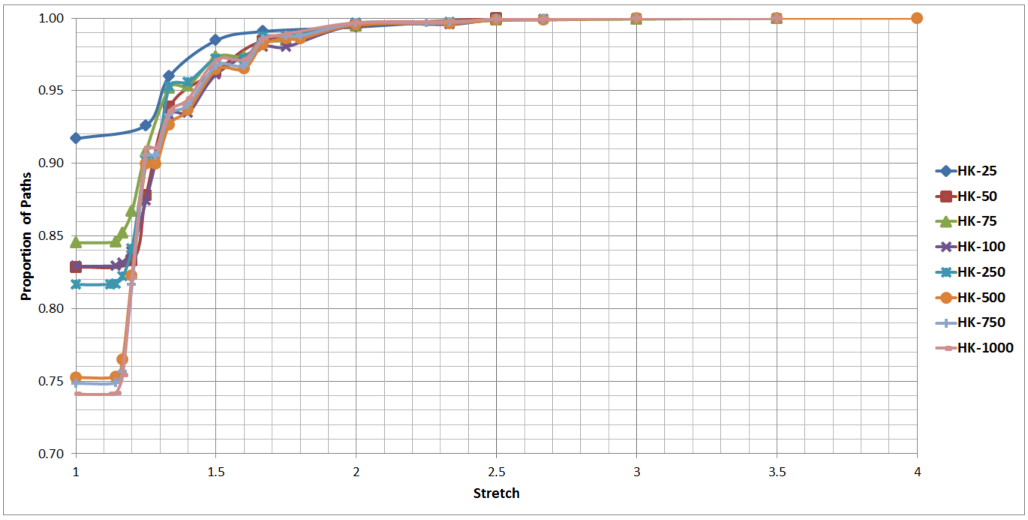


Figure 5.10: Stretch CDF for WM on several HK graphs.

.

Vertex label size and RT size

In Figure 5.11 we show the maximum length (in bits) of the vertex labels obtained by our WM scheme in the simulations compared with the theoretical upper bound, when the number of network nodes in the HK topologies is increased. The results show that the experimental values remain well below the theoretical ones in all the range of number of network nodes. Again, our WM scheme achieves a succinct representation of labels since $log_2^2(n) \subseteq O\left(log^2(n)\right)$. In Figure 5.12 we show that the RT grows sub-linearly in n, while it remains below the theoretical upper bounds.

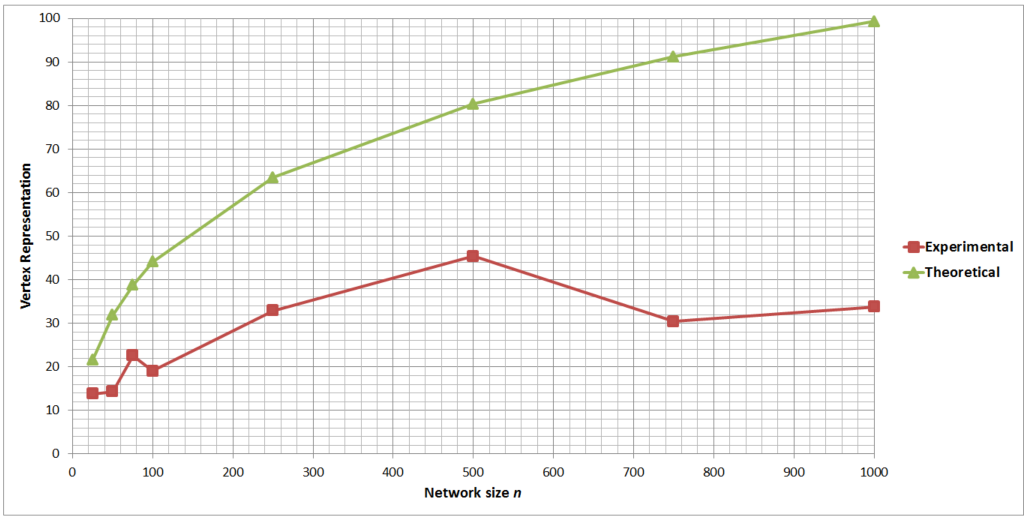


Figure 5.11: Theoretical upper bound vs experimental maximum vertex label size on HK graphs.

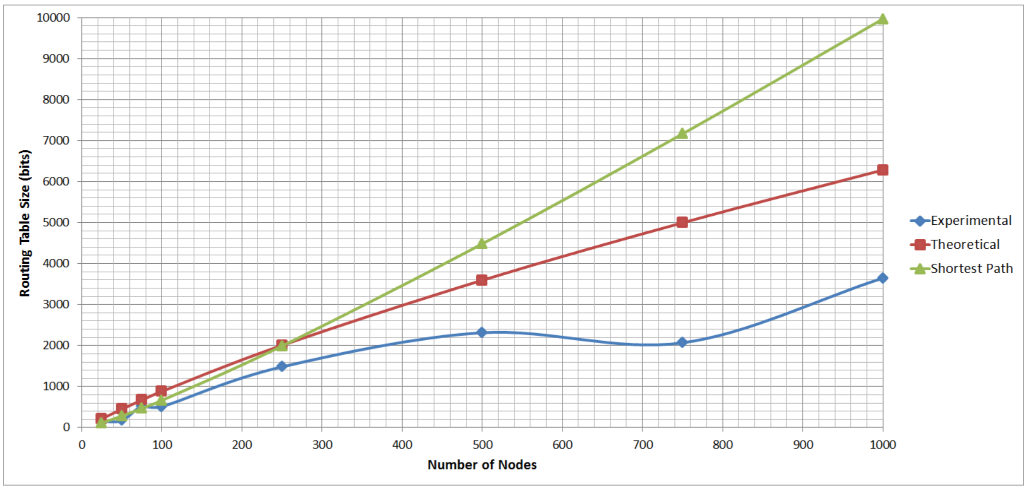


Figure 5.12: Theoretical upper bound vs experimental vs shortest path RT size on HK graphs.

5.1.5 Results in IR-level topologies

Stretch

In Figure 5.13 we show the CDF of the resulting stretch for our WM scheme and for GRH2 and SPRT schemes running on the IR level topology of the Internet AS 3257. As in the case of GRH2, the routes chosen by our WM have low stretch compared to SPRT. The number of routes with stretch 1 is the 82% in our scheme, over the 95% of the routes have stretch ≤ 1.25 and more than the 99% of them have an stretch ≤ 2. In addition, the stretch of our proposal is only 3.5 as for GRH2, while SPRT reaches a stretch of 7.

In Figure 5.14 we compare the CDF of the resulting stretch for our WM scheme in the topologies shown in Table 5.6. The 95% of the computed routes have a stretch ≤ 1.7, and more than the 99% of them have an stretch ≤ 2, in all the evaluated topologies. Finally, Table 5.6 presents the experimental value of the stretch, which remains below the theoretical upper bound of our scheme in scale-free networks.

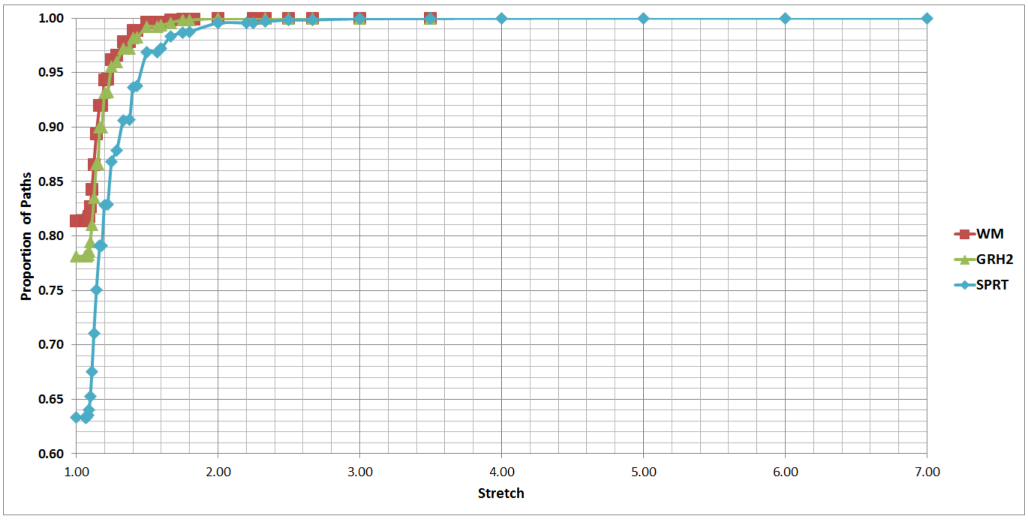


Figure 5.13: Stretch CDF for WM, GRH2 and SPRT on the AS 3257 (at IR-level topology).

