NTNU – Trondheim
Norwegian University of
Science and Technology

Swarm Routing in Information Centric Networks

Adrià Descamps Vilà

Submission date: June 2019
Responsible professor: Otto J Wittner, ITEM
Supervisor: Otto J Wittner, ITEM

Norwegian University of Science and Technology
Department of Information Security and Communication Technology

Title: Swarm Routing in Information Centric Networks

Student: Adrià Descamps Vilà

Problem description:

Information Centric Networks (ICN) is a recent concept of network architecture which tries to shape the Internet architecture to the current Internet usage. Its focus is to move from the host-to-host model of the current Internet, to a content-centric one where the distribution of data will be designed following this idea. There are a number of research projects investigating the ICN approach. In this thesis, as a result of preliminary studies, the ICN project Name Data Networking (NDN) will be the base of the experimental network architecture. If time allows the ICN project Network of Information (NetInf) may also be considered.

For two decades there has been ongoing research on a routing model based on Swarm Intelligence (SI). SI clings on the idea that a single organism by itself is a simple unit, nevertheless when multiple units work together towards a common goal they are able to solve more complex problems. The utilization of SI for routing in telecommunication networks has been researched, however it has never been approached from the ICN perspective before.

This thesis will study the applicability of SI over an ICN architecture. Work on integration design between SI and ICN will be a core part. The use of a simulator will help to show how these two technologies can work together and whether SI routing is an actual alternative to the traditional routing systems. Routing efficiency and scalability will be investigated. To the extent time allows the limitations and benefits of using SI and ICN as network architecture in contrast to the traditional Internet architecture will be looked into.

Responsible professor: Otto J Wittner, ITEM

Supervisor: Otto J Wittner, ITEM

Abstract

The current Internet usage is changing, with an important increase in the consumption of content-based applications. However, the current host-based architecture is not prepared to integrate this amount of information. Therefore, novel proposals for the future Internet architecture are researched, as Information Centric Networks (ICN). ICN tries to shape the Internet architecture to the current Internet challenges by moving from the host-to-host model to a content-centric substitute.

The survey on the related work shows how the routing techniques proposed to solve this subject take quite contrasting directions, from a deterministic link-state to a broadcast system. The literature review provides signs suggesting that Name Data Networking, a research project under ICN, might be able to adopt a swarm technique for the routing and forwarding planes due to the similarities between both concepts. Therefore, these techniques are thoroughly studied to present a design proposal implementing them, which applies the features from an ant-based system and complies with the NDN architecture. The system that evaluates the design is a simulator, implemented by means of several iterations of *design* $\circlearrowleft$ *implementation*.

The gathering of data comes from different experiments over two distinct topologies. The results certify the correct behavior of the system, operating as it is expected, and providing with interesting results. Moreover, the comparison between a flooding system with the ant-based design establishes a more fruitful outcome for the later in terms of content retrieved and path length.

The contribution done by this Thesis is to provide an alternative with promising results to the currently employed routing techniques in the NDN field. hopefully, the solution presented may support further work in this line of research.

Preface

This Thesis is the last step in order to obtain the MSc in Communication Technology by the Norwegian University of Technology and Science (NTNU). The project has been realized from February to June 2019.

It provides an insight into the concepts as Name Data Networking and Swarm Intelligence, intending to integrate them into a noveldesign to evaluate its synergy.

Enormous appreciation is given to my supervisor, Otto Wittner, for his invaluable mentoring and guidance throughout this project. Also, special gratitude to my family for all the support given along this journey. Finally, but not least, an extensive thanks to all the wonderful people who appeared during this time in Trondheim.

Contents

List of Figures

List of Tables

List of Acronyms

CDN Content Delivery Network.

CS Content Store.

FIB Forwarding Information Base.

ICN Information Centric Network.

LSA Link State Advertisement.

LSDB Link State Database.

NDN Named Data Networking.

NLSR Named-Data Link State protocol.

NTNU Norwegian University of Science and Technology.

PIT Pending Interest Table.

RTT Round Trip delay Time.

SI Swarm Intelligence.

Chapter 1
Introduction

In recent times, the Internet as architecture has been experiencing a core change. From the traditional client-server scheme and host-to-host interactivity, moving to a more content centered model. According to Cisco [4], in 2016 up to 52% of the world Internet traffic was carried by Content Delivery Network (CDN) solutions, with a forecast of 71% of the global traffic crossing a CDN in 2021. This means that the long-established architecture might become obsolete for this new design. Several elemental components are at stake and that will need to be addressed. One of these elements to be discussed is the routing towards content as opposed to a destination.

Throughout the last 20 years, insect behavior has been researched and categorized in a way it could be used for routing. There have been algorithms that have tried to imitate the natural side of insects to improve the current routing algorithms. Accordingly, there are many pieces of research on the topic, moreover, in recent years the efforts have been concentrated mostly on ants and bees' behavior. These algorithms, like the biological insects, are not deterministic, but rather use other decision systems which are called stochastic.

1.1 Motivation

Given the change of paradigm happening on the current Internet where the traffic is shifting and becoming more content centered, while the actual architecture is still the one designed in the 70s to interconnect a few computers, the TCP/IP solution formerly brought is starting to get outdated. Hence, the research community tried to come up with alternative architectures that could adapt better to this variation [17]. ICN is one of these alternatives, which relies on location-independent naming, in-network caching and name-based routing to distribute effectively the content through the network. Several challenges have to be resolved before this technology is feasible to be implemented as an alternative, however, it has significant chances to become the next worldwide Internet architecture.

Swarm Intelligence (SI) deals with collective behaviors that result from the local interactions of individual components with each other along with their environment [3]. The goal of SI is to treat complex and sophisticated problems with individual units which communicating and working as a group, a natural colony, can face as well as solve these obstacles. In the last years, SI has been applied as a solution for combinational problems, telecommunications, etc., with promising outcomes. Two of the most successful examples are Ant Colony Optimization [8] and AntNet [7], which have been extended on later papers. Latterly, they have been researched to use SI on wireless ad-hoc networks with good results [21].

The behavior adopted by swarm systems in nature typically shows how they explore the environment, looking for something rather than for a specific location. Hence, the form of search based on swarm systems towards content seems to be directly applicable to the ICN foundation. With the knowledge that SI is applicable, furthermore efficient, on the traditional network architecture together with that the ICN design seems to adapt conveniently well to the SI behavior, it provides a promising composition that has a big potential. The high mobility of the content is one of the main challenges a content-based architecture might face. In that regard, the SI may provide a flexible solution thanks to its stochasticity and adaptability.

1.2 Objectives

This section defines the specific goals for these studies refined from the main hypothesis, a swarm routing approach can be applied to an ICN network. It delimits the scope and precision for each one of the research questions asked, as well as for each stage of the research.

1.2.1 Research questions

The purpose of this MSc Thesis is to answer the following Research Questions.

- **RQ1** Can Swarm Intelligence become an alternative to the current routing schemes in an Information Centric Network architecture?
  - What is an effective design which integrates SI in an ICN architecture?
  - How can the ICN architecture be adapted to integrate SI?
- **RQ2** How does SI over an ICN architecture deal with routing efficiency and scalability?
  - How does the system scale faced with the growth of the network topology?
  - How does SI compare with similar approaches in terms of scalability?

1.2.2 Scope

Naturally, the extent of a solution solving these Research Questions could be enormous. Therefore, some boundaries are stated to the work to be done. The goal is to get proof that an SI approach may become an alternative, hence providing with simple routing results showing an encouraging behavior, but not offering a ready-to-implement solution. Furthermore, regarding the scalability of the proposal, the focus is to establish the correct operation of the components that can be sensitive to increase the network size.

1.3 Outline

An outline for the rest of the project is offered.

- **Chapter 2** It shows the literature review related to the work done in this project.
- **Chapter 3** Given a theoretical view on the technologies used. Besides, the related work and how it compares to the variant chosen in our proposal is introduced.
- **Chapter 4** A view on the methodology followed and a perspective of the design and results obtained are presented.
- **Chapter 5** The design, implementation and experiments are presented in extension.
- **Chapter 6** The results and their explanation and a discussion is offered.
- **Chapter 7** The final views on the work done in this project are stated.

Chapter 2
Background

This chapter focuses on the work done in the Specialization Project previous to this MSc Thesis. Two main concepts are introduced, Information Centric Networks and Swarm Intelligence, and some variants of each one are presented. Besides, there is a brief explanation of each one and their main contributors.

2.1 Information Centric Networks

There has been variate research to try to address this topic, with a common base where the routing paradigm shifts from host centered using the IP standard to content centered. The way they approach this transition varies in many manners, but they also share several design features [2, 1]. The following projects are being researched by competent institutions and are the major proposals to approach this transformation. The explanations are focused mainly on the naming of the content and the routing system, as they are the most important areas for this project.

2.1.1 DONA

DONA has a flat name-space that provides uniqueness on the name. However, it may become a problem when the scalability topic emerges. It needs a routing alternative structure, the Resolution Handlers, which are hierarchical [16]. Nevertheless, every node needs to have all the lower layer nodes in its database meaning that Tier-1 nodes must have all the Internet nodes saved. Some articles warn and are skeptical [2, 1] of this technology due to the scalability issue as they see unfeasible to implement it on the current Internet. Notice that this technology advocates for a clean slate redesign of the whole Internet without contemplating an incremental approach.

2.1.2 NetInf

Its naming scheme is flat and similar to DONAs. It gives a unique identifier to each content and it identifies the owner in the own ID (P:L). This provides anonymity,

but a mechanism to verify the content [6]. It uses asymmetric encryption with the public/private key pair mechanism. The routing is based on multilevel DHT. Previous technologies using DHT used flood-based and heuristic-based routing algorithms, which can give an impression of the possible characteristics of the routing algorithms that can be used. The resemblance between flooding and using heuristics with the way swarm routing works is significant.

2.1.3 PURSUIT

The naming scheme used is very similar to the one used by DONA. In another way, this technology uses rendezvous points to publish and subscribe to content. This approach also uses a flat namespace, which increases the concerns accumulated with DONA on how to implement this technique worldwide. This type of routing paradigm, as introduced by Fotiou et al. [11], would make it difficult to implement a swarm based routing algorithm without modifying the core concept of the network itself.

2.1.4 NDN

The naming scheme on NDN is hierarchical with multiple components. There is no restriction on the naming except some component structure, which relies on the user to generate and assign them. A concern that has arisen is how secure is this mechanism over ambiguity, albeit it is supposed to implement a SHA256 digest to resolve this issue. The routing paradigm uses two key messages, Interest and Data. It broadcasts the Interest message to find the content over the network and once found, it sends the Data message with the content [14]. It might be possible to extrapolate directly this scheme to the swarm routing paradigm where the ants would work as the Interest messages to find the path to the destination to retrieve the Data messages.

2.2 Swarm Intelligence

The concept of Swarm Intelligence hangs from the natural intelligence of some types of animals to communicate and solve several different problems. It is based on the core idea that a single unit from a group cannot solve high complex situations by itself, but with the interaction of various entities from its group, they are able to resolve it. This idea has been used as a basis to develop algorithms; in the area of telecommunication networks, to solve and/or optimize the routing problem. Different kinds of animals have been used as a reference to build this sort of algorithms, like ants or bees. Following, some of these algorithms are exposed with their main characteristics.

2.2.1 Ant-based algorithms

There are several studies and research which have developed algorithms working with Swarm Intelligence. The following ones are ant-based algorithms, which rely on the foundation concept of using a chemical substance named pheromone. This element is used by ants to generate a trail from their colony to a food source. The notion of leaving a trail once found the desired resource is the notion all these projects have been founded on.

Ant Based Control This method is designed to work on symmetric networks performing load balancing. To avoid or mitigate stagnation, it uses aging, delaying and noise [22]. There is a later method which shows how forwarding ants can update the path information without assuming symmetric path costs, Cooperative Asymmetric Forward [13].

AntNet This approach aims to optimize the performance of the entire network. It explores the neighboring nodes at initialization and in the case of a link failure, it sets the level of pheromones on the nodes to 0 and it redistributes it with the adjacent links [7].

Ant Colony Optimization The ACO mechanism differentiates the agent actions when they are on forward and on backward modes. The pheromone is being updated anytime the agent goes through a node, not only when this comes back as this implementation takes into account the possibility that the agent is taking a different path to go from destination to the original source. Regardless of this practice, it uses a different type of pheromones when the agent is returning [9, 8].

The IIK department at NTNU has been working for these past years on relevant research regarding SI applied to different network technologies as well as the optimization of its characteristics. Its work is of great interest to the ongoing project due to the similarities and particularities in both types of research.

- Wittner et al. [27] introduce a swarm based optimization algorithm that is capable of finding paths of resources in a complex network environment. The algorithm is fully distributed and may be implemented using simple ant-like mobile agents.
- The cross-entropy ant system (CEAS) is a distributed, robust and adaptive swarm intelligence system for dealing with path management in communication networks [12]. It shows excellent results in areas of great interest in this project. Posterior research has been done to improve some of the features, like the work done on the auto-regressive functions at the core of the system [18].

– Csorba et al. [5] presented this work which targets the problem of finding optimal deployment mappings involving multiple services, while capturing important non-functional requirements of distributed services, regarding performance and dependability.

2.2.2 Bee based routing

These algorithms are based on the concept of how bees find food. It follows the discovery techniques from the beehive to the aliment, how bees behave once they discovered food source and the methods they employ to communicate with other colleagues to spread the knowledge. They have an adaptive division of labor. The process goes as follows; one individual finds food, it comes back to the source and there it announces several characteristics like the quality, the distance, directions, etc. in a one-to-group communication way. Examples of work related to this technology are the following.

BeeHive Bee agents travel through network areas called foraging zones. It is fault tolerant, scalable and it relies on local information [25].

BeeAdHoc Routing algorithm for energy efficient routing in mobile ad-hoc networks. It mainly utilizes two types of agents, scouts and foragers, for doing routing in mobile ad-hoc networks [26].

Chapter 3
Foundations

This chapter presents the necessary ground knowledge to understand the rest of the work. The methods and systems chosen in later chapters and sections are explained in detail so that the reader can follow with all the procedures. First, Name Data Networking is explained, as a variant of an ICN introduced in Section 2.1. Then, the architecture and functions of an ant-based system as a branch of Swarm Intelligence is described. Finally, the related work in this project is introduced by exposing various schemes that try to solve similar goals as this project.

3.1 Named Data Networking

Named Data Networking is a new network architecture that proposes an evolution of the IP architecture. It changes the semantics of network services from *delivering packet to a given destination* to *retrieving data with a given name*.

3.1.1 Architecture

The communication in NDN is receiver-driven. It bases on the exchange of two different packets, Interest and Data. Both types of packet carry a name which identifies a content (see Figure 3.1). A consumer inserts a name to an Interest packet and sends it to the network to be transmitted. This name is used by the routers to forward the packet towards the data producer. Take into account that any node in the network where the Interest pass through can answer with the Data packet if it has it in its Content Store cached. The path the Data packet follows to arrive at the consumer is the same the Interest packet has employed. Interest packets carry a nonce field to detect forwarding loops.