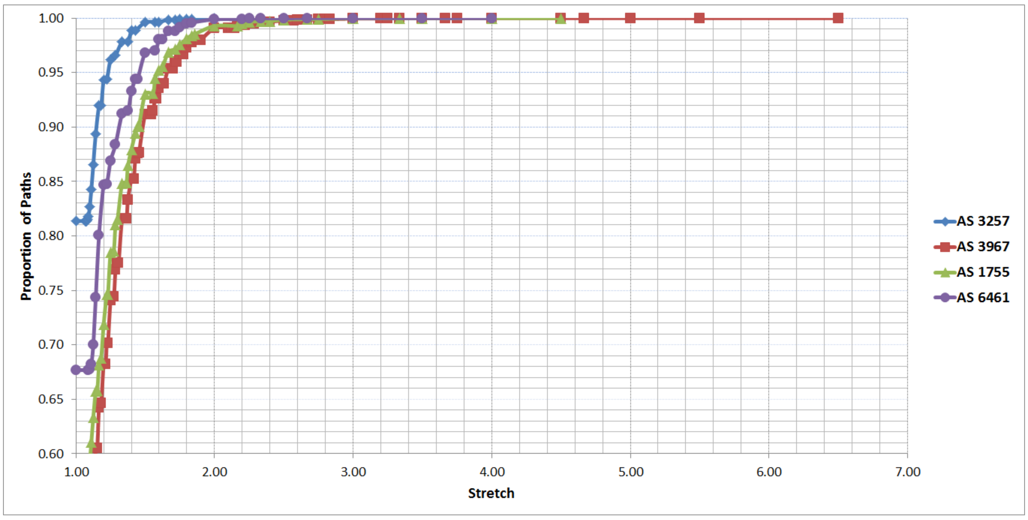


Figure 5.14: Stretch CDF for WM on several ASs (at IR-level topology).

In Table 5.9 we show the stretch with respect to the number of nodes in the IR level topologies for our scheme. The resulting stretch of our scheme WM is below the theoretical upper bound of $log_2(n) \subseteq O(log(n))$, in all the evaluated topologies.

Stretch		**AS**			
		3967	**3257**	**1755**	**6461**
Experimental	**Maximum**	6,50	3,50	4,50	4,00
	Average	1,18	1,04	1,15	1,10
Theoretical		9,90	10,26	10,53	11,41

Table 5.9: Theoretical vs experimental stretch in the evaluated IR level topologies.

Vertex label size RT size

In Table 5.10 we show the resulting vertex label size in the evaluated IR level topologies. Similarly to the synthetic topologies, the resulting vertex label of our scheme on real-world Internet topologies is succinct and it remains below theoretical upper bound. In addition, we show in Figure 5.11 that the RT grows sub-linearly in n, while it remains very close to the theoretical upper bounds. Therefore, these results together with the low stretch of our scheme prove that it can be seen as a CR scheme for Internet topologies.

	Vertex Label Size	
AS	**Experimental** $td(H) \cdot log_2(\Delta_H)$	**Theoretical** $log_2^2(n)$
1755	63	99
3257	72	106
3967	87	111
6461	96	131

Table 5.10: Experimental vs theoretical maximum vertex label size in the evaluated IR level topologies.

AS	Experimental	Theoretical	Shortest Path
	$\Delta_H \cdot td(H) \cdot log_2(\Delta_H)$	$2 \cdot n^{1/2} \cdot log_2^2(n)$	$n \cdot log(n)$
1755	2268	6126	9477
3257	6480	7414	12544
3967	8787	8541	15587
6461	15552	13665	31034

Table 5.11: Theoretical vs experimental vs Shortest Path RT size in the evaluated IR level topologies.

5.2 Evaluation of the C-GGR scheme for Cayley Graphs topologies

Section 4.4 showed that any network with underlying CG can use Algorithm 1 to perform a GGR scheme with shortest path guarantee. However, the required space for the CRS or any structure to compute the distance between vertices depends on the topological properties of the graph itself. This section evaluates the impact of memory consumption of such structures in some well-known families of CG.

5.2.1 Topologies

Among the many existing models for CGs (e.g., [6, 132–134]), we have selected the following six: Hypercube, Butterfly, Transposition, Bubble-Sort, Star and Pancake.

- **The Hypercube graph** $H(k)$ is the graph with vertex set $V(\Gamma) = \{x_1x_2...x_k : x \in \{0,1\}\}$. Two vertices $(a_1a_2...a_k)$ and $(b_1b_2...b_k)$ are adjacent if and only if $a_i = b_i$ for all but one i, $1 \leq i \leq k$. This graph is the CG on the group $\mathbb{Z}_k^2$ with generator set $S = \{(0,...,0,s_i,0,...,0) : s_i = 1, 1 \leq i \leq k\}$. It has 2^k vertices and both its diameter and degree are equal to k.

- **The Butterfly graph** $BF(k)$ is the CG with vertex set $V(\Gamma) = \mathbb{Z}_k \times \mathbb{Z}_k^2$. Any vertex $(i,x) \in V(\Gamma)$, where $x = (x_1x_2...x_k)$ and $1 \leq i \leq k$, is connected to the vertices $(i+1,x)$ and (i+1,x(i)) where x(i) denote the string which is derived from x by replacing x_i by $1-x_i$. All operations on i are made modulo n. This graph is isomorphic to the CG on the subgroup of the Symmetric Group Sym_{2k} generated by $S = \{(123...2k)^2, (123...2k)^2(12)\}$. It is a 4-regular graph of $n \cdot 2^k$ vertices and diameter $\lfloor 3(k)/2 \rfloor$.

- **The Transposition graph** $TP(k)$ on Sym_k has generation set $S = \{(i,j) \in Sym_k, 1 \leq i < j \leq k\}$, where (i,j) transposes the ith and jth elements of a permutation by right multiplication. This graph is a bipartite $\binom{k}{2}$-regular graph of order $k!$ and diameter $k-1$.

- **The Bubble-Sort graph** $BS(k)$ on Sym_k is generated by the set of transpositions $S = \{(i,i+1) \in Sym_k, 1 \leq i < n\}$, where $(i,i+1)$ interchanges the ith and $(i+1)$th elements of a permutation when multiplied on the right. The order of this graph is $k!$. It is also a bipartite $(k-1)$-regular and its diameter is $\binom{k}{2}$.

- **The Star graph** $ST(k)$ is the Cayley Graph on Sym_k with the generating set of transpositions $S = \{(1,i) \in Sym_k, 1 < i \leq k\}$, where $(1,i)$ interchanges the first and

ith elements of a permutation by right multiplication. This graph has $k!$ vertices, it is a bipartite $(k-1)$-regular with diameter $\lfloor 3(k-1)/2 \rfloor$.

- **The Pancake graph** $P(k)$ on Sym_k is generated by the set $S = \{r_i \in Sym_k, 2 \leq i \leq k\}$ for all prefix-reversal r_i that reverses the order of any sub-string [1,i], $2 \leq i \leq k$ of a permutation π by right multiplication. In other words, $[\pi_1 \dots \pi_i \pi_{i+1} \dots \pi_k] \cdot r_i$ = $[\pi_i \dots \pi_1 \pi_{i+1} \dots \pi_k]$. It is a $(k-1)$-regular graph with $k!$ vertices and satisfies $17k/16 \leq Diameter \leq (5k+5)/3$.

Note that for all cases, the number of vertices of the graphs depends on the value of k. So, for each model, we have generated six instances of graphs with different number of vertices for our analysis. For comparison purposes, the generated graphs have similar number of nodes in every of the six different instances. Therefore, there are six instances composed by one graph from each graph family. The values for k and the number of vertices for each one them in each family are presented in Table 5.12.

Cayley Graph	Instance	k	Number of Vertices
$H(k)$	1	6	64
	2	9	512
	3	10	1024
	4	12	4096
	5	16	65536
	6	18	262144
$BF(k)$	1	4	64
	2	6	384
	3	7	896
	4	9	4608
	5	12	49152
	6	14	229376
$TP(k), BS(k), ST(k), P(k)$	1	4	24
	2	5	120
	3	6	720
	4	7	5040
	5	8	40320
	6	9	362880

Table 5.12: The value of n and the resulting number of vertices for $H(k)$, $BF(k)$, $TP(k)$, $BS(k)$, $ST(k)$ and $P(k)$.

5.2.2 Measured parameters and metrics

In addition to the CRS, we have also evaluated experimentally the required space of the structures presented in sub-section 2.3.6, which can be used to compute distances in the word-metric of a CG. These structures are the following:

- CRS
- CRS+Index Automaton (IA)
- Word-Differences Automaton (WD)
- General-Multiplier Automaton (GM)

To compare the space requirements of the mentioned structures with respect to a RT generated by a SP Routing Scheme, it is necessary to define the size of an Equivalent Routing Table (ERT) for those structures. Since all structures (except CRS) are Finite State Automata (FSA), they can be stored as graphs. Therefore we consider for each FSA, the required space for its ERT is bounded by $O(t)$, where t is the number of states of such FSA. In general, for a network of size $|G| = |V(\Gamma)|$, the space complexity is $O(t \cdot |V(\Gamma)|)$. In the case of CRS, we also define its ERT as $O(t)$, where t is the number of equations of the CRS.

We compare the ERTs size of our WM scheme with the RT size of a general-purpose shortest-path routing scheme for CGs presented in [116]. The space complexity of this algorithm is bounded by $O(|V(\Gamma)|^2)$. Based on it, we define the metric Reducing Table Ratio (RTR) as $RTR = \frac{O(t \cdot |V(\Gamma)|)}{O(|V(\Gamma)|^2)} = \frac{O(t)}{O(|V(\Gamma)|)}$ to evaluate the efficiency of the implemented structures. If $cp \geq 1$, it indicates that the ERT consumes equal or more space than the one generated by the scheme used for comparison. Otherwise, the evaluated structure is more space-efficient.