There are three key data structures to carry out the Interest and Data packet forwarding, i.e. Content Store (CS), Pending Interest Table (PIT) and Forwarding Information Base (FIB). CS and PIT are unique to NDN, where the Content Store is responsible to temporarily cache the data retrieved and sent through its node. The

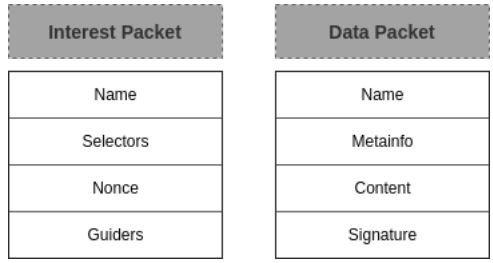


Figure 3.1: Standard packet structure in NDN Architecture.

Pending Interest Table saves the data name carried in the Interest that the router receives together with its incoming interfaces. Finally, the FIB is not unique to NDN but it is different from the IP FIB that one can be used to. It indexes by name-prefix which allows the FIB to provide multiple output interface instead of a unique best interface for each prefix. Its main function is to help with the routing and forwarding of Interest packets (shown in Section 3.1.3).

3.1.2 Namespace

Namespace management does not belong to the NDN architecture, as IP address space management does not belong to the IP architecture. Nevertheless, for NDN applications naming is the most important factor. Applications may choose the naming scheme fitting best their needs, as a result, naming can evolve independently from the network. Routers are able to determine the boundaries among components in a name as they are opaque to the network, namely, they do not ascribe any meaning to them.

Its design adopts hierarchical structure names, similar to */ntnu/telematics/description.pdf*, where the '/' is delimiting name components. In addition, this structure allows symbolizing the context and relationships of data element. For instance, the second version of the assignment description could be found in */ntnu/telematics/description.pdf/2*. Moreover, name aggregation is also supported in this design. For globally accessible data, there is a need to assure global unique names, whereas for local communications it might be sufficient to employ local routing to retrieve the data.

It must be possible to construct data names in a deterministic manner by the consumers in order to retrieve dynamically generated data. There can be two methods to achieve that; using a deterministic algorithm know by consumer and producer, so both can generate the same data name, or using Interest selectors together with a longest prefix matching strategy to retrieve the data through iterations. Partially known data can also be retrieved with the help of a set of selectors as Zhang et al.

[29] experienced. As an illustration, a consumer wanting the description of the assignment may request */ntnu/telematics/description.pdf* with the Interest selector "leftmost child" and receive a Data packet named */ntnu/telematics/description.pdf/1* corresponding to the first version of the document. Afterward, the consumer can request further versions of the document with the information received by the first Data packet combined with the naming protocol from the data producer.

By letting the applications design their naming scheme, the outline of the application data and its use of the network becomes much more relatable. Naming data also allows for capabilities such as content distribution, multicast, delay-tolerant network, and mobility.

3.1.3 Routing and Forwarding

The NDN design, respect routing and forwarding, gives three solutions to current IP's design on the grounds that it routes and forwards packets based on names instead of addresses. This allows fixing the NAT traversal problem, getting rid of both public and private addresses. Therefore, the address assignment is also not needed anymore. Finally, there is no address space consumption as the namespace is boundless.

The forwarding plane is the control plane in NDN. The forwarding strategy module makes the decisions on its own, based on the data found in the FIB. This circumstance raises the question of whether routing is still needed in NDN networks. The forwarding strategy module is responsible for taking several decisions such as the interface where an Interest must be forwarded, the number of unsatisfied Interest packets to be kept in the PIT, the serving preference of the Interests received and whether load-balancing Interests among multiple interfaces should be used, etc.

Even if the forwarding capabilities are remarkably wider, as Yi et al. [28] suggested, a minimal routing system is needed to fill up the FIB with some information to not let the network be flooded each time a node appears.

Forwarding in NDN's data plane hinges on the Pending Interest Table. It registers each Interest with its incoming interface. Once a Data packet name matches a saved Interest name or a timeout occurs, it is removed from the table. A benefit from the PIT state is that routers can identify packets in a loop and, therefore, discard them. This provides the possibility to reach the data producer through multiple paths, thus supporting multicast data delivery. Moreover, flow balance can be achieved by controlling the number of pending Interests, so the traffic load, by the router.

The forwarding process in NDN works as follows (shown in Figure 3.2). When a consumer wants to retrieve some data it inserts its name to an Interest. This Interest is sent to the network and received by a router, who checks whether it has

any data matching the Interest name in its Content Store. If so, the router delivers the data to the consumer. Otherwise, the PIT is inspected to verify whether it already exists an entry matching the Interest. In that case, the router would add the incoming interface to the PIT. Alternatively, it creates an entry within the PIT for that Interest name. To forward the Interest through the network, the router uses the FIB information to decide, in conjunction with the forwarding strategy, which of the entries in the table is a better option to use. If there is no feasible alternative in the FIB, the router may return an Interest NACK to the incoming interface. Once the Interest reaches the data producer or a node in the network has it cached in its CS, the Data packet is returned. It pursues the reverse path followed by the Interest packet. When a router receives a Data packet, it checks the list of pending Interest in its PIT and depending on whether the name from the Data packet is there, it drops or forwards it. This is due to the expiration times given to the packets, so the table can efficiently be managed. Once the router forwards the Data packet, it can decide to cache it in its CS following its caching strategy. It is important to mention that neither the Interest nor the Data packets carry any host or interface information. The Interest is forwarded to the data producers based on the data names, and the Data packets are forwarded based on the information saved in the Pending Interest Table. It differs completely from the IP's host-to-host concept as it removes the need for source and destination addresses.

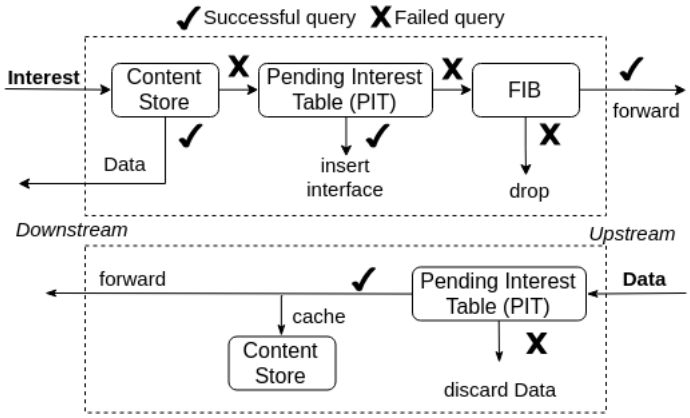


Figure 3.2: Forwarding Process followed by an NDN node.

3.1.4 In-network Storage

In-network storage might be one of the most important features brought by NDN. It allows the nodes in the network to store the data being forwarded so that it can be reused on a later petition. This caching scheme is promising and it changes the way routing and forwarding must be addressed. For a consumer, the data stored in a

node is the same as the one retrievable at the data producer, so the challenge might be to efficiently retrieve the data from the best source given some specific metrics.

The fact that data can be stored in the network raises some concerns regarding its privacy. This is an issue that the focus must be on as NDN as an alternative will not be feasible until data protection is properly assured. Further considerations must be taken regarding the fact that no addresses are used difficult, if not prevent, to know who the requester is. And hence, to know who has the rights to access the data stored in a node.

3.1.5 Data security

Contrary to TCP/IP, NDN has some built-in security in its architecture design. All data producers are required to sign cryptographically every Data packet so the data is secured. This allows the consumers to trust the data independently from where they got it, exposed in Section 3.1.4, as the publisher's signature provides data integrity. One of the prime research focus is currently the design and development of mechanisms to manage user trust. Routers, routing processes, routing data are identified by structured hierarchical names, like any other NDN data. Therefore routing and control messages require signatures providing a strong basis to secure routing protocols against attacks like tampering or spoofing. Also, it is more difficult to target a specific device since NDN design focus on naming content instead of addressing hosts.

3.2 Swarm Intelligence

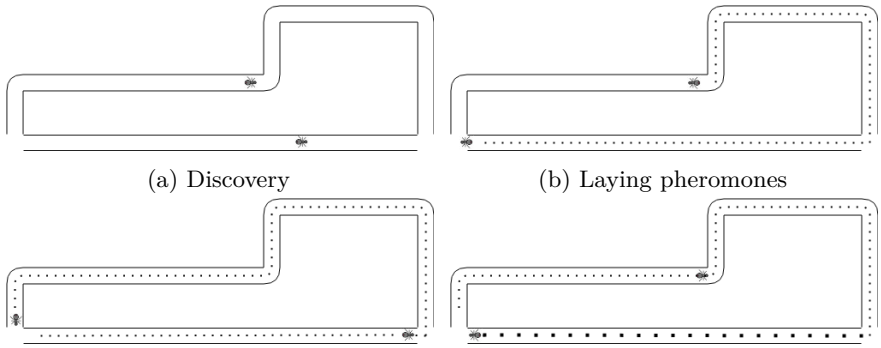
This chapter addresses Swarm Intelligence and how it is designed and implemented. It focuses on the approach it takes to simulate natural behavior on algorithmic and network routing and which mechanisms and methods can be applied for that purpose.

3.2.1 SI architecture

An SI system is a consolidation of several partially, or totally, autonomous units. These units possess some elemental processing and interaction capabilities that they use to fulfill a purpose. Although they have a reduced skill set as a single entity, the collective behavior of the system is based on their local interactions with each other and the local environment. Namely, they only interact and communicate within a short fraction of the environment they live in. Subsequently, no requirement for a centralized controller exists in this system as it is self-coordinate and/or self-organized, depending on the design. This is known as emergent behavior, where a complex behavior of a system is formed through the combination of simple behaviors of the agents.

These units are called agents. As mentioned before, the agents are simple units with limited processing and interaction competencies which cannot solve any difficult puzzle. Nevertheless, when several of them are working together towards the same objective they are able to achieve a solution that by themselves would be impossible to. A good example is the behavior of ant colonies. The algorithms based on them use mainly two kinds of agents; namely, forward and backward agents or ants. The forward agents are responsible to discover a path from the source, where they are generated, to the target they are seeking. The backward agents are forward agents that arrived at the destinations. Commonly, they follow the reverse path set by the forward agents to go back to the source, and they update the nodes on the way. In some implementations, some extra agents have different abilities. Examples of them can be exploring ants, created at network setup time to give a head up on the network topology to the system, so when the first content is requested the nodes have already some routing information in their tables. Another case can be ants that move around randomly watching the network, and update it constantly so it is always up to date.

The method used for these agents to communicate is pheromone laying (see Figure 3.3). The forward ant uses the available pheromone in the network to decide its next step. This pheromone is a value recorded on the node's tables to help forward the agents and the data. On each node, there is at least one table containing the pheromone values for each destination known by the node.



(a) Discovery (b) Laying pheromones

(c) Discovery with pheromones (d) Reinforcement of pheromones

Figure 3.3: Pheromone Laying process

Taking again the example of ant colonies, all these components can bound in a more general way. The communication method, pheromone laying, used by ants allows the colony to solve the shortest path problem which is far beyond the capacity of any single agent. In this case, the ants use a non-deterministic approach to the decision-making process. They incline to move forward following the strongest pheromone trail, but some ants can end up following alternative paths. This stochastic

decision simulates the exploration in nature and allows the system to overcome the limitations of the reduced outlook of their agents.

3.2.2 Stagnation mitigation

One of the challenges that SI systems tend to face is stagnation. In other words, when the network leads to its convergence point and the optimal path would be the one chosen by all the agents. To try to overcome this circumstance, several techniques can be used.

- **Pheromone Control.** This technique regulates the amount of pheromone assigned to each path while the agent is progressing. Different approaches are implementing this procedure.
  - Evaporation. It reduces the effect of past experiences by decreasing the amount of pheromones in each path for a factor p (see Figure 3.4).
  - Aging. Accordingly, past experience effect can be reduced by regulating the amount of pheromone left by an ant-related to its age; the older the ant the fewer pheromones it leaves.
  - Limiting and Smoothing Pheromone. An upper bound on the amount of pheromones allowed on each edge is set to balance the choice of the paths.
- **Pheromone-Heuristic Control.** This technique does not rely solely on pheromone for routing decisions. It adds a heuristic probabilistic function which allows a stochastic choosing process to avoid the convergence of the network.
- **Privileged Pheromone Laying.** This solution centers on giving more privileges to specific ants so they can deposit extra or more pheromones.

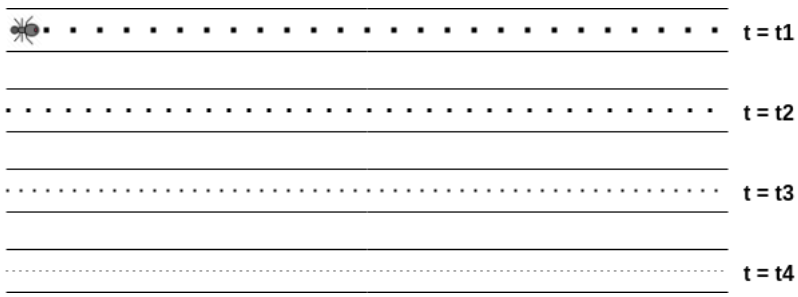


Figure 3.4: Evaporation process.

On top of these techniques, there are more rudimentary ways of dealing with it like using a delay on the flow rates of the ants from a congested node to its neighbors. Noise can also be used to allow the ants to ignore the influence of pheromones and choose a path randomly, enabling the possibility of exploring new possible optimal routes. These two methods are examples of other manners to tackle stagnation. Note that they are usually used along with the previous techniques exposed in this section to increase their global performance.

3.2.3 Routing Principles

A routing algorithm has the goal to discover paths from a source to a destination through the network and trying to optimize them by following specific metric requirements. The optimal path might be altered by changes made in the network structure. In SI a *bottom-up* approach is emphasized so that the overall response of the system relies on self-organization, redundancy, and stochasticity. Namely, the total routing solution results from the individual actions performed by each node in the network, which are given the necessary tools to act individually. There are some advantages to employ a swarm strategy. The first one is *adaptivity*. The main idea is that a routing algorithm must be able to modify its solution during run time to answer to changes in the network. In SI, multiple agents work in parallel towards an outcome providing a continuous exploration of the system. This helps it to be more adaptive to changes in the environment. These multiple agents provide redundancy in the system. *Robustness* exploits this feature to resist individual agent errors and losses, communication failures or other system-related errors. An SI system is also beneficial when it comes to scalability regarding the size of the problem due to its intrinsic redundancy and parallelism in the system. Finally, combining the previous properties results in a system that is *portable* among different scenarios.

Most of the routing algorithms implementing Swarm Intelligence present some common general principles. The routing information in swarm systems is gathered via the repeated sampling of full paths. The *redundancy* given by the multiple agents working in parallel allows all the decisive information to be based on real experience instead of on estimates or updates provided by neighbors. Agents construct a hop by hop path in a stochastic manner employing pheromone information. The use of pheromones helps to construct over the experience gathered by previous agents, using an indirect type of communication. This way of communicating and interacting with the local environment is called *stigmergy*. By updating the nodes with the experience from multiple ants, it automatically makes multiple paths available. These paths can be employed as a backup option or to provide load-balance capabilities to the routing system.

"Using separate mechanisms for exploration and exploitation allows to build a flexible system." [10].

3.2.4 SI algorithm behavior

The routing process in SI uses the forward and backward agents, or ants, to create and distribute the routing information together with the information placed in the node tables along the network.

At the beginning, the node n generates and sends *forward ants*, small control packets, in a regular basis. These agents will search for a destination d moving through the network. Forward ants decide the path towards destination at every hop; on each node, they determine which will be the next step based on a probabilistic rule. The choosing probability for the next action is given by the amount of pheromone placed on that path and a heuristic value based on some node or trail metric. By moving away from a deterministic decision process by introducing the heuristic function, the behavior of this algorithm resembles the nature of a real ant. Besides, the usage of a stochastic method provides adaptability to the system. Once the forward ant has arrived at the destination d, it becomes a *backward ant*, or backward agent. It goes back to the source node n following the reversal path tracked by the forward agent. During the route, the backward ant updates each node information about the destination d. The entries updated can be the trip time experienced by the ant, the level of pheromone left on the trail, etc. This information is used to guide the following agents towards the destination d in such a manner that it mimics the behavior of ants in nature. Data packets are routed in the same way as forwarding ants by choosing the next hop stochastically based on the pheromone values in the node's tables. Data packets are routed through the best path and they are not used for exploration.

3.3 Related Work

This section presents two different approaches which try to solve the same, or related, hypothesis than this project. Furthermore, they are compared to the expected behavior of a swarm technique, focusing on features like the traffic load, the decision-making process, how it deals with content mobility and in-network storage, the routing approach of the system or the domain they are designed for.

All these specific aspects are summarized in Table 3.1.

3.3.1 NLSR

The Named-Data Link State Routing protocol (NLSR) is a protocol for intra-domain routing in NDN. It is an application-level protocol that uses NDN's Interest/Data

packets to distribute data objects benefiting from NDN's built-in data authenticity [24]. Their work focus on the design of a routing protocol based on four main areas, i.e. a naming scheme for routing and control messages, a security trust model for routing, routing information dissemination and multipath routing.

The basic functionality of NLSR is to discover adjacencies and distribute both connectivity and name prefix information, thinking in terms of data retrieval. Two aspects distinguish NLSR from the previous link-state routing protocols; these are to provide multiple next hops for each name prefix and signing and verifying all LSAs.

This design denominates each router according to the network it resides in, the specific site it belongs to and the assigned router identifier. With this naming scheme is easier to filter out invalid router messages. Periodic *hello* messages are sent to detect link failures, using a router's name as a prefix for the NLSR process.

It has two types of LSA messages, the Adjacency LSA to advertise its links to neighbors and the Name LSA to advertise its name prefixes. Each LSA has a name with a segment */localhop/* which scope limits the LSA to be forwarded to the immediate next hop. The Name LSA contains all the name prefixes registered, while the Adjacency LSA contains all the active links of the router. They are created at router startup time and whenever there is a change in the status of the router's links.

A new version of a router's Adjacency LSA is distributed to the entire network whenever a router establishes or removes a link with a neighbor router. Likewise, whenever any name prefix is added or removed, the router creates and transmits a new Name LSA to its neighbors. Both LSAs are stored in a Link State Database (LSDB) at each node. Moreover, each LSA has a lifetime associated and, when this expires, the LSA is removed from the LSDB.

Based on the information in the Adjacency LSAs, each node builds its own network topology. It runs Dijkstra's algorithm to generate multiple next hops to reach each node. This process is repeated for each link. Knowing which name prefixes are associated with which nodes, a list of next hops can be obtained to reach each name prefix. This design is planned to be further investigated to get a more efficient multipath computation algorithm.

The adjacency establishment works as follows. The router sends periodic hello Interests with its name to each neighbor node to detect its status. If the neighbor responds, it is considered up. Otherwise, the adjacency with the neighbor is considered down, being the cause unknown for the sender router, i.e. link-failure, the neighbor node has disconnected, etc.

This work provides a proactive approach for intra-domain routing, which has

a high impact at startup but it provides the best path via a deterministic choice. Therefore, it does not deal rapidly with content mobility. Moreover, it does not take into account the in-network storage if it deviates from the shortest path stated in its tables.

3.3.2 On Broadcast-based Self-Learning

A different approach to face the forwarding in NDN is to use simple broadcasting techniques. These are based on flooding packets in the network to find the desired content. An example of this technique is *On Broadcast-based Self-learning in NDN* presented by Shi et al. [23], who implemented a more refined version of a broadcast scheme.

The On Broadcast-based Self-learning method is based on the common broadcast mechanism. This method broadcasts the packets across the network when the destination is unknown. Once the first packet is returned, the forwarding tables in the nodes create an entry with the incoming information. Then, later packets intended for that same location are forwarded using a unicast mechanism. This process employs a *discovery flag* that indicates whether an Interest is to be flooded or not. Moreover, the consumer tags an Interest as 'discovery' if it wants it to be flooded or as 'non-discovery' otherwise, allowing the consumer to regulate the flooding frequency.

This design builds forwarding tables entirely in the data plane, without requiring any other control protocols. Besides, Data packets cannot create loops as they follow the traverse path of an Interest. Moreover, fast reaction to link failures is expected by relying on the network retransmission of NACK packets backward, from the node with the failed link to the consumer. Finally, in-network storage is not considered using unicast communication. Therefore, Shi et al. proposed a simple heuristic function that takes outdated information from the CS to give a probable path for the cached data.

The main contribution of this work is the development of an NDN self-learning. A reactive forwarding scheme that applies self-learning to local area networks, intra-domain, which builds forwarding tables in the data plane, recovers quickly from link failures and makes use of off-path caches for Internet contents.

3.3.3 Ant-based algorithm

An ant-based approach is the basis for this work. It can benefit from the properties of swarm solutions in similar architectures, and apply them to adapt to the requirements of NDN.

For the basic behavior, the work from Di Caro and Dorigo on AntNet can be used. Their design employs two different types of agents. Forward ants are the ones responsible for the discovery of the path, using a metric called pheromone and a heuristic function. Backward ants follow the reverse path lead by the forward agent, and they are responsible to update the forwarding tables along the trail. In the NDN context, the forward ant can be part of an Interest while the backward ant can be part of a Data.

One of the peculiarities of NDN is that it uses a hierarchical namespace. On top of it, domains can be used to name data objects from the same producer. This allows the use of name aggregation in the nodes to optimize their internal tables. The work of Wittner et al. [27] gives an idea of how to treat domains and name aggregation in a swarm-based system. They provide a solution implementing different types of pheromones based on the QoS required by each consumer. The similarities between the profiles are comparable to the ones found in the NDN domains. Their solution points to a possible approach to optimize routing using domain similarities in the content names.

In-network caching and content mobility can be addressed by the stochasticity given by a swarm system. As shown by Paquereau and Helvik [19, 20], swarm-based techniques are promising in terms of adaptability when the precise location of the destination is not known or it is not fixed. This also works regarding cached data, as it has the same effects as if a producer would have moved to another location.

This approach may provide a reactive intra-domain routing scheme, with a stochastic decision-making process. Moreover, it is aware of in-network storage and content mobility while it does not require the previous advertisement from the producer of this content.

	NLSR	**Ant-based**	**Broadcast**
Traffic-load	It has a high impact on start-up but a low impact once the network is stable.	There is a medium impact when content is requested.	High impact when content is requested due to flooding the network.
Decision-making	Deterministic decisions are taken with the link information previously obtained.	Stochastic decision. A weighted random choice based on pheromone levels.	Transmit to all interfaces available. When the name is known, unicast transmission.
Content-mobility	Big repercussion due to advertising all the content to the network.	Copes with content moving by means of the adaptability by stochasticity.	Low impact for unknown content, a medium impact for known names via a heuristic function.
Caching-awareness	In-network storage is not contemplated by the routing system.	Caching is considered and used as part of the routing.	Heuristic function to fetch the probable closest copy.
Content advertisement	All content is announced to the whole network	The content is not advertised	The content is not advertised.
Routing Approach	Proactive	Reactive	Reactive
Domain	Intra-domain	Intra-domain	Intra-domain.

Table 3.1: Comparison between NDN routing approaches

Chapter 4
Methodology

This Thesis started as a Specialization Project, where a literature review on Information Centric Networks and Swarm Intelligence and their classes were conducted, shown in Section 2.1 and Section 2.2 respectively. Accordingly, this was the background at the start of this work with the decision to use Name Data Networking as an ICN variant and an ant-based approach for the SI. From this point, the procedure followed is explained during this section and displayed in Figure 4.1.

This MSc Thesis had the main objective to study the possibility to use a Swarm approach for the routing plane in Named Data Networking. With that purpose, two main research questions were stated in Section 1.2.1.

After the Specialization Project, there was an overall understanding of ICN and SI, but a more profound knowledge of the NDN and the ant-based system was needed in order to work with them. These concepts are the foundation over which this Thesis is built and, therefore, a proper study on them has been done in Section 3.

The first Research Question tries to assess whether an SI method can be an alternative routing solution for NDN, while the second RQ focus on the scalability of the system and its performance in the routing and forwarding planes.

To answer the RQs, a theoretical design has been proposed as a possible solution integrating both SI and NDN concepts, and further implemented in a simulator to examine its behavior and performance. To study the scalability and performance

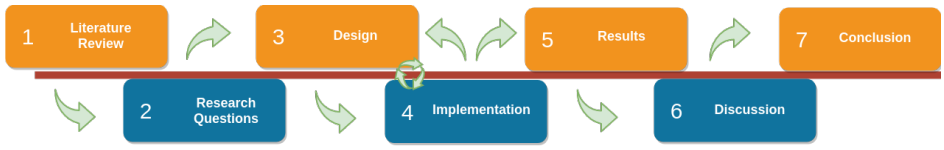


Figure 4.1: Methodology flow

of the implemented system, the experiments count with a real backbone network topology, as explained in Section 5.3.2. Furthermore, a basic flooding system has been later implemented to compare both techniques over this same scenario. This decision is based on the research done in Section 3.3, given that the broadcast system designed by Shi et al. takes also a reactive approach, does not advertise the contents of the producers in the network and it considers the copies cached in the network.

As shown in Figure 4.1, there are several iterations between the Design and the Implementation phases. This is because the design is closely related to how the system should be implemented and, thus, during implementation, improvements to the design are found and realized. Moreover, both design and implementation started with simple routing and along with the iterations, the complexity of the system increased, as can be seen in the experiments in Section 5.3.

Each experiment states its purpose, the data to be collected and the expected results. For this evaluation, alternative approaches have been taken. To prove the correct functionality of the routing and forwarding systems, as well as the implemented architecture, a qualitative approach has been used by analyzing the results determinedly employing raw data and events collection. On the other hand, to study the consistency, scalability and overall performance of the system a quantitative approach has been adopted. Data gathered through several simulations is studied using statistical methods. Three main methods have been used, the maximum values through the simulations, the average measurement of a specific parameter and a 95% confidence interval over this average value. The maximum is used to study the worst-case scenario when treating with tables, which indicates the necessary resources to use the design given. The average measurement, used over retrieval times and retrieved content, is used to give an idea of the value to expect. Although it is important to show this value with the 95% confidence interval, which measures the uncertainty associated with the sampling method. Therefore a tight range is preferred to prove consistency in the experiment outcomes.

When presenting the results, an explanation of what they represent and a brief discussion of each experiment is done. Besides, the discussion section (Section 6.2) focuses on the RQs and how the experiments as a set help answer them and in which way. Finally, the conclusion of this MSc Thesis along with the future work is presented.

There are several limitations when it comes to the implementation done in this MSc Thesis, although the theoretical design might address some of these issues. First, it is the case of an ideal scenario, where no network failures occur, both link and node. Besides, no change of producer location has been contemplated because the caching in the network already shows to some extent the degree of how it affects its

mobility in the system. The use of a self-made network simulator might arise some concerns. This decision was taken to obtain the flexibility and freedom required to design and implement the system needed. The use of a well-known simulator as NS-3[1], with its NDN model[2], would have enabled comparative studies with e.g. NLSR, but due to its steep learning curve and rigidness in the network architecture (due to the NDN model) it was disregarded. To counter these concerns, a second forwarding strategy, based on a flooding technique, has been implemented in the system using the same architecture and parameters as the own design.

[1] https://www.nsnam.org/
[2] http://ndnsim.net/current/

Chapter 5
Simulation Work

This chapter defines the solution proposal given in this project and how to implement and evaluate it. First, the theoretical design is explained and described in different areas and degrees. Then, the implementation section outlines the most important features enforced and the specific peculiarities of how some of the design features have been carried out. Finally, a series of experiments are proposed to evaluate the functionality and performance of this design.

5.1 Design

This section shows the novel design produced as a solution proposal for the objectives of this work. It focuses on two main areas, the changes in the architecture level and the ones at the application level. The architectural changes are intended for integrating the swarm scheme into NDN, while the changes on the application level expect to implement the functionality of the swarm technique.

5.1.1 Architecture level

To start designing a new scheme that adapts Swarm Intelligence into a Named Data Networking topology, the focus is first on the architecture. Taking as a base the current NDN architecture shown by Zhang et al. [29], the most important changes are done at table and packet levels. These modifications do not affect the core concept of ICN but allow the integration of different routing techniques.

Packets The NDN architecture supports two classes of packets: Interest and Data (see Figure 3.1). To adapt swarm routing to this design, additional forms for both packets are sketched. In total, there are four varieties with different behaviors and properties.

- **Ant packet** Ant packets are light-weighted packets that travel over the control plane. Their main function is to update the routing information in the network. The forward ant is an Interest and it contains the content name, its lifetime (in the number of hops) and a nonce to identify the packet. It is responsible to discover the content. The backward ant is a Data and it contains the same information as its Interest, and it does not contain any data. It is responsible to update the pheromone in the nodes.

- **Domain ant** This is a similar packet to the previous one. It is small, light-weighted with routing purposes moving over the control plane. The main difference is that this one does not contain a content name but a domain name. It can be used by both, consumers and nodes, to help to prepare the routing phase with the known domains.

- **Content packet** This is the common packet in NDN. It consists of an Interest, which contains the content name, a lifetime and a nonce. It is used to request specific content from a consumer. The response packet is a Data, which contains the content name and the actual data being requested.

- **Hello packet** As shown by Wang et al. [24], a Hello variant of Interest and Data packet can help in the discovery and advertisement of neighbors. It is used by the nodes in the network to inspect the neighbors' state. It is a light packet with just the necessary information for the router to know their neighbors and their state.