5.2.3 Results

In Figures 5.15 to 5.20 we compare the RTR metric for different structures and different models (increasing in sizes) in log_2-scale. The $H(k)$, $BS(k)$ and $TP(k)$ models present the best RTR metric. Their RTR have an exponential decrease (tending to zero) in all the instances of such models. The only exception is the GM structure that has a RTR over 1 in the first instance of both $BS(k)$ and $TP(k)$ models. On the other hand, $BF(k)$, $ST(k)$ and $P(k)$ models have a slow decrease in the RTR value. However, after the second instance, the $BF(k)$ and $ST(k)$ models obtain a RTR less than 1 in all the structures, except the GM one. In general, the $P(k)$ model has the worst behavior with RTR over 1 in all the structures with the exception of its WD.

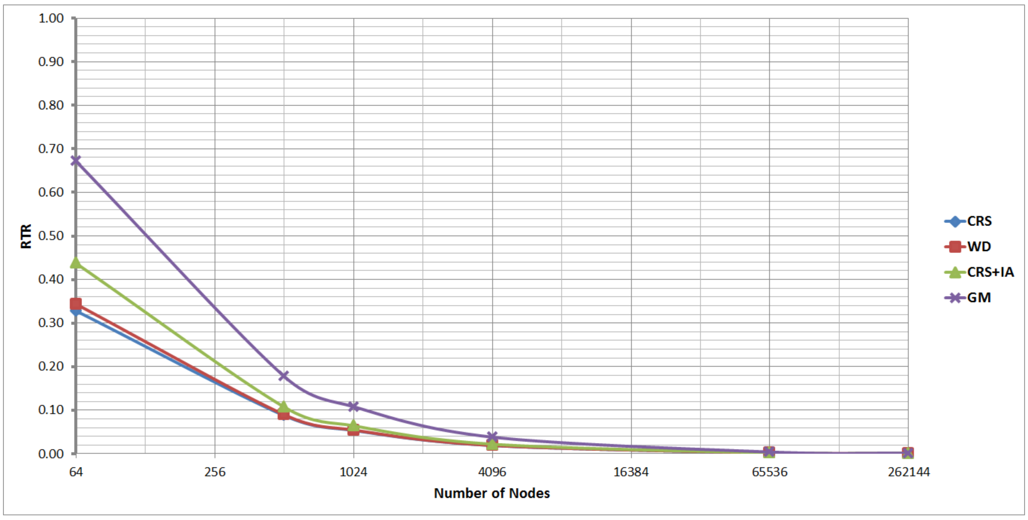


Figure 5.15: RTR metric in the Hypercube graph family.

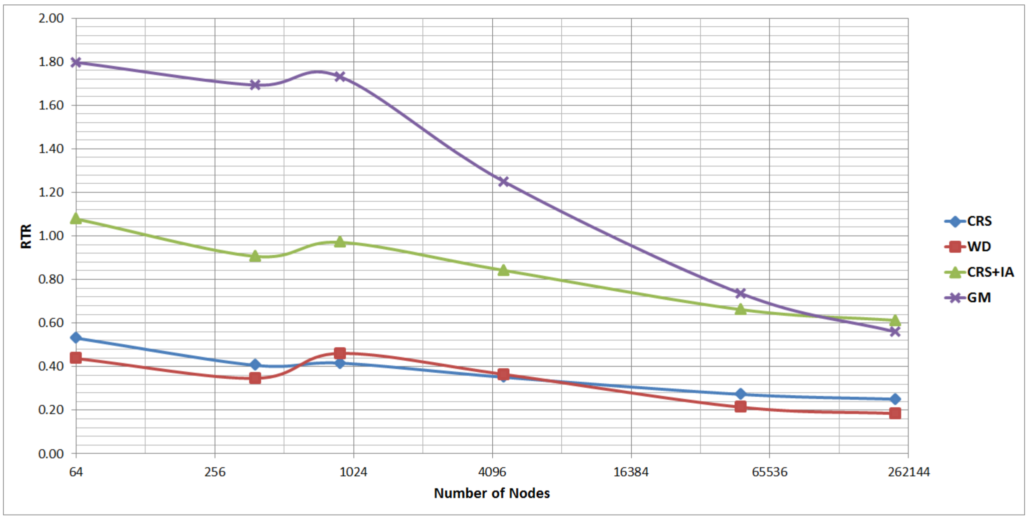


Figure 5.16: RTR metric in the Butterfly graph family.

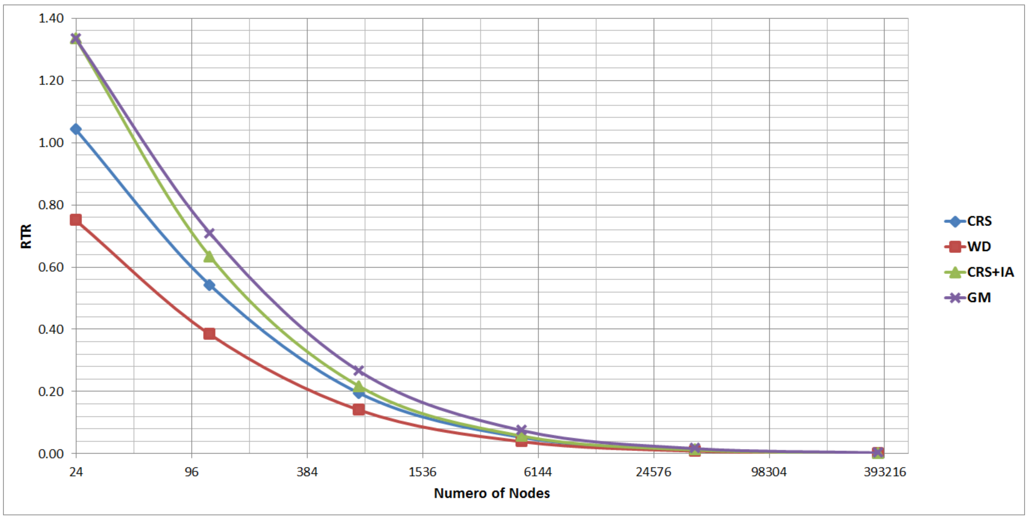


Figure 5.17: RTR metric in the Transposition graph family.

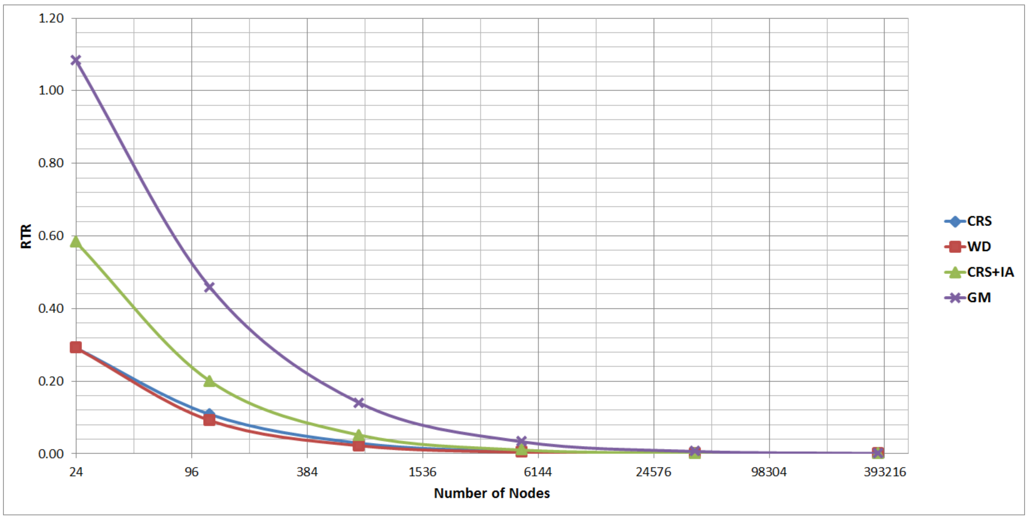


Figure 5.18: RTR metric in the Bubble-Sort graph family.

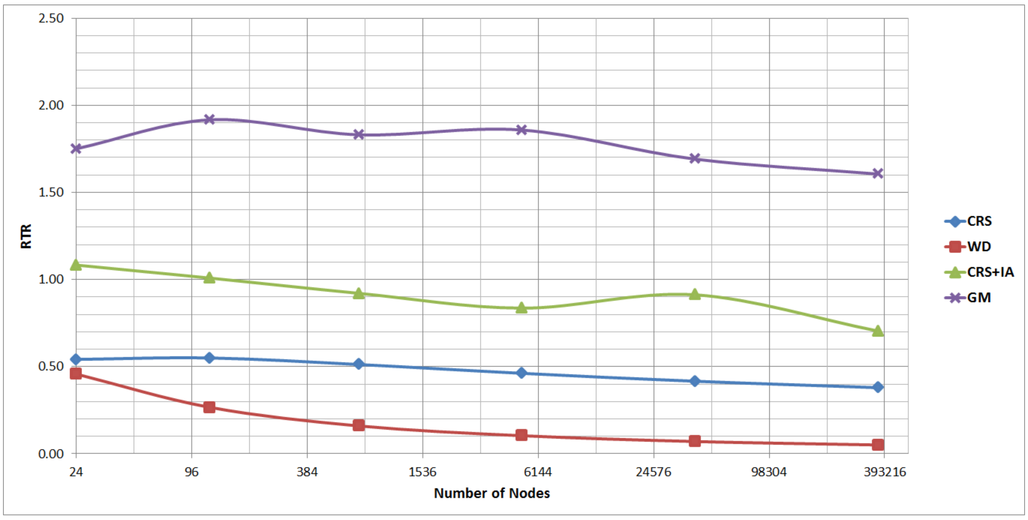


Figure 5.19: RTR metric in the Star graph family.

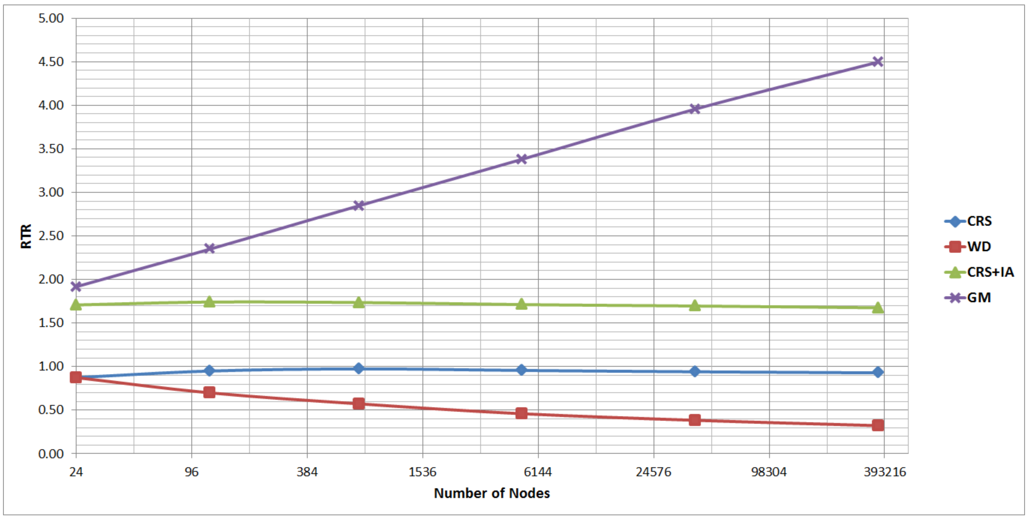


Figure 5.20: RTR metric in the Pancake graph family.

Among the evaluated structures, the WD has the smallest size. Figure 5.21 shows the RTR of such structure for each CG model, when the number of vertices increases. In all the instances, this combination performs better in terms of space requirements compared with the scheme used for comparison. Note that the $BF(k)$ and $ST(k)$ models obtain a fast decrease of the RTR with this structure. GM structure will have more states than WD structure because it is more complex, which is a consequence of the Shortlex Automatic Structure (SAS) definition.

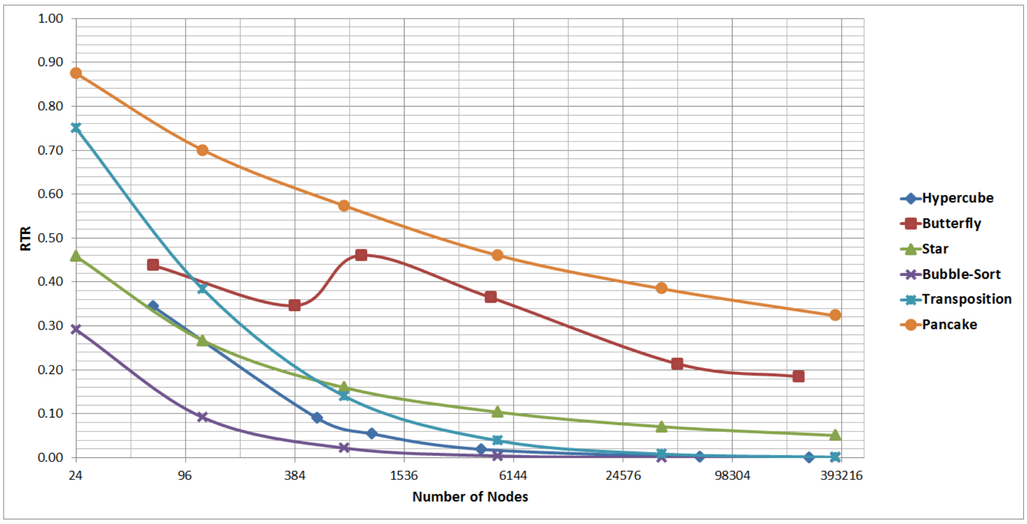


Figure 5.21: RTR of the WD in each family of graph.

We have also analyzed the fast decrease of the RTR metric of $H(k)$, $BS(k)$ and $TP(k)$ models with respect to the other three CG models. By analyzing their respective SAS (see [40], Lemma 2.3.2), we saw that the exponential decrease of the size of its structures is because the *k-fellow-traveler* property. Figure 5.22 shows the value of k for all the instances of each family of graphs. For $H(k)$, $BS(k)$ and $TP(k)$ the value of k is constant in all their instances. It means that their ERT always have an entry, or equivalently an state in the FSA, for each vertex in the ball around the identity vertex with constant ratio k. On the contrary, the value of k for the $BF(k)$, $ST(k)$ and $P(k)$ models is not constant and it depends on the number of nodes. Therefore, the SAS size (and the size of their intermediate structures) also increases with k.

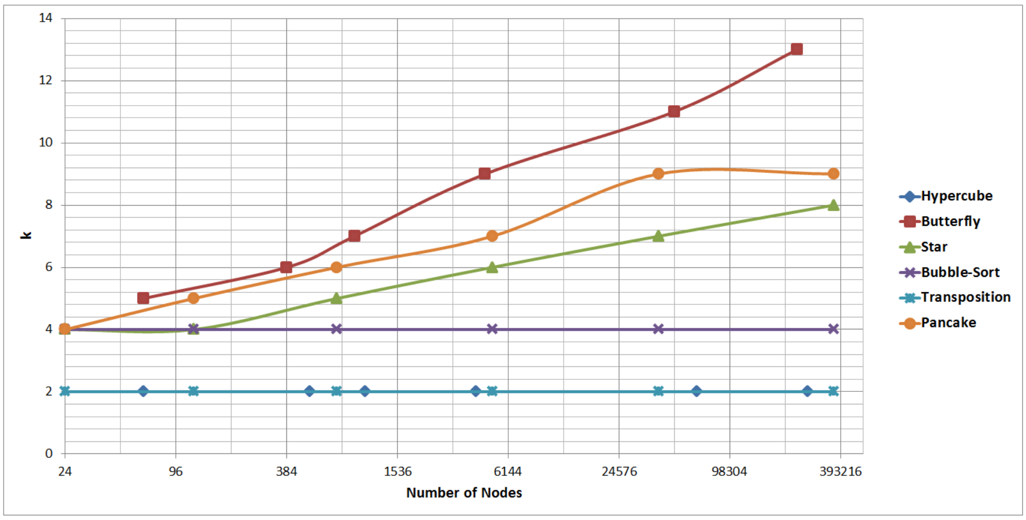


Figure 5.22: The k-fellow property of each family of graphs.

5.3 Summary and conclusions

We have presented and discussed the experimental evaluation through simulation of our two specialized schemes, the C-GGR scheme for scale-free networks and the C-GGR scheme for CG topologies.

In the experiments for scale-free graphs we have measured the stretch, the RT size and the vertex label size. The performance results of the C-GGR schemes for scale-free graphs on two synthetic networks (the Barabasi-Albert and the Holme-Kim models) and one real-world network (IR level topologies) have shown that the three metrics remain well below the theoretical upper bounds. The stretch obtained by our scheme grows very slow (less than $log(n)$) and improves over the results obtained by the Greedy Geometric Routing in $\mathbb{H}^2$ scheme proposed in [32] and the traditional Shortest Path Routing on Trees scheme. The vertex label size shows a logarithmic growth while the RT size grows sub-linearly in the number of nodes of the network. In other words, we have confirmed experimentally it is an specialized CR scheme for scale free graphs (which includes Internet-like networks).

In the experiments for CG-based Data Center topologies we have measured the space complexity (the size of the CRS and other structures) and we have compared it with other structures that allow the distance computation in CGs models. The WD structure is the one with less space complexity and its size is smaller than a RT computed by a shortest path routing scheme for the same model, in all the performed experiments. We have observed the fast decrease of the size of such structure in several of the CG models, and we have seen that the exponential decrease is caused by the *k-fellow traveler* property, which is an intrinsic topological property of each CG model. We have also shown that for a lower or constant value of k, the size of the CRS and the SAS (and the size of their intermediate structures) is very small compared to the RT itself or any RT constructed by a shortest path routing scheme.

Chapter 6

Conclusions and future work

In this chapter we summarize the main contributions of this work and point out possible directions for future research.

6.1 Summary and conclusions

This thesis explored the use of the Greedy Geometric Routing (GGR) schemes to solve the Compact Routing Problem (CRP) in Internet-like networks and several families of Data Centers (DCs) architectures. Our main objective was to design GGR schemes that have low time and space complexity, and that achieve routes as close as possible to the shortest ones (low stretch), suitable for general graphs and also specialized for these two types of networks. We have detected that the GGR schemes proposed so far experience one or more of the following problems: 1) they do not guarantee packet delivery, 2) they produce vertex labels of size linear (or higher) in n, 3) they can not be implemented in a distributed way, 4) they require a full knowledge of the network topology, or 5) they have unbounded stretch.