Besides their properties, an important adjustment is the introduction of priority in respect of the class of packet. In this design, ant and content Interests and Datas have different preferences when received by a node. The content packets have higher priority, while the rest of the packets move under the control plane in a low-priority manner. Therefore, content packets (both Interest and Data) are forwarded before any other class, despite arriving at a later moment. This decision is taken from the perspective that ants can utilize the maximum amount of bandwidth as long as they are not affecting the content forwarding and retrieving processes.

Forward Information Base The NDN FIB contains one or more interfaces for each content name stored and, for each interface, a metric to judge the quality of the path. Usually, it is a deterministic value like Round Trip delay Time (RTT), the number of hops or the bandwidth of the channel. To apply a swarm solution on routing we can use the same structure on the FIB using the metric entry to save the amount of pheromone on each pair interface-name in the table. The amount of pheromone will provide a measurement that gives information to where content is more probable to be found. The case where several interfaces have high pheromone

levels can indicate the existence of the specified content in several locations or similar redundant paths to the same location.

Pending Interest Table and Pending Agent Table Having different types of Interest and Data packets helps to discern what each one of them is doing and to adapt better to the work they are supposed to accomplish. In this matter, the forward agents as Interest packets and the backward agents as Data packets fit in the structure of NDN. This suggestion tries to adopt a swarm approach to the NDN routing paradigm. While the Interest packets are forwarded towards a possible solution or destination, they fill the Pending Interest Table with some information like the interface they come from together with their names. Once they reach their destination, this one returns a Data packet, namely backward ant, towards the source. This Data packet uses the information from the PIT to travel the forward ant's reverse path while updating the information on the FIB table regarding the original destination, the taken interface towards it and the amount of pheromones. Even though Interest and Data packets are used as forward and backward agents, the packets itself are different from the usual content Interest and Data packets. Therefore, instead of using the same PIT table for both classes, it is more practical to separate them on two different tables: Pending Interest Table (PIT) for the content retrieval and Pending Agent Table (PAT) for the swarm routing system. This action is taken due to the way both tables are filled. The PIT creates an entry each time an Interest arrives, but in the case it is an Interest with an existing name, the PIT just adds the incoming interface to the entry. While in the PAT, for each incoming ant Interest, an entry is created. This is since content Interest are aggregated to minimize link and processing consumption while ant Interest are forwarded to improve the routing system capabilities. It is also important to note that in a real implementation, these two tables could be easily combined to fit the existing NDN architecture, and therefore optimize the processing of packets as they would be done in the physical layer instead of in the application one.

5.1.2 Application level

For routing, done in the control plane, the NDN network allows any type of algorithm to work with it. There is no limitation on the technology to use rather than to stick to the NDN architecture. Thus Interest and Data packets must be used for communication and retrieval of data, the Content Store can be used for caching, the Pending Interest Table must be used to have a control of the different Interest packets that have gone through the node and the Forward Information Base must be updated with the routing information obtained by the routing algorithm, if one. There can be additional structures to help with the routing, but the main NDN structure must be followed.

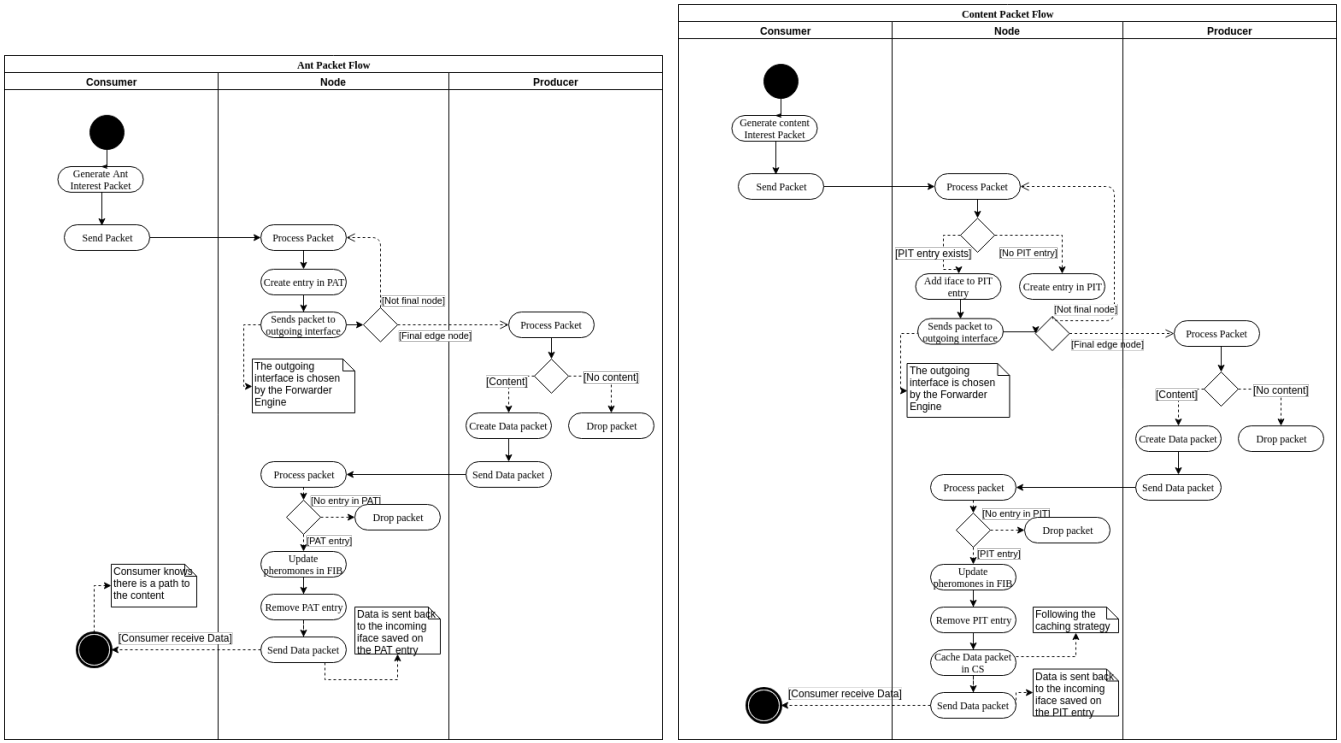
Routing-forwarding strategy

The routing strategy for content and control packets will be different as well. Although they will use the same basis when it comes to route the packet, the insights of the function will differ. That is because the requirements for the control plane are different than for the forwarding plane. While the ants must try to discover the optimal path, at the same time that they provide backup paths, the content needs to be retrieved in a fast and stable way. Therefore, the ants will have a stochastic strategy that will lead to discovering new paths rather than converge to the optimal one, while content packets will use a stochastic strategy endorsing the best route known at the time. Note that in both cases the strategy is stochastic, which means that there is a heuristic function in the decision process. This makes the choice non-deterministic and, therefore, the packets will not always follow the same direction towards the same destination. Another difference is how the Interest tables work. On the one hand, when an Interest arrives at the PIT, this will forward it and save the related information. Besides, all incoming Interest with the same name will be saved in the table but not forwarded ahead. On the other hand, when an ant Interest arrives at the PAT, this will be forwarded and saved with the corresponding information. In this case, consecutive ant Interest packets with the same name will be also forwarded. This difference in actions tries to empower pheromone laying on the trails to destinations while optimizing resources when requesting content. It is important to mention the importance of the stochastic method on the content routing, as the NDN architecture allows for in-network storage, i.e. caching, so there might be content on a node diverging from the optimal path. Hence, a non-deterministic approach discovering an alternative path to content is interesting.

Routing process

The Consumer is the one responsible to generate the ant and content type of Interest packets. This segment explains the procedure of routing, forwarding and retrieving of a specific content c as shown in Figure 5.1.

When the Consumer wants to retrieve a certain content c, it first generates ant Interest packets with the name of the content c and forwards them to the immediate node. The router receives the Interest packet through the incoming interface and processes it. It creates an entry on the PAT table for the Interest with the name, incoming interface and the id of the packet, then the Forward Engine decides to which outgoing interface should it be forwarded and, finally, the node sends the packet towards it. If the interface belongs to a router, it follows the same steps explained, otherwise, it means that it has reached the Producer. When this happens, the Producer checks whether it has the content c requested. If that is the case, it responds to the same edge node with an ant Data packet which contains c's name. When a node receives an ant Data packet, it looks up in the PAT table to see if there



(a) Flow of ant packets

(b) Flow of content packets

Figure 5.1: Routing packet flow

is a match. In that case, the router checks if there is a FIB entry with the destination name and, if it exists, it updates the information regarding the incoming interface and the amount of pheromone or, otherwise, it creates it. It gets the incoming interface of the Interest from the PAT entry and it sends the ant Data through it. When the ant Data arrives at the Consumer, this knows that a path exists in the network to retrieve the actual content.

Therefore, it can now send a content Interest to acquire the desired data object. The Consumer sends an Interest with the content c's name to the immediate node. Once the router has received the packet, it searches in the PIT table whether it exists an entry with the same name. In this case, it updates the information about the incoming interface and waits for an answer to that Interest. If it does not exist, it creates a new entry on the PIT table and the Forwarding Engine decides where to forward the packet taking into account the information saved on the FIB table. After that, the router sends the packet through that interface. When, ultimately, the packet arrives at the Producer, this one checks whether it possesses the content. If it does, it generates a Data with the name of content c and the data object and it sends it to the interface from where the Interest arrived. When a router receives a Data containing a data object, it checks if there is an entry on the PIT with the same name. If so, it updates the FIB table as the previous process. Then, as the PIT can have several interfaces aggregated to the same name, it takes these interfaces, one or more, to send the packet. Finally, it removes the entry from the PIT. Later, the Consumer receives the content requested.

This process is presented in Figure 5.1. The first procedure applied, where the routing with ants is done, is offered in Figure 5.1a where the behavior of a single ant is shown. An ant burst is sent, i.e. several ants are sent in a constant time interval, for example, 10 ants leave with a 1ms gap each, and then, they roam the network concurrently. When an ant has been able to successfully find a path to the content, the Consumer can request the data object. Then, the process in Figure 5.1b is started.

Specific choices

This section shows in more detail some of the peculiarities of the overall design. It exposes some of the ideas introduced above more in-depth so it is easier to follow with the implementation and results in a later section of this Thesis.

Domain sharing Routers can assist in the routing by sharing the router domains between the different nodes. When the network topology is started, the nodes can request the known domains to help the consumers retrieve the data using a more

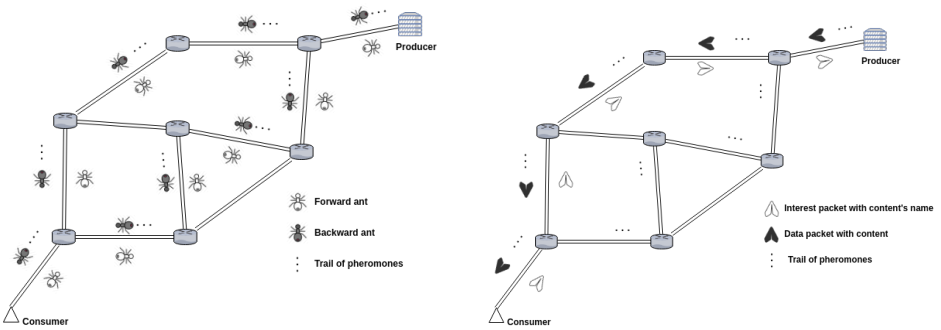
direct path. This approach tries to minimize the convergence time when consumers request content for the first time.

To implement this alternative, it is necessary to develop a new type of Interest and Data packet since the destination, in this case, is not a specific data object but a domain. Thus, it is the proposed *Domain ant*, an ant-like packet with a domain destination with the same or very similar capabilities; stochastic discovery of a path, use of the PAT, update the pheromone amount in the FIB using the returning Data packet.

Prepare the network Further work on this design could implement an optimized version of the previous feature. For a consumer to speed up the content retrieval, it can use the hierarchical naming scheme in NDN. For that purpose, it can have a list of domains saved which can be requested at the consumer's boot-up time. These domains would be the most common ones used by the consumer, therefore it would help to accelerate the request for content. Afterward, Interest packets with the domain would be sent out from the consumer node to the network, and wait for the corresponding Data packet to return. This way when content from that domain is requested, the network already has a working path to the destination's domain, helping on the discovery phase.

Content request arrangement To be able to route an Interest to the content, it is necessary to have some information regarding the data object in the network. Therefore, one way to do it is to send first some ants to discover an active path to the content and afterward, to send the Interest to retrieve the data (shown in Figure 5.2). First, several forwards agents are sent over the control plane to discover a path to the content with the specified name. These ants move around the network following the strategy designated and, once they find the content, they return as backward agents through the same path, updating the tables in the network nodes. Then, the network already has a working path to the data and, hence, the consumer can send an Interest to retrieve that content. The Interest is sent following the forwarding strategy on each node, and once the Interest is satisfied, the content is returned following its reverse path.

Neighbor discovery To be able to route packets, the nodes in the network need to know who their neighbors are and whether they are up and running or they are down. For this purpose, another type of packets is used, the Hello packet. It follows a similar mechanism than the one used by Wang et al. [24]. The Hello packet is used to discover if there is a neighbor connected at the other end of a node's link and, if so, it obtains its name for routing purposes.



(a) Ant packets flow (b) Content packets flow

Figure 5.2: Content discovery and recovery

Stagnation mitigation The method chosen to control stagnation is evaporation, as introduced in Section 3.2.2. This approach tries to imitate the natural system giving a close animal behavior solution through an exponential distribution.

5.1.3 Flooding scheme

In order to compare the design exposed above, another forwarding strategy has been implemented. From the variants explained in Section 3.3, the chosen one is the broadcast system from Shi et al.. Their scheme is the base for this design, with similar core concepts but a rather simpler approach in its characteristics.

Its behavior is the basis for a flooding technique. For each name the consumer wants to retrieve, an Interest packet is sent. When a node receives an Interest, it checks in the CS whether it has the content cached. If no content is in the store, it checks the PIT table for previous Interests with the same name and, if no entry matches the content name received, it forwards the Interest to all available interfaces except the incoming one. This solution still applies all the NDN capabilities and it is adapted to its architectural peculiarities. The in-network storage is enabled, Interest and Data packets are used, the PIT table manages the incoming Interests and it uses the hierarchical name scheme.

5.2 Implementation

This section describes the implementation process undergone to realize the design explained in Section 5.1. The implementation uses Python[1] version 3.6 as a programming language, which provides high flexibility and integrates systems more effectively. The structure is in the form of a library so the different modules can be used to

[1]https://www.python.org/

execute distinct scenarios. The files composing this implementation can be found in the online repository of the project[2].

To be able to run the scenarios and obtain their results, the library uses a simulator to provide queue and time management capabilities. The module used for that purpose is SimPy[3], a process-based discrete-event simulation framework that implements an environment linking events with the time plane.