GGR schemes are composed by two procedures, the vertex labeling, through the embedding of the graph in a metric space, and the packet forwarding based on the Greedy Forwarding (GF) strategy, which selects the closest neighbor to the destination. The GF procedure requires a mechanism to determine the distance between vertices in the metric space of the embedding. In order to design GGR schemes with low time and space complexity in both procedures and that do not experience the problems mentioned above, we have proposed a simple and novel embedding of any finite connected graph into a Word-Metric (WM) space (i.e., a metric space generated by algebraic groups). Then we have proved that any GGR scheme built on top of this embedding guarantees the packet delivery (i.e., the embedding is greedy).

We have proposed three GGR schemes by combining word processing in algebraic groups with graph search algorithms: a GGR scheme that works on any kind of graph, whose computational complexity depends on the parameters $D(T_H)$ and Δ_H, and two specialized Compact Greedy Geometric Routing (C-GGR) schemes for Internet-like networks and several families of DCs architectures, whose computational complexity only depends on n. Specifically, we have proposed the following three GGR schemes:

- A GGR scheme for any kind of graph, with stretch of $O(D(T_H))$, $O(D(T_H) \cdot log(\Delta_H))$ bits per vertex label, Routing Tables (RTs) of size $O(\Delta_H \cdot D(T_H) \cdot log(\Delta_H))$ and routing decisions that take $O\left(\Delta_H^2 \cdot D(T_H)\right)$ steps.
- A C-GGR scheme for scale-free graphs (which include many real-world topologies such as Internet), with stretch of $O(log(n))$, $O(log^2(n))$ bits per vertex label, RTs of size $O(n^{1/2} \cdot log^2(n))$ and routing decisions that take $O(n \cdot log(n))$ steps.
- A C-GGR scheme for Cayley Graphs (which are used as a model for Data Center interconnection networks), with shortest paths, $O(log(n) \cdot log(log(n)))$ bits per vertex label, routing tables of size $O(log^2(n) \cdot log(log(n)))$, and routing decisions that take $O\left(log^3(n)\right)$ steps.

The main advantage of the GGR scheme for any kind of graph is its simplicity. However, since this scheme depends on topological parameters, its performance is limited by the input graph itself. In the worst-case, the complexity of the vertex label and the RT size is similar to the complexity of either the point-to-point or the source routing scheme with shortest paths, while the stretch grows linearly in n. On the other hand, the two C-GGR schemes solve the limitation of the first scheme by exploiting the topological properties of several families of networks.

The Internet topology at the level of either Internet Router (IR) or Autonomous Systems (ASs), can be modelled as a scale-free network (low diameter and power-law vertex degree distribution), while the family of DCs based on Cayley Graph (CG) has low diameter and constant degree. In order to exploit such topological properties, we have shown that the construction of the spanning tree T_H plays a fundamental role in the complexity and performance of these schemes. We have studied the following issues with respect to the spanning tree computation:

1. How the selection of the vertex root would affect the structure of the resulting spanning tree.
2. How the maximum vertex degree would affect the complexity of the label description and RT size.

3. How the exploration strategy to create such spanning tree, specifically the diameter of the resulting spanning tree, would affect the stretch of the resulting routing scheme.

We have shown that the selection of the root has a minimum impact in the structure of the computed spanning tree, and that the resulting stretch of the greedy embedding in tree metrics is almost independent of the degree of such root. Therefore, the main parameter to be considered for selecting a spanning tree is the tree depth. We have analyzed the time and space complexity of several spanning tree construction algorithms. We have shown that the Breadth-First Search (BFS) algorithm maintains the topological properties (degree and diameter), thus enabling the specialization of the routing schemes, and moreover, it has low time and space complexity. As a result of combining the topological properties of Internet-like networks and the family of DCs based on CG, with the BFS algorithm, we have constructed two GGR schemes built on top of the proposed embedding that are scalable and distributed. The complexity analysis of both C-GGR schemes have shown that our scheme do not experience the five problems previously mentioned. However, the cost our schemes must pay is that the routing decision time grows linearly with respect to n, for Internet-like networks, and polylog in n for CG-based DCs.

In the experimental evaluation through simulation of our two specialized schemes, the main goal has been to assess how far the experimental results were from the theoretical upper bounds and the comparison with other related works. For Internet-like graphs we have measured the stretch, the RT size and the vertex label size, on two synthetic networks (the Barabasi-Albert and the Holme-Kim models) and one real-world network (IR level topologies). We have shown that the three metrics remain well below the theoretical upper bounds. The stretch obtained by our scheme grows very slow (less than $log_2(n)$) and overcomes the results obtained by the Greedy Geometric Routing in $\mathbb{H}^2$ [32] and the traditional Shortest Path Routing on Trees (SPRT) scheme. We have observed that almost the 70% of the routes computed by our scheme have a stretch 1, and almost the 99% have a stretch lesser than 2, in all the evaluated topologies. The vertex label size have shown a logarithmic growth while the RT size grows sub-linearly in n. In other words, our proposed C-GGR scheme solves the CRP for Internet-like networks.

In the experiments for CG-based DC topologies we have measured the space complexity (the size of the Confluent Rewriting System (CRS) and other structures) and we have compared it with other structures that allow the distance computation in CGs models. The Word-Differences Automaton (WD) structure is the one with less space complexity and its size is smaller than the size of the RT computed by a shortest path routing scheme for the same model, in all the performed experiments. We have observed the fast decrease of the size of such structure with respect to n in several CG models, and we have seen that the exponential decrease is caused by the *k-fellow traveler* property, which is an intrinsic topological property of each CG model. We have also shown that for a lower or constant value of k, the size of the CRS and the Shortlex Automatic Structure (SAS) (and the size of their intermediate structures) is very small compared to the size of the RT itself or any RT constructed by a shortest path routing scheme. In conclusion, our proposed GGR scheme guarantees the shortest paths and solves the CRP for any CG-based DC.

6.2 Future work

There are several future research lines in which our research work on GGR schemes in Word-Metric spaces can be continued in the future:

- Designing GGR schemes on top of the proposed greedy embedding specialized for other families of graphs.
- Reducing the complexity of the routing decision time by reducing the time complexity of the distance computation using CRS or other structures that solve the minimum-length word problem (MWP) more efficiently and with a space complexity up to the size of the resulting RTs.
- Designing a C-GGR scheme on top of the proposed greedy embedding for any graph and comparing its performance with other universal Compact Routing (CR) schemes and C-GGR schemes.
- Analysing the behavior and effects of network dynamics in the proposed schemes and proposing strategies to mitigate such effects.
- Studying other algebraic groups and their WM spaces such that the properties of small-world effect and the power-law vertex degree distributions of scale-free graphs can be exploited more effectively.
- Studying other automatic groups, their topological properties and its automatic structures in order to design new CG-based DCs with high efficient and scalable routing schemes.

Bibliography

[1] R. Pastor-Satorras and A. Vespignani. *Evolution and Structure of the Internet: A Statistical Physics Approach*. New York, NY, USA: Cambridge University Press, 2004. ISBN: 0521826985.

[2] M. Faloutsos, P. Faloutsos, and C. Faloutsos. "On Power-law Relationships of the Internet Topology". In: *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*. SIGCOMM '99. Cambridge, Massachusetts, USA: ACM, 1999, pp. 251–262. ISBN: 1-58113-135-6.

[3] Y. Rekhter, T. Li, and S. Hares. *RFC 4271: A Border Gateway Protocol 4 (BGP-4)*. Tech. rep. IETF, 2006.

[4] D. Krioukov, K. Fall, and A. Brady. "On Compact Routing for the Internet". In: *SIGCOMM Comput. Commun. Rev.* 37.3 (July 2007), pp. 41–52. ISSN: 0146-4833.

[5] M. Thorup and U. Zwick. "Compact Routing Schemes". In: *Proceedings of the Thirteenth Annual ACM Symposium on Parallel Algorithms and Architectures*. SPAA '01. Crete Island, Greece: ACM, 2001, pp. 1–10. ISBN: 1-58113-409-6.

[6] M.-C. Heydemann. "Cayley graphs and interconnection networks". English. In: *Graph Symmetry*. Ed. by G. Hahn and G. Sabidussi. Vol. 497. NATO ASI Series. Springer Netherlands, 1997, pp. 167–224. ISBN: 978-90-481-4885-1.

[7] W. Xiao, H. Liang, and B. Parhami. "A Class Of Data-Center Network Models Offering Symmetry, Scalability, and Reliability". In: *Parallel Processing Letters* 22.04 (2012).

[8] M. Al-Fares, A. Loukissas, and A. Vahdat. "A scalable, commodity data center network architecture". In: *ACM SIGCOMM Computer Communication Review*. Vol. 38. 4. ACM. 2008, pp. 63–74.

[9] C. Guo et al. “BCube: a high performance, server-centric network architecture for modular data centers”. In: *ACM SIGCOMM Computer Communication Review* 39.4 (2009), pp. 63–74.

[10] G. Stamoulis and J. Tsitsiklis. “The efficiency of greedy routing in hypercubes and butterflies”. In: *Communications, IEEE Transactions on* 42.11 (Nov. 1994), pp. 3051–3061. ISSN: 0090-6778.

[11] J. Snyder. *Microsoft: Datacenter Growth Defies Mooreś Law*. 2007. URL: `http://www.pcworld.com/article/130921/article.html`.

[12] *2013 Visual Networking Index*. Sept. 2014. URL: `http://www.cisco.com/web/solutions/sp/vni/vni_forecast_highlights/index.html`.

[13] *Cisco Cloud Computing - Data Center Strategy, Architecture, and Solutions*. Tech. rep. Cisco System Inc., 2009.

[14] K. Bilal, S. Khan, and A. Zomaya. “Green Data Center Networks: Challenges and Opportunities”. In: *Frontiers of Information Technology (FIT), 2013 11th International Conference on*. Dec. 2013, pp. 229–234.

[15] *EULER FP7 project website*. Jan. 2014. URL: `http://www.euler-fireproject.eu/`.

[16] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. “Geometric Ad-hoc Routing: Of Theory and Practice”. In: *Proceedings of the Twenty-second Annual Symposium on Principles of Distributed Computing*. PODC ’03. Boston, Massachusetts: ACM, 2003, pp. 63–72. ISBN: 1-58113-708-7.

[17] S. Ruhrup. “Theory and practice of geographic routing”. In: *Ad Hoc and Sensor Wireless Networks: Architectures, Algorithms and Protocols* (2009), p. 69.

[18] T. Leighton and A. Moitra. “Some Results on Greedy Embeddings in Metric Spaces”. English. In: *Discrete & Computational Geometry* 44.3 (2010), pp. 686–705. ISSN: 0179-5376.

[19] B. Karp and H. T. Kung. “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks”. In: *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*. MobiCom ’00. Boston, Massachusetts, USA: ACM, 2000, pp. 243–254. ISBN: 1-58113-197-6.

[20] D. Eppstein and M. Goodrich. “Succinct Greedy Geometric Routing Using Hyperbolic Geometry”. In: *Computers, IEEE Transactions on* 60.11 (Nov. 2011), pp. 1571–1580. ISSN: 0018-9340.

[21] J. Fournier. *Graphs Theory and Applications: With Exercises and Problems*. ISTE Ltd. John Wiley & Sons, Inc., 2009. ISBN: 9781848210707.

[22] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. New York: Elsevier Science Ltd., 1976. ISBN: 0444194517.

[23] R. Diestel. *Graph Theory*. 3rd, electronic. Graduate Texts in Mathematics. Springer, 2005. ISBN: 9783540261827.

[24] R. Sedgewick and P. Flajolet. *An Introduction to the Analysis of Algorithms*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1996. ISBN: 0-201-40009-X.

[25] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Third Edition*. 3rd. The MIT Press, 2009. ISBN: 9780262033848.

[26] M. Bridson and A. Häfliger. *Metric Spaces of Non-Positive Curvature*. Die Grundlehren der mathematischen Wissenschaften in Einzeldarstellungen. Springer, 1999, 1999. ISBN: 9783540643241.

[27] J. L. Gross and T. W. Tucker. *Topological Graph Theory*. New York, NY, USA: Wiley-Interscience, 1987. ISBN: 0-471-04926-3.

[28] J. Saxe. "Embeddability of weighted graphs in k-space is strongly NP-Hard". In: *Proceedings of the 17th Allerton Conference in Communications, Control, and Computing*. 1979, pp. 480–489.

[29] M. Badoiu et al. "Ordinal Embedding: Approximation Algorithms and Dimensionality Reduction". In: *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*. Ed. by A. Goel, K. Jansen, J. D. Rolim, and R. Rubinfeld. Vol. 5171. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, pp. 21–34. ISBN: 978-3-540-85362-6.

[30] A. M.-C. So. "A Semidefinite Programming Approach to the Graph Realization Problem: Theory, Applications and Extensions". PhD thesis. 2007.

[31] C. H. Papadimitriou and D. Ratajczak. "On a conjecture related to geometric routing". In: *Theoretical Computer Science* 344.1 (Nov. 2005), pp. 3–14. ISSN: 03043975.

[32] R. Kleinberg. "Geographic Routing Using Hyperbolic Space". In: *26th IEEE International Conference on Computer Communications (INFOCOM 2007)*. May 2007, pp. 1902–1909.

[33] A. Hofer, S. Roos, and T. Strufe. "Greedy Embedding, Routing and Content Addressing for Darknets". In: *Networked Systems (NetSys), 2013 Conference on*. Mar. 2013, pp. 43–50.

[34] P. Maymounkov. *Greedy Embeddings, Trees, and Euclidean vs Lobachevsky Geometry*. Tech. rep. Massachusetts Institute of Technology, 2006, p. 20.

[35] A. Cayley. "On the Theory of Groups". In: *American Journal of Mathematics* 11.2 (1889), pp. 139–157. ISSN: 00029327.

[36] W. Magnus, A. Karrass, and D. Solitar. *Combinatorial Group Theory: Presentations of groups in terms of generators and relations*. Courier Dover Publications, 2004.

[37] J. Rotman. *An introduction to the theory of groups*. 4th ed. Springer, 1995. ISBN: 3-540-94285-8.

[38] E. Ghys, A. Verjovsky, and A. Hafliger. *Group Theory from a Geometrical Viewpoint*. Die Grundlehren der mathematischen Wissenschaften in Einzeldarstellungen. World Scientific Publishing, 1999. ISBN: 9810214308.

[39] J. Meier. *Groups, graphs and trees: an introduction to the geometry of infinite groups*. Cambridge University Press, 2008.

[40] D. B. A. Epstein et al. *Word Processing in Groups*. Natick, MA, USA: A. K. Peters, Ltd., 1992. ISBN: 0867202440.

[41] D. F. Holt, B. Eick, and E. A. O'Brien. *Handbook of computational group theory*. Discrete mathematics and its applications. Boca Raton: Chapman & Hall/CRC, 2005. ISBN: 1-584-88372-3.

[42] C. C. Sims. "Computer Algebra Handbook". In: ed. by J. Grabmeier, E. Kaltofen, and V. Weispfenning. Springer Berlin Heidelberg, 2003. Chap. Computational Group Theory, pp. 65–83. ISBN: 978-3-642-55826-9.

[43] M. Dehn. "On infinite discontinuous groups (Uber unendliche diskontinuierliche Gruppen)." In: *Mathematische Annalen* 71.1 (1912), pp. 116–144. ISSN: 0025-5831.

[44] W. W. B. G. Baumslag and B. H. Neumann. "Some unsolvable problems about elements and subgroups of groups". In: *Mathematica Scandinavica* 7 (1959), pp. 191–201.

[45] S. Even and O. Goldreich. "The minimum-length generator sequence problem is NP-hard". In: *Journal of Algorithms* 2.3 (1981), pp. 311–313. ISSN: 0196-6774.

[46] D. Epstein, D. Holt, and S. Rees. "The use of Knuth-Bendix methods to solve the word problem in automatic groups". In: *Journal of Symbolic Computation* 12.4 (1991), pp. 397–414. ISSN: 0747-7171.

[47] P. E. S. Roger C. Lyndon. “Combinatorial Group Theory”. English. In: *The Mathematical Intelligencer* 2.1 (1979), pp. 42–42. ISSN: 0343-6993.

[48] D. Holt. *KBMAG Package: A Knuth-Bendix on Monoids, and Automatic Groups*. Jan. 2014. URL: `http://homepages.warwick.ac.uk/~mareg/kbmag/`.

[49] A. Williams. *Monoid automata factory*. Jan. 2014. URL: `http://maffsa.sourceforge.net/manpages/MAF.html`.

[50] D. Knuth and P. Bendix. “Simple Word Problems in Universal Algebras”. English. In: *Automation of Reasoning*. Ed. by J. Siekmann and G. Wrightson. Symbolic Computation. Springer Berlin, 1983, pp. 342–376. ISBN: 978-3-642-81957-5.

[51] F. Otto and Y. Kobayashi. “Properties of Monoids That Are Presented by Finite Convergent String-Rewriting Systems - A Survey”. English. In: *Advances in Algorithms, Languages, and Complexity*. Ed. by D.-Z. Du and K.-I. Ko. Springer US, 1997, pp. 225–266. ISBN: 978-1-4613-3396-8.

[52] C. Brown, G. Cooperman, and L. Finkelstein. “Solving permutation problems using rewriting systems”. In: *Symbolic and Algebraic Computation*. Ed. by P. Gianni. Vol. 358. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1989, pp. 364–377. ISBN: 978-3-540-51084-0.

[53] D. F. Holt. “The Warwick Automatic Groups Software”. In: *Geometric and Computational Perspectives on Infinite Groups, volume 25 of Amer. Math. Soc. DIMACS Series*. 1994, pp. 69–82.

[54] C. C. Sims. *Computation with Finitely Presented Groups*. Cambridge University Press, 1994.

[55] C. Hedrick. *RFC 1058: Routing Information Protocol*. Tech. rep. IETF, 1988.

[56] J. Moy. *RFC 2328: OSPF Version 2*. Tech. rep. IETF, 1998.