5.2.1 Components

Several components comprise the library. The base for these components are elements in the architecture level(see Section 5.1.1), while the functionality implemented in each module are given by the application level (see Section 5.1.2).

Packet The Packet is the container for the Interests and Datas, both ant and content based. It is the unit of information that carries not only the content when required, but also the necessary information to achieve routing and forwarding. It is identified by a unique Nonce, which is decided by architectural design in NDN and it is the key to avoid loops in the network.

Interface The Interface module is responsible to forward the packets between nodes in the network. The bandwidth assigned to the interface is the metric which gives the transmission delay for each packet. It also implements a queue of received packets to be transmitted.

Consumer It is the module that requests content via the one Interface it contains with the network. Two procedures are expected; on the one hand, if the content name requested refers to the actual content, the data object is retrieved. On the other hand, if the content name requested is not a match of existing content, the response received is a data object with the meta-data naming the actual content.

Producer This module distributes data objects assigned to content names. If a request matches an existing object, a Data is returned. Otherwise, a Data is created with the meta-data filled with the name or names of the existing content using the longest name prefix matching strategy profiting from the hierarchical name scheme and the existence of domains.

Node The node is the core object in the network. It is composed of a FIB, a PIT and a PAT tables, a Content Store and one or more Interfaces. It is responsible for

[2]https://github.com/adriadescamps/swarmNDN
[3]https://pypi.org/project/simpy/

the routing and forwarding processes, and it can implement different strategies to achieve that working over the same architecture.

Forwarding engine This module is responsible to select the Interface where the packet is sent. Depending on the strategy stipulated by the Node, it can be a deterministic decision or a stochastic one given a certain metric.

When the decision process is stochastic, it uses the Algorithm 5.1. If the FIB contains an entry with the Interest name, it takes the interfaces in that entry, otherwise, it takes an aggregation of the interfaces containing a portion of the domains of the name (taking advantage from the hierarchical name scheme). Depending on which case takes place, the algorithm assigns a different value to the *pwr* variable, which will serve as a parameter for the Heuristic function. Finally, in case no entry in the FIB matches totally or partially with the incoming Interest, the function returns randomly one of the Node's Interface. Otherwise, it returns the response of the Heuristic function.

Algorithm 5.1 Forwarding engine's algorithm

```
1: function FORWARD ENGINE(pkt)                    ▷ The outgoing interface for pkt
2:     if pkt.name ∈ FIB then                      ▷ One exact match of name in the FIB
3:         if pkt.ant == True then
4:             pwr ← 1.5
5:         else
6:             pwr ← 2
7:         end if
8:         ifaces = FIB[pkt.name].outgoings        ▷ ifaces is the matching entry
9:     else                                        ▷ One partial match of name in the FIB
10:        ifaces = FIB.domain[pkt.name]           ▷ ifaces has all partial matches
11:        pwr ← 1
12:    end if
13:    if ifaces == ∅ then
14:        return RANDOM(interfaces)               ▷ Return a random Node's interface
15:    else
16:        return HEURISTIC(pwr, ifaces)           ▷ Return interface given by Heuristic
17:    end if
18: end function
```

The Heuristic function mentioned above is implemented in the Algorithm 5.2. This method gives an interface selection scheme based on a probability distribution given by

$$p(i) = \frac{pheromone_i^{pwr}}{\sum_{j \in ifaces} pheromone_j^{pwr}} \tag{5.1}$$

where $p(i)$ gives the probability of interface i being chosen as outgoing interface. The *pwr*, assigned in Algorithm 5.1, reinforces the strength of *pheromone* in the forwarding decision. This compromise tries to boost the shortest path for content Interest while allowing ant Interest to be more arbitrary and promote the discovery of alternative paths. Finally, in cases where a domain match occurs, a lower *pwr* is employed so the experience from previous agents is taken into account but with a certain reluctance.

Algorithm 5.2 Heuristic's algorithm

```
1: function HEURISTIC(pwr, ifaces)          ▷ One interface in ifaces given pwr
2:     total = 0.0
3:     for pheromone ← ifaces do
4:         total+ = i^pwr
5:     end for
6:     rand =RANDOM(0.0, total)             ▷ Uniform distribution
7:     for iface, pheromone ← ifaces do
8:         if rand < pheromone^pwr then
9:             return iface
10:        else
11:            rand− = pheromone^pwr
12:        end if
13:    end for
14: end function
```

Tables The function of router tables is to store the information required for the routing and forwarding states.

- **Forwarding Information Base** It contains the content name, the outgoing interfaces for that name and a metric for each interface to be used during the forwarding action.
- **Pending Interest Table** It aggregates for each content name all the incoming interfaces from where it was received the corresponding content Interest. This information is used to hand back the Data matching the received Interests for that specific name.
- **Pending Agent Table** Similar to the PIT table, it differs in the way incoming ant Interests are handled. For each one received, it creates a new entry in the table so all incoming Interests are forwarded further in the network.
- **Content Store** It is the database where the data is stored giving the nodes the capability for in-network storage.

5.3 Experiments

This section displays and explains the experiments realized in this project, summarized in Table 5.1. All of them focus on specific goals that try to validate the expected behavior. As the objectives are different among procedures, the gathered data will also vary. The experiments are presented in increasing order of complexity regarding implementation and objectives.

	Topology	Consumers	Producers	Content Names	Simulations
Experiment 1	Simple	1	1	2	1
Experiment 2	Simple	2	1	2	1
Experiment 3	Simple	2	1	2	1
Experiment 4	Simple	2	1	2* 10 - 20 - 30 chunks	1000
Experiment 5	Real-world	30	3	2 * 10 chunks	20 - 200
Experiment 6	Real-world	Random	Random	2 * 10 chunks	20 - 200

Table 5.1: Summary of experiments

The experiments mentioned above are divided into two main scenarios. Each scenario has a specific network topology adapted to the objectives of the related experiments. Therefore, the focus of the experiments in both scenarios is distinct.

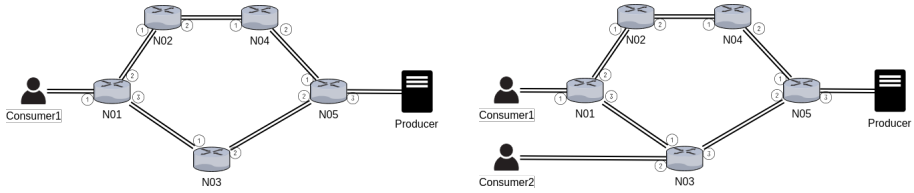
Every experiment has several variables, presented in Table 5.2, susceptible to be tuned either to modify the stochastic behavior of the system or to optimize its performance. Some regarding the routing process and some the forwarding.

Stochastic behavior	
Ant generating rate	Rate at which ants are generated and sent
Ants per request	Number of ants sent before requesting the content
Pheromone laid	Amount of pheromone left by each backward agent
Pheromone reduction	Constant used by the evaporation process to reduce the amount of pheromone in the interfaces
Evaporation rate	Rate at which the pheromone has to be reduced
Pheromone reinforcement *pwr*	Value used to boost pheromone influence in the heuristic method (Eq. 5.1). Variables: Ant, Content or Domain
General behavior	
Packet size	Size of the Interest and Data sent
Packet lifetime	Time To Live for the packets in number of hops
Table timeout	Time before an entry is deleted from a table
Seed	Seed used in the simulation

Table 5.2: Variable descriptions

5.3.1 Scenario 1 - Routing behavior

This scenario concentrates on testing the basic routing principles and requirements in the implementation. Using diverse executions targeting different goals, this scenario expects to obtain convincing results of the correct operation of the routing system. It takes an incremental approach in the tests, with an increased complexity on each execution. The topology used can be seen at Figure 5.3. It allows us to study in detail each event in the simulation and follow all the steps in the routing process.



(a) One consumer (b) Two consumers

Figure 5.3: Network topology - Scenario 1

This scenario uses the variables in Table 5.3 as a default values. In case any experiment deviates from the information shown in this table, it is stated in its constraints field.

Variable	**Value**
Ant generating rate	Exponential distribution with a mean of 0.9 units
Ants per request	10 ants
Pheromone laid	1 unit
Pheromone reduction	0.1 units
Evaporation rate	Exponential distribution with a mean of 1 unit
Pheromone reinforcement	Ant: 1 Content: 2 Domain: 0.5
Packet size	100 bytes
Packet lifetime	20 hops
Table timeout	30 units
Seed	2

Table 5.3: Variables Scenario 1

Experiment 1

This scenario focuses on testing the basic routing principles and requirements in the implementation (Table 5.4). Therefore, just ant packets are used and no content is requested. This decision comes from the understanding that a correct routing system must be implemented to be able to retrieve content objects.

Experiment 1	Study of the ant behavior in the system
Description	A consumer carries out the routing process previous to request content by sending a burst of ants towards the network using a specific content name.
Components	One static consumer and one static producer.
Constraints	Default values. Topology as shown in Figure 5.3a
Objective	The goal in this experiment is to validate the correct behavior of each module, with a specific focus on: – Whether the path discovery is correctly performed by the ants. – How the node's tables are filled with information. – Whether the backward agent lays the correct amount of pheromone in the chosen path by the forward agent.
Data gathered	The data gathered to study the results of this experiment is associated with the monitoring of the PAT and FIB during the operation of the ants. Therefore, the number of entries in the PAT, as well as the entries and amount of pheromone for each entry in the FIB are collected. Besides, the events triggered by the simulator are stored so every single occurrence can be studied. The events are stored in case of requiring to follow a flow of packets.
Expected result	The outcome should show the PAT table being filled by the forwarding ants during the discovery period, and then being emptied and the FIB filled during the retrieval period. Further, each backward agent should trace back the path discovered by its forward agent.

Table 5.4: Experiment 1. Ant behavior in the system

Experiment 2

Similar to the first scenario, the focus in this experiment (Table 5.5) is on the routing accuracy. It differs by adding another consumer in the topology, being two sources sending ant Interests with the same content name.

Experiment 2	Study of the harmony between consumers
Description	Two consumers carry out the process previous to request content in order to fill the network with routing information by sending ant Interests.
Components	Two static consumers and one static producer.
Constraints	Default values. Topology as shown in Figure 5.3b
Objective	This scenario tries to study whether two consumers can work together to improve the routing performance in a network. Specifically, the goals are: – Do the ants generated by different consumers operate jointly? – Does the path discovery improve with two consumers?
Data gathered	Similar to the previous scenario, the data gathered is the evolution of the PAT and FIB tables and the simulator events. In this case, the progression of the node's tables is the most valuable data to compare with the previous results.
Expected result	It is expected to obtain evidence that having more agents in the network helps the routing system. The PAT should fill more rapidly than in the previous experiment, while the FIB entries should experience faster growth in terms of pheromone amount.

Table 5.5: Experiment 2. Synergy between consumers

Experiment 3

This scenario, described in Table 5.6, is the first one which includes data. That means that not only content is requested and expected to be served, but the nodes in the topology enable the in-network storage. When the consumer wants to request content, it sends first the burst of ants and, then, the content Interest to request the data object.

Experiment 3	Analysis of Data packets and in-network storage capabilities
Description	Two consumers request content adopting the full process. They first send the ants for routing purposes and afterwords they request the data by sending the content Interest.
Components	Two static consumers and one static producer.
Constraints	Default values. Topology as shown in Figure 5.3b
Objective	The goals in this simulation are to inspect the behavior of the system when data is involved. It is important to validate that the content Interests and Datas follow the expected behavior, as well as the proper implementation of the in-network storage capability in the nodes as Content Store. Regarding caching, it is interesting to see how the requests from the first and the second consumers differ in terms of the retrieval time of the content and whether the use of caching in the nodes is an improvement.
Data gathered	This experiment collects three types of data. First, as the previous experiments, the evolution of the tables during the execution of the experiment. Then, the packets received by the consumers with their fetching time. Finally, the list of events produced by the simulator and the data associated with them.
Expected result	The foreseen results are the consumers receiving the data object via a Data. The forwarding engine in the nodes should accentuate the pheromone value when deciding the interface to forward the Interest. In regards to the caching, we expect to see a lower fetching time from the second consumer expecting the content to be retrieved from a node in the network instead of being retrieved from the producer.

Table 5.6: Experiment 3. Analysis of Data and in-network storage

Experiment 4

This experiment, explained in detail in Table 5.7, uses the same logic as the previous one but adding a more realistic practice. Instead of requesting a single file as in the past experiments, in this case, the content is divided into chunks of data. This experiment simulates a real-life case where a video file is sliced in several blocks. The consumer makes the request with the known general name of the content and the producer answers with a list of names corresponding to the chunks forming the file in the meta-data of a Data packet. Then, the consumer is able to request the chunks of data composing the whole content.

Experiment 4	Video and audio request
Description	Two consumers request data using a generic name and, after receiving the Data response, they request the chunks composing the content using their specific names. There are three executions in this experiment, with a varied number of chunks per content. Moreover, to try to provide high consistency to the results each execution is simulated a 1000 times, changing the seed.
Components	Two static consumers and one static producer. It uses the topology in Figure 5.3b.
Constraints	Ant generation rate: 0.5 units Ants before sending chunk request: 30 units Content packet size: Uniform distribution between 1500 and 2000 bytes Table timeout: 20 units Chunks per content: 10, 20 and 30 Seed: From 0 to 999
Objective	This experiment has several objectives. The first one is to study an experiment with a closer behavior to reality. The three executions with a variant number of chunks are to study the balance of the PAT table. The goal is to examine whether the number of entries in the PAT table stabilizes with the increment of chunks requested, or on the contrary, it increases comparable to the number of chunks. Furthermore, not only in the routing plane is important to be reliable but also in the forwarding one. Therefore, it is an objective to study how the fetching times of content evolve during the three executions of the experiment and how consistent they are along with the simulations.
Data gathered	Similarly to the previous experiment, three main groups of data are collected. The PAT information regarding the number of entries it has, the packets received by the consumers and their fetching time and, finally, the event logs from the simulator. In this case, because of the amount of simulation per execution, the information varies a bit. The saved values are the maximum entries at the PAT table, the average of the times per content name together with the 95% confidence interval to provide consistency in the results.
Expected result	The evaluation on the PAT table should give one of two outcomes, having a notable increase in the number of entries and then stabilizing in the time domain until the Interests are served, or having a continuous increase of the entries until the Interests are served. The first outcome being the preferred as would mean a balanced utilization of the table resources. Concerning the content retrieval time, it is expected to become constant throughout the simulations among all the names. A good result would be having a small 95% confidence interval related to the total fetch time.

Table 5.7: Experiment 4. Video and audio request

5.3.2 Scenario 2 - Real-life behavior

This scenario uses a real topology (see Figure 5.4) from Uninett, [4] the Norwegian National Research and Education IP backbone network (NREN). Its purpose is to study the effects and performance of the design in a real network scheme. This network topology consists of nodes with several links and variant bandwidth. The nodes are located in a geographical position and they are assigned to a specific area. Moreover, the areas marked in the topology are applied by the link state routing system currently operating the real network. These areas are used in the implementation as domains, so name aggregation can be applied and the hierarchical namespace tested. These areas or domains are shown in the figure with different colors, being one color assigned to one area.