[57] D. Johnson, Y. Hu, and D. Maltz. *RFC 4728: The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4*. Tech. rep. IETF, 2007.

[58] I. Abraham et al. “Compact Name-independent Routing with Minimum Stretch”. In: *Proc. 16th Ann. ACM Symp. Parallelism in Algorithms and Architectures*. ACM, 2004.

[59] L. J. Cowen. “Compact Routing with Minimum Stretch”. In: *Journal of Algorithms* 38.1 (2001), pp. 170–183. ISSN: 0196-6774.

[60] M. Breuer. “Coding the vertexes of a graph”. In: *Information Theory, IEEE Transactions on* 12.2 (Apr. 1966), pp. 148–153. ISSN: 0018-9448.

[61] S. Kannan, M. Naor, and S. Rudich. “Implicit Representation of Graphs”. In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*. STOC ’88. Chicago, Illinois, USA: ACM, 1988, pp. 334–343. ISBN: 0-89791-264-0.

[62] D. Peleg. “Proximity-Preserving Labeling Schemes and Their Applications”. English. In: *Graph-Theoretic Concepts in Computer Science*. Ed. by P. Widmayer, G. Neyer, and S. Eidenbenz. Vol. 1665. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1999, pp. 30–41. ISBN: 978-3-540-66731-5.

[63] C. Gavoille and O. Ly. “Distance Labeling in Hyperbolic Graphs”. English. In: *Algorithms and Computation*. Ed. by X. Deng and D.-Z. Du. Vol. 3827. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, pp. 1071–1079. ISBN: 978-3-540-30935-2.

[64] M. Camelo et al. “Functional Model of a Routing System Architecture”. In: *Proceedings of the 1st Workshop ”Future Internet: Efficiency in high-speed networks” (W-FIERRO 2011)*. Ed. by P. P. Mariño. July 2011, pp. 19–26. ISBN: 978-84-96997-69-1.

[65] G. G. Finn. *Routing and addressing problems in large metropolitan-scale internetworks*. Tech. rep. DTIC Document, 1987.

[66] E. Kranakis, H. Singh, and J. Urrutia. “Compass Routing on Geometric Networks”. In: *In Proc. 11th Canadian Conference on Computational Geometry*. 1999, pp. 51–54.

[67] T.-C. Hou and V. Li. “Transmission Range Control in Multihop Packet Radio Networks”. In: *Communications, IEEE Transactions on* 34.1 (Jan. 1986), pp. 38–44. ISSN: 0090-6778.

[68] H. Takagi and L. Kleinrock. “Optimal Transmission Ranges for Randomly Distributed Packet Radio Terminals”. In: *Communications, IEEE Transactions on* 32.3 (Mar. 1984), pp. 246–257. ISSN: 0090-6778.

[69] A. Rao et al. “Geographic Routing Without Location Information”. In: *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*. MobiCom ’03. San Diego, CA, USA: ACM, 2003, pp. 96–108. ISBN: 1-58113-753-2.

[70] A. Caruso, S. Chessa, S. De, and A. Urpi. “GPS free coordinate assignment and routing in wireless sensor networks”. In: *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*. Vol. 1. Mar. 2005, 150–160 vol. 1.

[71] S. De. “On hop count and euclidean distance in greedy forwarding in wireless ad hoc networks”. In: *Communications Letters, IEEE* 9.11 (Nov. 2005), pp. 1000–1002. ISSN: 1089-7798.

[72] F. Kuhn, R. Wattenhofer, and A. Zollinger. “Worst-Case Optimal and Average-case Efficient Geometric Ad-hoc Routing”. In: *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking &Amp; Computing*. MobiHoc ’03. Annapolis, Maryland, USA: ACM, 2003, pp. 267–278. ISBN: 1-58113-684-6.

[73] P. Bose, P. Morin, I. Stojmenović, and J. Urrutia. “Routing with Guaranteed Delivery in Ad Hoc Wireless Networks”. In: *Wirel. Netw.* 7.6 (Nov. 2001), pp. 609–616. ISSN: 1022-0038.

[74] M. Huson and A. Sen. “Broadcast scheduling algorithms for radio networks”. In: *Military Communications Conference, 1995. MILCOM ’95, Conference Record, IEEE*. Vol. 2. Nov. 1995, 647–651 vol.2.

[75] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. “Geographic Routing Made Practical”. In: *Proceedings of the 2Nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2*. NSDI’05. Berkeley, CA, USA: USENIX Association, 2005, pp. 217–230.

[76] K. Alzoubi et al. “Geometric spanners for wireless ad hoc networks”. In: *Parallel and Distributed Systems, IEEE Transactions on* 14.4 (Apr. 2003), pp. 408–421. ISSN: 1045-9219.

[77] J. Gao et al. “Geometric spanners for routing in mobile networks”. In: *Selected Areas in Communications, IEEE Journal on* 23.1 (Jan. 2005), pp. 174–185. ISSN: 0733-8716.

[78] R. Dhandapani. “Greedy Drawings of Triangulations”. English. In: *Discrete & Computational Geometry* 43.2 (2010), pp. 375–392. ISSN: 0179-5376.

[79] A. Cvetkovski and M. Crovella. “Hyperbolic Embedding and Routing for Dynamic Graphs”. In: *INFOCOM 2009, IEEE*. Apr. 2009, pp. 1647–1655.

[80] P. Indyk. “Algorithmic applications of low-distortion geometric embeddings”. In: *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*. Oct. 2001, pp. 10–33.

[81] P. Indyk and J. Matousek. “Low-distortion embeddings of finite metric spaces”. In: *Handbook of Discrete and Computational Geometry* 37 (2004), pp. 177–196.

[82] M. Badoiu et al. “Approximation Algorithms for Low-distortion Embeddings into Low-dimensional Spaces”. In: *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA ’05. Vancouver, British Columbia: Society for Industrial and Applied Mathematics, 2005, pp. 119–128. ISBN: 0-89871-585-7.

[83] N. Alon et al. “Ordinal Embeddings of Minimum Relaxation: General Properties, Trees, and Ultrametrics”. In: *ACM Trans. Algorithms* 4.4 (Aug. 2008), 46:1–46:21. ISSN: 1549-6325.

[84] M. Badoiu et al. “Ordinal Embedding: Approximation Algorithms and Dimensionality Reduction”. English. In: *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*. Ed. by A. Goel, K. Jansen, J. D. Rolim, and R. Rubinfeld. Vol. 5171. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, pp. 21–34. ISBN: 978-3-540-85362-6.

[85] M. T. Goodrich and D. Strash. “Succinct Greedy Geometric Routing in the Euclidean Plane”. In: *CoRR* abs/0812.3893 (2008).

[86] N. Moidu et al. “On Generalized Planar Skyline and Convex Hull Range Queries”. English. In: *Algorithms and Computation*. Ed. by S. Pal and K. Sadakane. Vol. 8344. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 34–43. ISBN: 978-3-319-04656-3.

[87] M. Newman. *Networks: An Introduction*. New York, NY, USA: Oxford University Press, Inc., 2010. ISBN: 0199206651, 9780199206650.

[88] A.-L. Barabási and R. Albert. “Emergence of scaling in random networks”. In: *science* 286.5439 (1999), pp. 509–512.

[89] D. J. Watts and S. H. Strogatz. “Collective dynamics of small-world networks”. In: *nature* 393.6684 (1998), pp. 440–442.

[90] S. Milgram. “The small world problem”. In: *Psychology Today* 2.1 (1967), pp. 60–67.

[91] T. Ng and H. Zhang. “Predicting Internet network distance with coordinates-based approaches”. In: *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*. Vol. 1. 2002, 170–179 vol.1.

[92] S. Lee, Z.-L. Zhang, S. Sahu, and D. Saha. “On Suitability of Euclidean Embedding of Internet Hosts”. In: *SIGMETRICS Perform. Eval. Rev.* 34.1 (June 2006), pp. 157–168. ISSN: 0163-5999.

[93] Y. Shavitt and T. Tankel. "On the curvature of the Internet and its usage for overlay construction and distance estimation". In: *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*. Vol. 1. IEEE. 2004.

[94] M. Á. Serrano, D. Krioukov, and M. Boguñá. "Self-Similarity of Complex Networks and Hidden Metric Spaces". In: *Phys. Rev. Lett.* 100 (7 Feb. 2008), p. 078701.

[95] M. Boguna, D. Krioukov, and K. C. Claffy. "Navigability of complex networks". In: *Nature Physics* 5.1 (2008), pp. 74–80.

[96] F. Papadopoulos, D. Krioukov, M. Boguñá, and A. Vahdat. "Greedy Forwarding in Dynamic Scale-free Networks Embedded in Hyperbolic Metric Spaces". In: *Proceedings of the 29th Conference on Information Communications*. INFOCOM'10. San Diego, California, USA: IEEE Press, 2010, pp. 2973–2981. ISBN: 978-1-4244-5836-3.

[97] D. K. Marian Boguna Fragkiskos Papadopoulos. "Sustaining the Internet with hyperbolic mapping". In: *Nature Communications* 1 (6 2010), pp. 1–8.

[98] J. Herzen, C. Westphal, and P. Thiran. "Scalable Routing Easy As PIE: A Practical Isometric Embedding Protocol". In: *Proceedings of the 2011 19th IEEE International Conference on Network Protocols*. ICNP '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 49–58. ISBN: 978-1-4577-1392-7.