Within this topology, two experiments are performed. The first one focuses on the implementation of the Swarm routing over the NDN in a real topology, to evaluate its performance. Further, the second experiment studies the performance of ant routing compared to basic flooding based routing. For these experiments, the values of the variables exposed at Table 5.2 are different from the previous section as they must be adjusted to adapt to the new scenario. Therefore, it uses the parameters indicated in Table 5.8.

Variable	**Value**
Ant generating rate	0.2 units
Ants per request	20 ants
Ants per chunk request	0 ants
Pheromone laid	1.5 unit
Pheromone reduction	0.05 units
Evaporation rate	Exponential distribution with a mean of 1 unit
Pheromone reinforcement	Ant: 1.5 Content: 2 Domain: 1
Ant packet size	Uniform distribution between 80 and 100 bytes
Content packet size	Uniform distribution between 1500 and 2000 bytes
Packet lifetime	100 hops
Chunks per content	10 chunks
Table timeout	1500 units
Seed	2200 + simulation number

Table 5.8: Variables Scenario 2

[4] https://www.uninett.no/
[5] https://github.com/Uninett/PyMetric

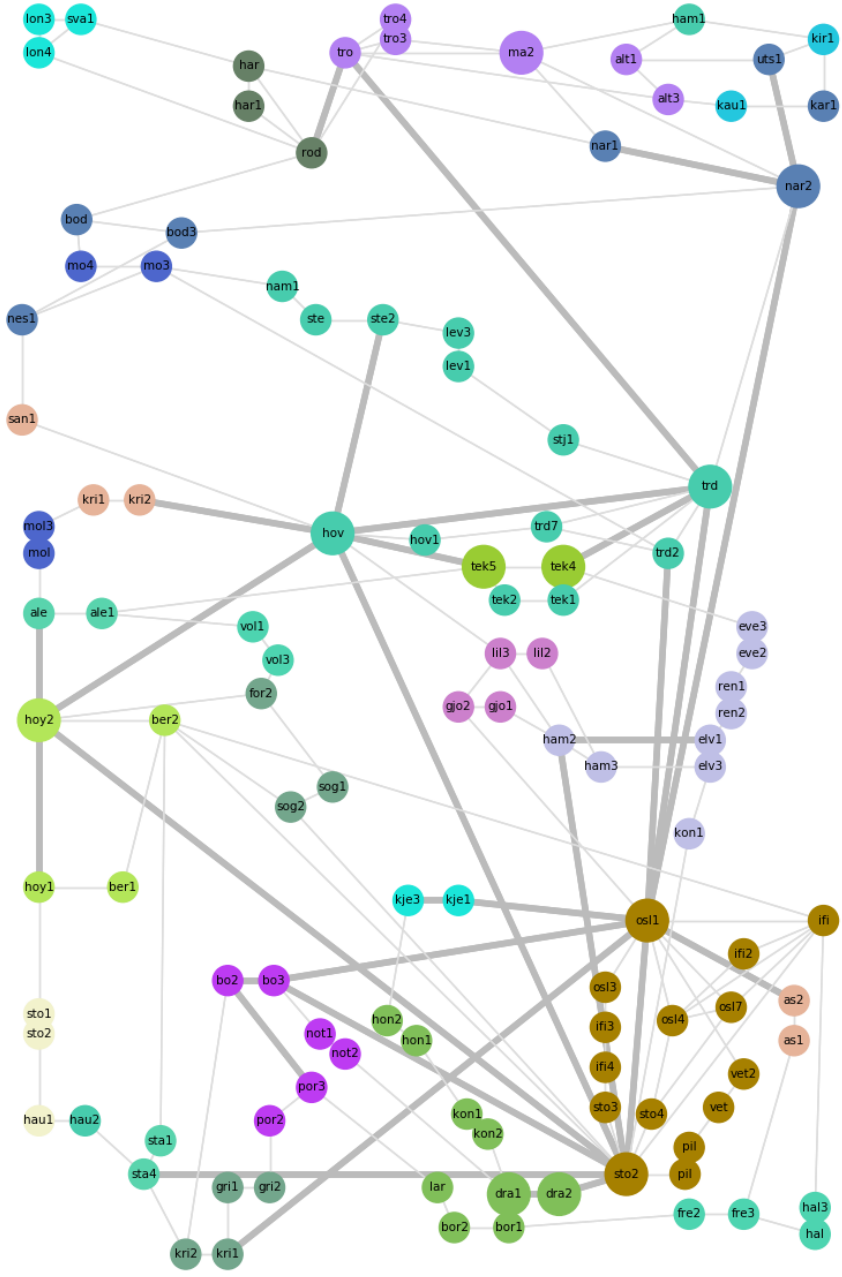


Figure 5.4: Network Topology - Scenario 2. Created with PyMetric[5]

Experiment 5

This experiment is the first one to use the real topology to evaluate the routing and forwarding in the NDN architecture buy the ant-based approach. It includes two executions, one employing 20 simulations containing 30 consumers and 3 producers randomly allocated, and one employing 200 simulations with the same conditions than the previous execution, in order to validate the stability and consistency of the system.

Experiment 6

This is the last experiment in the project. It is the general evaluation of the design and implementation realized by comparing it to a known flooding system. This experiment implements both the swarm and the flooding methods, with the same variable values and properties. The generation process applies the same seed for both approaches giving the exact same topology, consumer and producer location, and characteristics. Therefore, it is a fair comparison between them.

It has two different executions, the first one featuring 20 simulations with a fixed number of 30 consumers and 3 producers, while the second one features 200 simulations with a random number of consumers between 20 and 50 and of producers between 2 and 5. The purpose of these two executions is to compare the performance of both within a fixed scenario as the first one and to analyze the consistency of the techniques when a bigger load is produced through the network in the second one.

Experiment 5	Uninett topology
Description	This experiment tries to simulate a real network topology running a swarm method for the routing state. It also integrates and implements the domain aggregation property of NDN, as well as the general architecture. The nodes initiate the bootstrapping process of the domains at start-up to improve the routing system. Later, the consumers around the topology request the data objects and the producers, in the 'Trondheim' domain, serve the petitions received. All aspects of the design are implemented in this experiment to evaluate their performance.
Components	30 consumers with 3 producers
Constraints	The 30 consumers are randomly located in the topology map. The 3 producers are randomly located in the 'Trondheim' area or domain. The Shortest Path is calculated using Dijkstra and the number of hops as a metric. The number of simulation is 20.
Objective	The goal of this experiment is to validate the Swarm design implemented in a real topology. Confirming the correct functionality of the routing and forwarding states, the retrieval of content by the consumers, the delivery from the producer and the proper distribution by the routers in the network. More specific objectives are: Study the effects of the larger topology over the amount of content successfully retrieved by the consumers and the number of wasted packets generated. Consider the content retrieval time per name and its consistency through a 95% confidence interval. Analyze the accurate operation of the domain sharing process (explained in Section 5.1.2) realized by the nodes in the network.
Data gathered	This experiment generates a great amount of data. The times for each packet received by every consumer in all simulations is stored. The amount of data objects received by each consumer in all simulations. The wasted packets for each simulation categorized in: Data lost, Interest lost and timeout lost. The number of names the producers receive in each simulation. Every event triggered by the simulator.
Expected result	The overall expected result is the consumers receiving the requested content. In more detail, the retrieval times for content should be stable between a reasonable limit throughout the simulations. Besides, the number of data objects received by the consumers should be around the maximum possible, and the wasted packets (Data and Interest) should be a genuinely small portion compared to the received ones.

Table 5.9: Experiment 5. Uninett topology

Experiment 6	Comparison between Ant-based and Flooding routing
Description	This experiment integrates all the characteristics of the previous one. It adds a new forwarding strategy, flooding, which uses the same architecture implemented. Two executions are part of this experiment, the first one with 20 simulations and a second one with 200.
Components	Variant in relation with the execution.
Constraints	The Shortest Path is calculated using Dijkstra and the number of hops as a metric. For the 20 simulations execution: The 30 consumers are randomly located in the topology map. The 3 producers are randomly located in the 'Trondheim' area or domain. For the 200 simulation execution: A number between 20 and 50 consumers are randomly located in the topology map for each simulation. A number between 2 and 5 producers are randomly located in the 'Trondheim' area, or domain, for each simulation.
Objective	The main objective is to compare the performance of both methods. An important goal is to have a similar stretch, i.e. the relation of hops with respect to the shortest path from consumer to producer, between the two techniques as the flooding system always retrieves the content from the closest location possible. Therefore, having a similar stretch would mean that the swarm algorithm is able to retrieve the content close to the optimal path. The difference between the load exert in the network by the two options is important to examine, as it is one of the drawbacks of the broadcast system. Thus, the goal is for the swarm system to be noticeably better than the flooding in terms of performance, content retrieved and packets wasted.
Data gathered	Similar to the previous experiment, the data gathered by the two methods is the same. In addition, it has the data from both executions which reflects the evolution of both systems when the demands increase.
Expected result	The result expected for this experiment is a better performance by the swarm routing system in terms of content retrieved, content wasted and the load in the network. Besides, the flooding system is expected to have better results in the stretch in terms of length and time.

Table 5.10: Experiment 6. Comparison between Ant-based and Flooding routing

Chapter 6
Results and Discussion

Two main sections structure this chapter. The first one shows the results obtained from the experiments realized after implementation, along with an explanation of what they are and how to interpret them, with a brief discussion a the end of each piece. The second section discusses the two main scenarios, giving an insight regarding the Research Questions stated at Section 1.2.1 and based on the results acquired.

6.1 Results

This section shows the results for all experiments in Section 5.3. Each experiment results are explained with the help of figures and illustrations followed by a short discussion of what these results convey. Several experiments use similar figures, therefore, their meaning and composition are explained in detail the first time they are introduced and, in successive appearances, only the results are explained. The discussion carried out in this section is directly related to the objectives and expected results for each individual experiment.

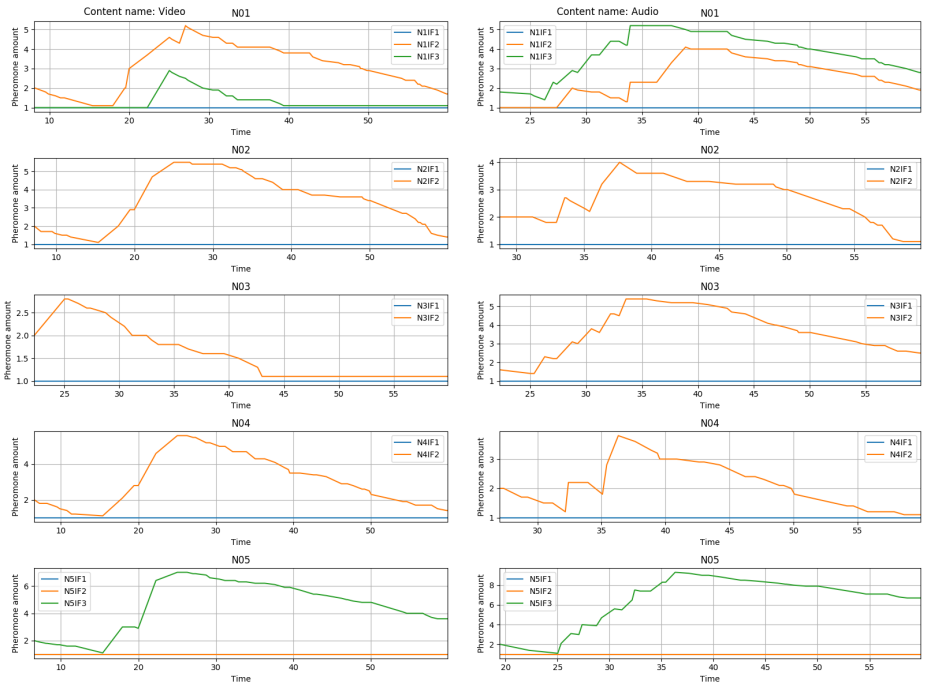
All figures shown in this section, together with the raw data used to generate them and some further information files can be found under the online repository where the project is hosted[1]. Note that the image resolution of the figures is better in the original files.

6.1.1 Experiment 1

The description of this experiment can be found in Table 5.4. From the information stated there, three data elements are collected, the pheromone evolution in the FIB (Figure 6.1), the evolution of the PAT (Figure 6.2) and the simulation events, not shown in this report due to their extension.

[1] https://github.com/adriadescamps/swarmNDN

The pheromone evolution is divided in two figures, Figure 6.1a and Figure 6.1b, based on the 'video' and 'audio' content names n, respectively. The figures show pheromones p along the y-axis and time t along the x-axis. Note that the different sub-figures, indicating the nodes, start at a different point in time t due to the nonexistence of data until the moment the plot starts. The pheromones in the FIB are affected by the backward agents, or ants. Therefore, when these ants are moving backward they increase the pheromone values p in the nodes. Notwithstanding, pheromones decrease as a result of evaporation, which is a non-deterministic method to reduce its value. This is the reason behind the irregular reduction of pheromone levels in the nodes.



(a) Pheromone in the nodes for $n = video$ (b) Pheromone in the nodes for $n = audio$

Upper path. $N1.IF2 \rightarrow N2.IF2 \rightarrow N4.IF2 \rightarrow N5.IF3$
Lower path. $N1.IF3 \rightarrow N3.IF3 \rightarrow N5.IF3$

Figure 6.1: Experiment 1. Pheromone evolution

In the first figure, using the topology in Figure 5.3a, the first ant returns around the second 7, as it can be seen at node *N05* and interface *IF3*. Then, node *N04* receives it through its interface *IF2*, and later, node *N02* and node *N01* before sending it to the consumer. It can be observed that, on each node, the pheromone value on the incoming interface has increased (to 2 as 1 is the default value) at the

time the agent was received. Node *N05* and *N01* are endpoints in the topology, thus, they receive all ants, resulting in the highest pheromone levels. On the other hand, the two paths between *N01* and *N05* share the load of the traffic, as seen by the lower pheromone amount. Moreover, the levels of pheromone on each path relate to the levels of the interfaces in *N01*.

The PAT shows the number of ants moving through each node in the network. Every time an agent arrives at a node, a new entry is created, while they can either be removed by having the same ant returning as a backward agent or dropped when a timeout is triggered in the entry.

Figure 6.2 shows entries e along the y-axis and time t along the x-axis. The node *N01* is the first in increasing its size, as it is the endpoint where the consumer is connected, while *N05* is the first one in decrease due to its proximity with the producer. This fact also affects the evolution of e, as *N05* sends and receives the related forward and backward agents faster than the other nodes in the network, resulting in lower resource consumption.

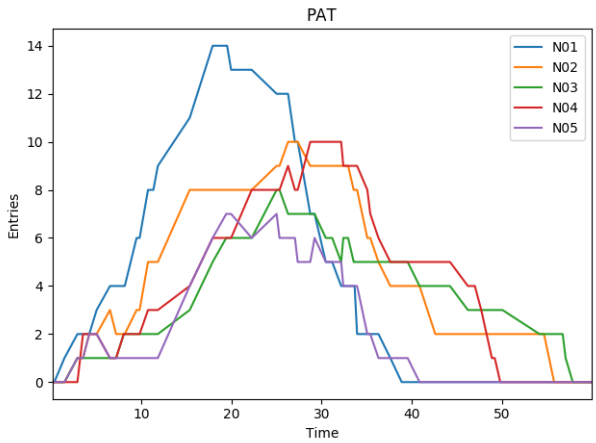


Figure 6.2: Experiment 1. PAT evolution

The two figures are directly related; when an ant arrives at a node, an entry from the PAT is removed while an entry from the FIB increases the pheromone levels. For example, the first ant leaving a pheromone trail also removes the entries in nodes *N05* and *04*, as seen in Figure 6.2. But the pheromones stop increasing just before time $t = 40$ while the entries $e > 0$ in the PAT, which are completely removed at time $t = 60$. The reasons for that behavior are multiple; first, some ants can die because of their TTL if they do not find the producer, thus not being able to return as backward agents to remove the PAT entry. But there is another case; when ants loop in the topology before finding the consumer, as explained in Chapter 5, the

Nonce in the packets avoids non-detected loops, but that does not mean that they are not allowed in the discovery state. When a forward ant loops, only the loop free return path is applied by the backward ant. That leaves the nodes from the other paths with unused PAT entries, which are dropped later due to an expired timeout. This happens with nodes *N02, N03* and *N04*.

The results from Figure 6.1 show how, for $n = video$, the upper path *N02-N04* is more utilized than the lower path *N03*. Although, for $n = audio$ it is the contrary, converging into the lower path. This relates to the first ant retrieved, that in the $n = video$ case takes the upper route while in the $n = audio$ case takes the lower one.

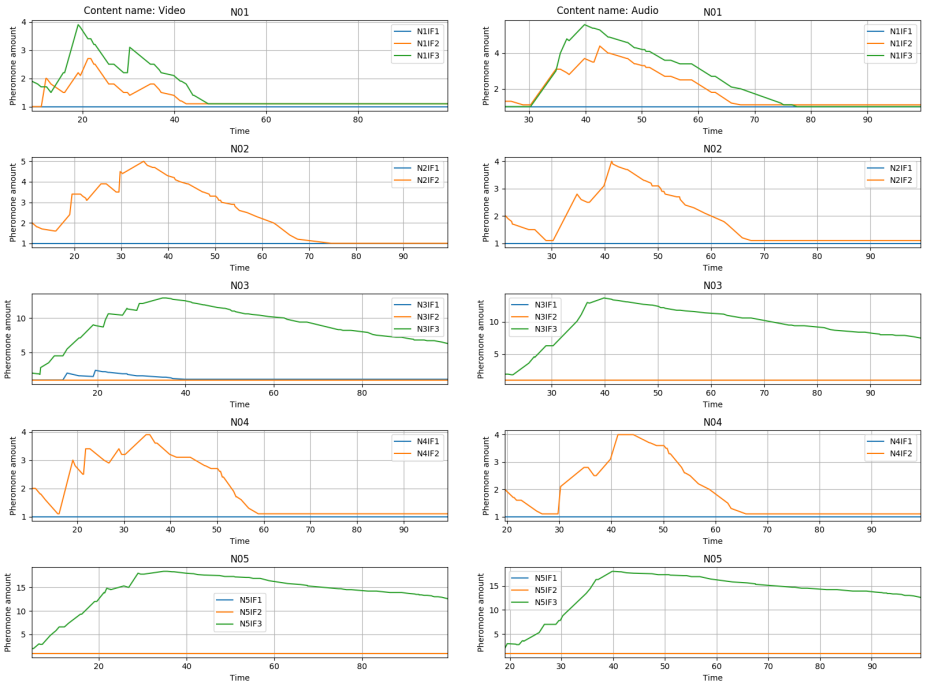
This experiment has three main objectives, the verification of correct routing function, the validation of the implementation of the NDN infrastructure including ants in the control plane and the evolution over time of this architecture. The forward and backward paths taken by the agents are evaluated using the events generated by the simulator, not shown in the previous figures. These events confirm that the backward agent takes the reverse path followed by the forward, avoiding the possible loops generated during the discovery phase. Regarding the operation of the tables, the correlation between the entries in the PAT and the pheromone levels, as explained above, shows that the components in the network behave as expected, with a double-check done by the examination of the simulation events. Therefore, the final results in this experiment align with the expected exposed in Table 5.4.

6.1.2 Experiment 2

This experiment is rather similar to the previous one in the collected data and the presentation of the results. It includes a new consumer, Figure 5.3b, which shifts the focus of the results to how the interaction between the two of them affects, and to which extent, the routing system in the network.

The pheromone evolution in Figure 6.3 shows a constant inclination in using the lower, and shortest, path *N03*. In these figures, two elements stand out; the first one for $n = audio$, the first path taken is the upper one as shown by *N04* increasing pheromones p earlier than *N03*. Nevertheless, because of the cooperation between the agents sent by the two consumers, it ends up converging to the shortest path. Secondly, in *N03* for $n = video$, the interface connecting with *N01* increases its pheromones p connoting that it is being used. This is as a result of the stochasticity of the system, allowing for alternative path discovery even when it has already converged to the shortest path.

Figure 6.4 shows how the entries e in the PAT table support the explanations above. The *N03* is highly traversed, while *N04* (the last node from the upper trail)



(a) Pheromone in the nodes for $n = video$ (b) Pheromone in the nodes for $n = audio$
Upper path. $N1.IF2 \rightarrow N2.IF2 \rightarrow N4.IF2 \rightarrow N5.IF3$
Lower path. $N1.IF3 \rightarrow N3.IF3 \rightarrow N5.IF3$

Figure 6.3: Experiment 2. Pheromone evolution

is less than half filled. The correlation between the increase of the pheromone levels p with the decrease of PAT entries e shows the expected behavior of the routing system, further supported by the simulation events.

When two consumers request the same data, the agents collaborate to find the content by the shortest path available. This experiment uses a small topology with just a couple of consumers and, therefore, the results might not be directly applicable to a bigger and more complex network. Nevertheless, the goal of this experiment is to study the behavior of the routing system when another consumer is included and validate its correct activity. And for that, the qualitative approach followed allows the examination of each event in detail and ensure that the routing is done as expected and that the anticipated results are obtained.

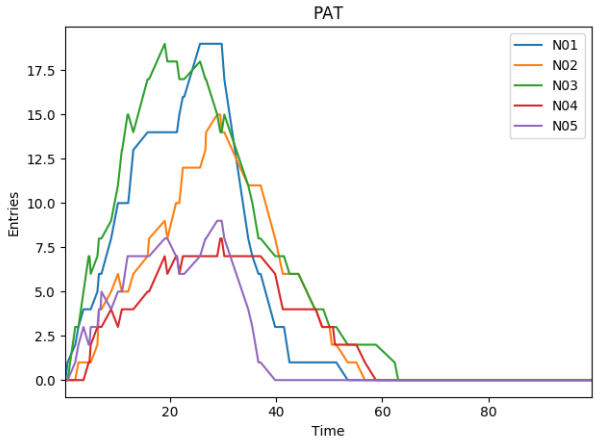


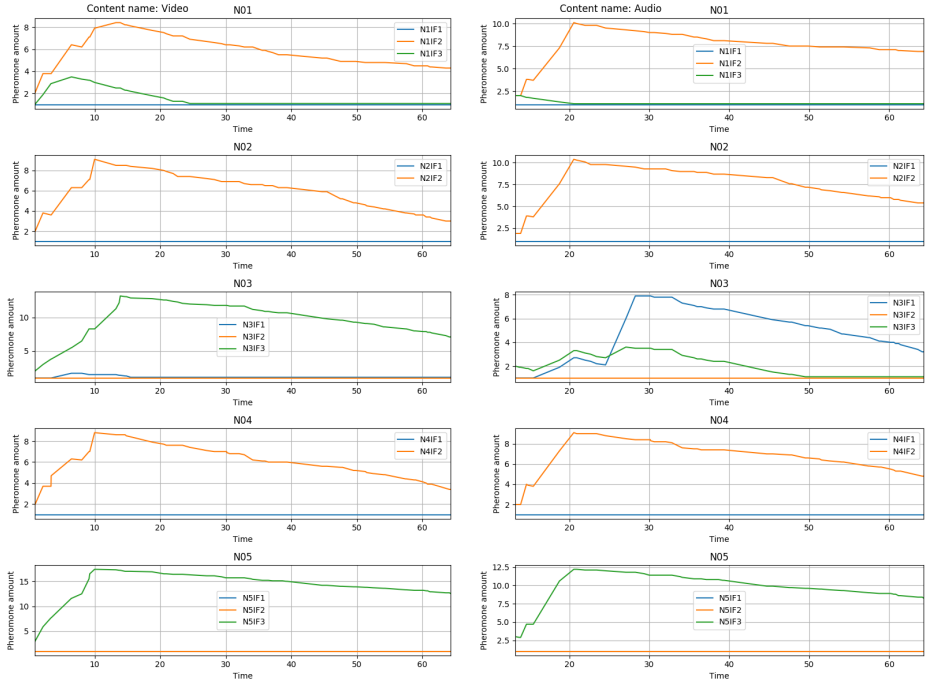
Figure 6.4: Experiment 2. PAT evolution

6.1.3 Experiment 3

In this experiment, data content is enabled, so are the forwarding state and the caching capabilities in the nodes. The center of attention is not just the routing operations, but it includes now how the forwarding is performed and the caching affects the two previous tasks. For this purpose, both consumers request the data at the same time intending beneficial cooperation.

Figure 6.5 shows how the first consumer retrieves the data through the upper path for both contents. It is displayed by the pheromone levels p in the interfaces connecting *N01-N02-N04*. When it comes to the caching role in the network, two disparate choices are made when the second consumer requests the data. When requesting the 'video' object, *N03* forwards most of the agents to *N05*, as it can be seen in Figure 6.5a through its interface *IF3*. Contrary, when requesting the 'audio' object, *N03* routes most of the agents towards *N01*, as seen in Figure 6.5b in the *N03-IF1*. This distinct behavior is caused by the cached copies in the nodes. In the first case, the ants find a copy in *N05* and so, they reinforce that path; while in the second case, they find a copy in *N01*, directing the successive agents there. Moreover, in this last case *N03* had already some amount of pheromones pointing to *N01 and N05*, and continued sending through both interfaces until the path towards *N01* was emphasized, which was then used to retrieve the cached copy.

Figure 6.6 relates to the explanation above, where the content names n are requested at different times t and, therefore, there are two main peaks in the table. Contrasting the times between the evolution of the FIBs and the evolution on the PAT entries, there is a clear course showing how the first peak corresponds to the



(a) Pheromone in the nodes for $n = video$ (b) Pheromone in the nodes for $n = audio$
Upper path. $N1.IF2 \rightarrow N2.IF2 \rightarrow N4.IF2 \rightarrow N5.IF3$
Lower path. $N1.IF3 \rightarrow N3.IF3 \rightarrow N5.IF3$

Figure 6.5: Experiment 3. Pheromone evolution

$n = video$ request while the second one corresponds to the $n = audio$ one. It also correlates with the reductions before $t = 20$ and the increase of pheromones p for the 'audio' object in the nodes *N05-N04-N02-N01* and the one at $t = 25$ and the increase of pheromones p in *N03*.

Figure 6.7 shows the content retrieval time rt, or Round Trip delay Time (RTT), along the y-axis and content names n along the x-axis . This figure shows the retrieval times for each consumer grouped by the content name. As stated by the simulation events, the first consumer retrieves the content from the producer and the second one from a cached copy in the network. This can be assessed by comparing the times shown in the figure, where consumer *C1* employs noticeably more time to retrieve the content from the producer than *C2* from a cached copy.

The results obtained from this experiment show how the implementation copes with the data objects and caching capabilities. Data objects are received by the consumers, they are stored in the nodes during the retrieving time and then, they

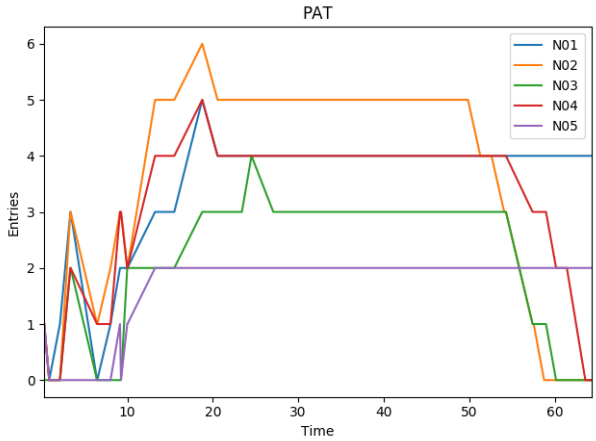


Figure 6.6: Experiment 3. PAT evolution

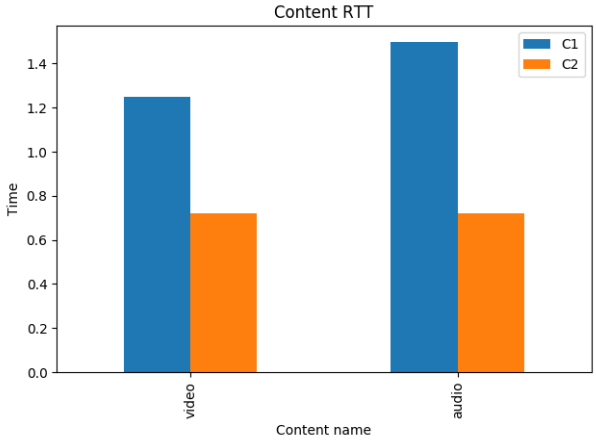


Figure 6.7: Experiment 3. Content retrieval time

are served when an Interest is received in that same node. Assessing the results from the objectives perspective, this experiment shows that the forwarding plane works as expected to direct the Datas towards the consumers through the reverse path taken by the Interest. Besides, during this same operation, the data objects inside the Datas have been stored in the nodes and further, this saved content has been correctly delivered when an Interest matching its name arrived at the node. Finally, the retrieval time rt of the content fetched from the producer and the content fetched from a copy in the network shows the importance of this feature in the NDN architecture. The final result has exceeded the anticipated outcome.

6.1.4 Experiment 4

This is the last experiment employing the first topology. Its main objective is to inspect the changes in the system when the number of chunks increases. Two elements are considered for that purpose, the evolution of the PAT entries e, showing the resource exploitation, and the content retrieval times rt, showing the effect on the transmission state (Figure 6.8).