[99] K. Bilal et al. "Quantitative comparisons of the state-of-the-art data center architectures". In: *Concurrency and Computation: Practice and Experience* 25.12 (2013), pp. 1771–1783. ISSN: 1532-0634.

[100] K. Bilal et al. "A taxonomy and survey on Green Data Center Networks". In: *Future Generation Computer Systems* 36 (2014). Special Section: Intelligent Big Data Processing Special Section: Behavior Data Security Issues in Network Information Propagation Special Section: Energy-efficiency in Large Distributed Computing Architectures Special Section: eScience Infrastructure and Applications, pp. 189–208. ISSN: 0167-739X.

[101] C. Guo et al. "Dcell: a scalable and fault-tolerant network structure for data centers". In: *ACM SIGCOMM Computer Communication Review* 38.4 (2008), pp. 75–86.

[102] D. Li et al. "FiConn: Using backup port for server interconnection in data centers". In: *INFOCOM 2009, IEEE*. IEEE. 2009, pp. 2276–2285.

[103] M. Csernai et al. “Poincare: A Hyperbolic Data Center Architecture”. In: *Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on*. June 2012, pp. 8–16.

[104] H. Abu-Libdeh et al. “Symbiotic Routing in Future Data Centers”. In: *SIGCOMM Comput. Commun. Rev.* 40.4 (Aug. 2010), pp. 51–62. ISSN: 0146-4833.

[105] C. Glass and L. Ni. “The Turn Model for Adaptive Routing”. In: *Computer Architecture, 1992. Proceedings., The 19th Annual International Symposium on*. 1992, pp. 278–287.

[106] J.-Y. Shin, B. Wong, and E. G. Sirer. “Small-world Datacenters”. In: *Proceedings of the 2Nd ACM Symposium on Cloud Computing*. SOCC ’11. Cascais, Portugal: ACM, 2011, 2:1–2:13. ISBN: 978-1-4503-0976-9.

[107] J. M. Kleinberg. “Navigation in a small world”. In: *Nature* 406.6798 (2000), pp. 845–845.

[108] S. Akers and B. Krishnamurthy. “A group-theoretic model for symmetric interconnection networks”. In: *Computers, IEEE Transactions on* 38.4 (Apr. 1989), pp. 555–566. ISSN: 0018-9340.

[109] S. T. Schibell and R. M. Stafford. “Processor interconnection networks from Cayley graphs”. In: *Discrete Applied Mathematics* 40.3 (1992), pp. 333–357. ISSN: 0166-218X.

[110] M. Miller and J. Širán. “Moore graphs and beyond: A survey of the degree/diameter problem”. In: *Electronic Journal of Combinatorics* 20.2 (2013).

[111] W. Xiao and B. Parhami. “Cayley graphs as models of deterministic small-world networks”. In: *Information Processing Letters* 97.3 (2006), pp. 115–117. ISSN: 0020-0190.

[112] F. L. Wu, S. Lakshmivarahan, and S. Dhall. “Routing in a Class of Cayley Graphs of Semidirect Products of Finite Groups”. In: *Journal of Parallel and Distributed Computing* 60.5 (2000), pp. 539–565. ISSN: 0743-7315.

[113] B. Chen, W. Xiao, and B. Parhami. “Internode Distance and Optimal Routing in a Class of Alternating Group Networks”. In: *Computers, IEEE Transactions on* 55.12 (Dec. 2006), pp. 1645–1648. ISSN: 0018-9340.

[114] C. Wei-Kuo and C. Rong-Jaye. “The (n, k)-star graph: A generalized star graph”. In: *Information Processing Letters* 56.5 (1995), pp. 259–264. ISSN: 0020-0190.

[115] S. B. Akers, D. Harel, and B. Krishnamurthy. “Star Graph: An Attractive Alternative to the n-cube”. In: cited By (since 1996)259. 1987, pp. 393–400.

[116] K. W. Tang and B. W. Arden. “Vertex-transitivity and Routing for Cayley Graphs in GCR Representations”. In: *Proceedings of the 1992 ACM/SIGAPP Symposium on Applied Computing: Technological Challenges*. SAC ’92. Kansas City, Missouri, USA: ACM, 1992, pp. 1180–1187. ISBN: 0-89791-502-X.

[117] J. Ryu, E. Noel, and K. Tang. “Fault-tolerant routing on Borel Cayley graph”. In: *Communications (ICC), 2012 IEEE International Conference on*. June 2012, pp. 2872–2877.

[118] J.-Y. Shin, E. G. Sirer, H. Weatherspoon, and D. Kirovski. “On the Feasibility of Completely Wireless Datacenters”. In: *Proceedings of the Eighth ACM/IEEE Symposium on Architectures for Networking and Communications Systems*. ANCS ’12. Austin, Texas, USA: ACM, 2012, pp. 3–14. ISBN: 978-1-4503-1685-9.

[119] D. Helic, M. Strohmaier, and W. Wojcik. “Navigational evaluation of breadth first search spanning trees”. In: *Information Communication Technology Electronics Microelectronics (MIPRO), 2013 36th International Convention on*. May 2013, pp. 998–1003.

[120] A. Cvetkovski and M. Crovella. “Low-stretch greedy embedding heuristics”. In: *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*. Mar. 2012, pp. 232–237.

[121] M. Khan and G. Pandurangan. “A Fast Distributed Approximation Algorithm for Minimum Spanning Trees”. In: *Proceedings of the 20th International Conference on Distributed Computing*. DISC’06. Stockholm, Sweden: Springer-Verlag, 2006, pp. 355–369. ISBN: 3-540-44624-9, 978-3-540-44624-8.

[122] C. Boulinier, A. K. Datta, L. L. Larmore, and F. Petit. “Space efficient and time optimal distributed {BFS} tree construction”. In: *Information Processing Letters* 108.5 (2008), pp. 273–278. ISSN: 0020-0190.

[123] S. Makki and G. Havas. “Distributed Algorithms for Constructing a Depth-First-Search Tree”. In: *Parallel Processing, 1994. Vol. 1. ICPP 1994. International Conference on*. Vol. 3. Aug. 1994, pp. 270–273.

[124] M. Bui, F. Butelle, and C. Lavault. “A distributed algorithm for constructing a minimum diameter spanning tree”. In: *Journal of Parallel and Distributed Computing* 64.5 (2004), pp. 571–577. ISSN: 0743-7315.

[125] L. Blin and F. Butelle. “The first approximated distributed algorithm for the minimum degree spanning tree problem on general graphs”. In: *Parallel and Distributed Processing Symposium, 2003. Proceedings. International*. Apr. 2003.

[126] J. Szymański. "Concentration of Vertex Degrees in a Scale-free Random Graph Process". In: *Random Struct. Algorithms* 26.1-2 (Jan. 2005), pp. 224–236. ISSN: 1042-9832.

[127] N. A. Lynch. *Distributed algorithms*. Morgan Kaufmann, 1996.

[128] Y. Rabinovich and R. Raz. "Lower Bounds on the Distortion of Embedding Finite Metric Spaces in Graphs". English. In: *Discrete & Computational Geometry* 19.1 (1998), pp. 79–94. ISSN: 0179-5376.

[129] Y. Qin, W. Xiao, and S. Miklavic. "Connected graphs as subgraphs of Cayley graphs: Conditions on Hamiltonicity". In: *Discrete Mathematics* 309.17 (2009), pp. 5426–5431. ISSN: 0012-365X.

[130] V. Diekert. "Complete semi-Thue systems for abelian groups". In: *Theoretical Computer Science* 44 (1986), pp. 199–208. ISSN: 0304-3975.

[131] D. G. Corneil and R. Krueger. "Simple vertex ordering characterizations for graph search: (expanded abstract)". In: *Electronic Notes in Discrete Mathematics* 22 (2005). 7th International Colloquium on Graph Theory, pp. 445–449. ISSN: 1571-0653.

[132] C. Lavault. "Interconnection Networks: Graph-and Group Theoretic Modelling". In: *Proceedings of the 12th International Conference on Control Systems and Computer Science (CSCS12)*. Vol. 2. 1999.

[133] E. Konstantinova. "Some problems on Cayley graphs". In: *Linear Algebra and its Applications* 429 (2008), pp. 2754–2769. ISSN: 0024-3795.

[134] J. Xu. *Topological Structure and Analysis of Interconnection Networks*. 1st. Springer Publishing Company, Incorporated, 2010. ISBN: 9781441952035.

[135] G. Marsaglia and J. C. W. Marsaglia. "A New Derivation of Stirling's Approximation to n!" English. In: *The American Mathematical Monthly* 97.9 (1990), pp. 826–829. ISSN: 00029890.

[136] J. Schmidt. "Finite groups have short rewriting systems". In: *Computational Group Theory and the Theory of Groups, II* 511 (2010), p. 185.

[137] P. Holme and B. J. Kim. "Growing scale-free networks with tunable clustering". In: *Phys. Rev. E* 65 (2 Jan. 2002).

[138] C. Biemann. *Structure discovery in natural language*. Springer Berlin Heidelberg, 2012. ISBN: 978-3-642-25923-4.

[139] *Rocketfuel: An ISP Topology Mapping Engine*. URL: `http://research.cs.washington.edu/networking/rocketfuel/`.