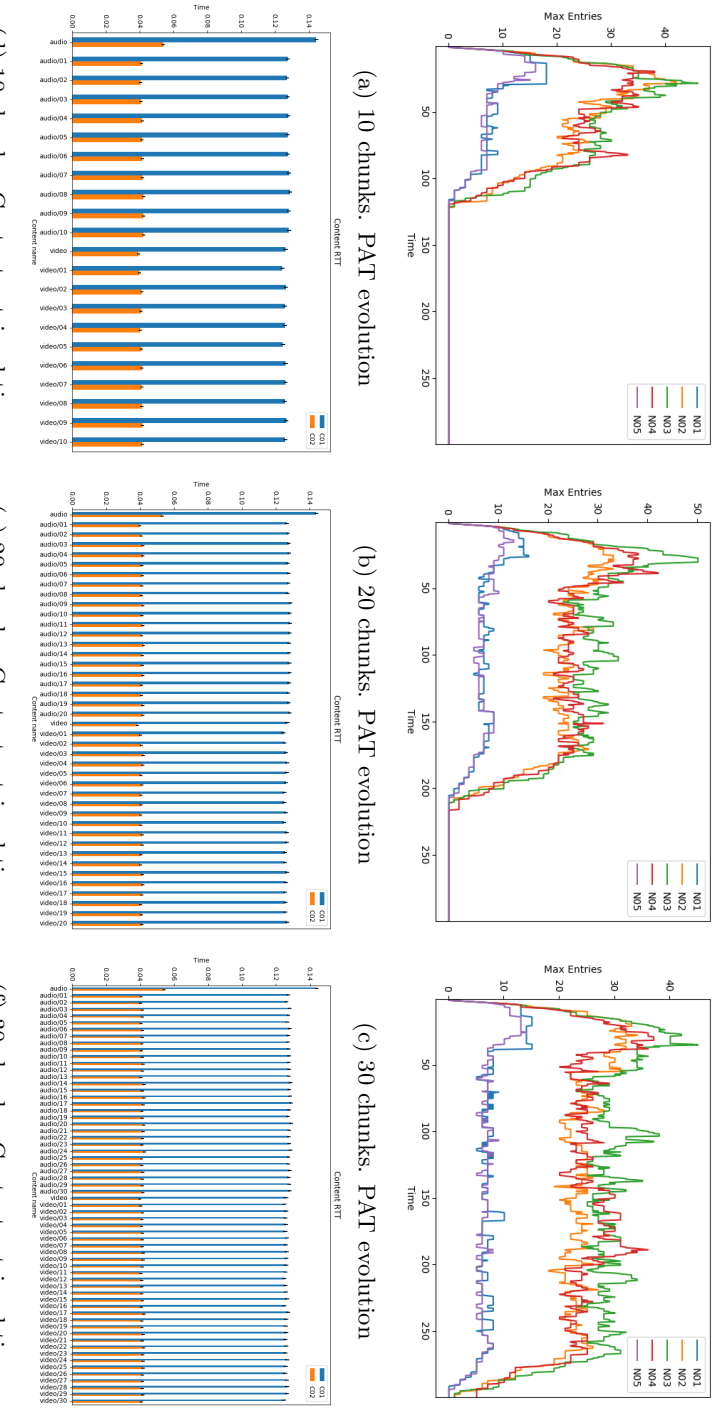
The figures in the first row show the evolution of the PAT between 10, 20 and 30 chunks per data name. The plotted values are the maximum measurements from 1000 diverse simulations using the same topology, but with different seeds. For that reason, the results presented by these figures reveal the worst-case scenario thus providing interesting information regarding the resource requirements of the system. The progression in the PAT indicates how the system copes with the chunk number variance, having some similarities among the executions.

All figures show two main peaks in the tables at similar times, one at the start $t = 0$ and one at $t = 100$. In the beginning, the system is still bootstrapping and generating some forward loops in the discovery for the general names. This is the reason behind the first, and highest, peak in all three executions. The following requests exploit the pheromones left by previous agents to avoid this success and take a more direct path. The second one is the result of the request for the chunks related to the second content, $n = audio$ in this case. Due to the process taken to request the chunks, presented in Section 5.1.2, where the consumer requests the three first chunks at the same time and then, the following up in a three-second interval. Therefore, the nodes receive a slightly higher pressure as seen in this second increment. On the other hand, the rest of the chunks do not stress out the network as it is highly stable until all requests are served.

The second row contains the illustrations of the content retrieval time average for each data name n and its confidence interval. The average of the times over simulations is taken to have an insight into what can be expected and to observe the difference between content names or content types. On the other hand, a 95% confidence interval indicates the variation and consistency that the average result has over the simulations and therefore, how reliable that measurement is.

In this case, the average retrieval time for every name n is rather similar. Moreover, the confidence interval of each one of them indicates that the majority of executions result in a retrieval time close to the average one. For each execution, the times have been comparable on all chunks which shows that the system is consistent and stable, as seen earlier over the PAT table.

Comparing the results obtained by *C1* and *C2*, where *C1* fetches the content from



(a) 10 chunks. PAT evolution

(b) 20 chunks. PAT evolution

(c) 30 chunks. PAT evolution

(d) 10 chunks. Content retrieval time

(e) 20 chunks. Content retrieval time

(f) 30 chunks. Content retrieval time

Figure 6.8: Experiment 4

the producer while *C2* does it from a cached copy stored in a node, the difference is evident. For each content name, among the three executions, the first consumer employed up to three times more time to retrieve the data object than the second one.

Overall, the increment of the chunks per content has no adverse effect neither on the PAT tables nor in the retrieval time of the content or the caching process. Moreover, the system manages fairly well these adjustments.

6.1.5 Experiment 5

This is the first experiment operating over the Uninett topology, in Figure 5.4, hence adopting a realistic size experiment. The experiments using this topology take a quantitative approach to evaluate the global behavior of the system. The reason is the number of elements composing this experiment, with 116 nodes, 357 interfaces, and several consumers and producers. Therefore, the data collected is massive and statistical methods must be applied to evaluate these results. Besides, 20 simulations are done in this experiment using 30 consumers and 3 producers randomly located in the network. The results from the execution with 200 simulations are not presented in this report due to their similarities with the 20 simulation execution. It can be found in the project repository.

Figure 6.9 presents the packet management on each simulation, in absolute numbers. The packets are divided into four categories: *hits*, *waste*, *timeout* and *interest*. The *hits* are the only received packets during the simulation, while the rest are missed or lost packets. Thus, *hits* refers to the Data objects received by the consumers. On the other hand, *waste* is the number of Data packets received by nodes without a matching entry in the PIT because a previous Data with the same name already arrived. This can happen because of a specific implementation in the simulator, which looped Interests are still forwarded, otherwise, the Interest would die there. The matter with it is that, if the incoming interface from the looped Interest is not stored, the nodes which the Interest visited before arriving at this one would never get the corresponding Data, as the interface saved in the PIT is the one from where the looped Interest arrived first from. Therefore, the incoming interface is stored resulting in the possibility to receive already served Datas in some nodes in the network. Further, *timeout* is the Data packets received by nodes without a PIT entry because the entry was removed due to the expired timeout of the table. Finally, *interest* refers to the number of content Interests that die because of their TTL.

In this same figure, the right axis considers the total number of distinct names received, or likewise, distinct data objects distributed, by the producers in the simulation. In this case, all simulations result in 22 data names served.

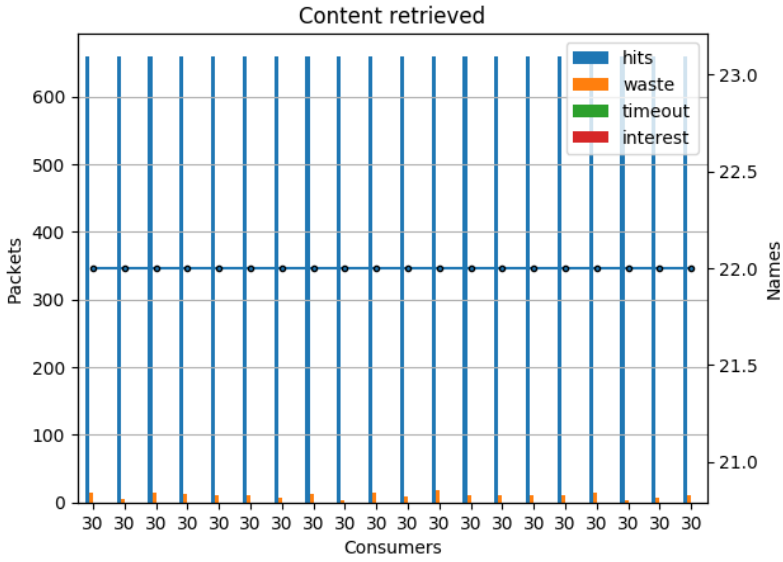


Figure 6.9: Experiment 5. Received by consumers

Figure 6.10 introduces the average content received per consumer on each simulation. It shows the number of packets *pkt* along the y-axis and the consumers per simulation *s* along the x-axis. In this case, there is not much variation, already expected from the previous illustration, as all consumers received all the requested content on every simulation.

Figure 6.11 uses the same principles as Figure 6.8d, differing in the fact that the consumers are also taken as part of the statistical calculations. This figure introduces the average times for each data name, throughout consumers and simulations jointly with a confidence interval to examine the variation over these results. In this experiment, the variation is greater as a result of the randomly allocated consumers and producers and the difference in the times when a data object is retrieved from a producer or a cached copy in the network. This last evidence is not presented in the figures in this report but can be found in the project repository.

This experiment shows the behavior of the swarm implementation over a real network topology. It can retrieve all the content it requests, with consistency over simulations. It is rather effective when it comes to waste of packets, as they are a small portion of the total retrieved, which results in a low bandwidth loss. The expected results stated in Table 5.9 are fulfilled.

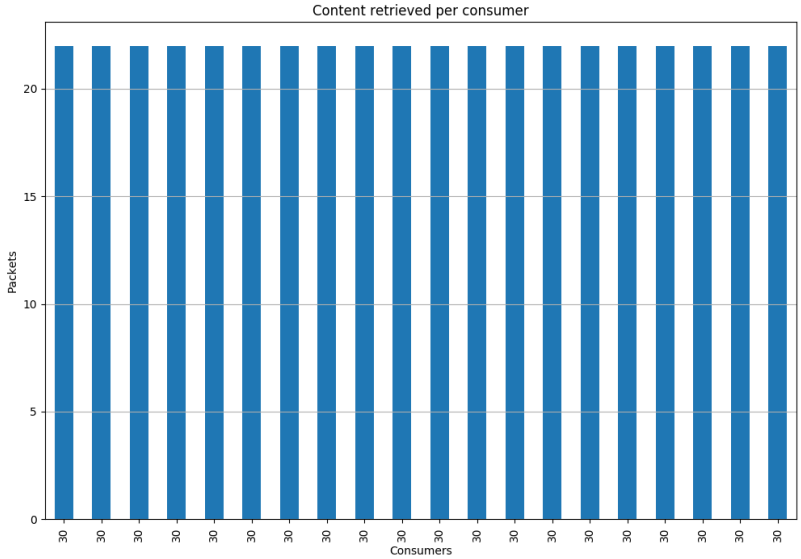


Figure 6.10: Experiment 5. Average data objects received

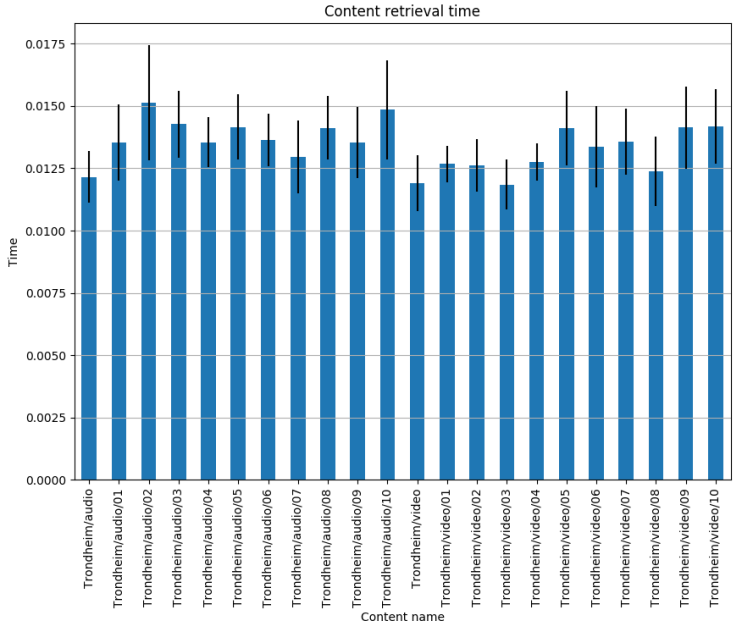


Figure 6.11: Experiment 5. Content retrieval time

6.1.6 Experiment 6

This experiment is divided into two segments; first, the results from the flooding system are presented and then, the comparative results between the two strategies applied. The results presented for the flooding system are in the same form as the ones from the ant-based method. While the comparison between techniques incorporates a new measurement to analyze them, the stretch.

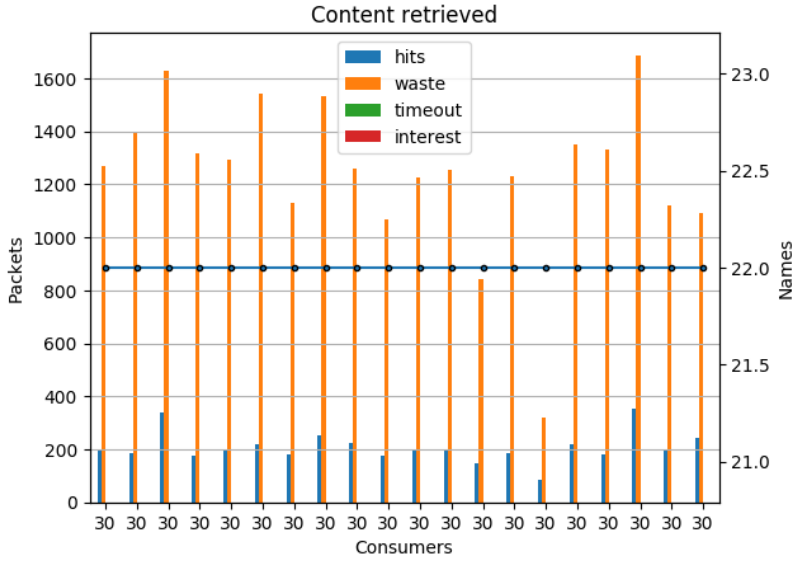


Figure 6.12: Experiment 6. Received by consumers

Figure 6.12 shows the content management in the network throughout the simulations, as explained in Section 6.1.5. The *hits* are low compared to what the requested content is, interpreted as the network not being able to manage to retrieve the demanded data objects towards the consumers. This issue is directly related to the amount of *waste* content; the more packets the consumer received, the higher the waste in the network. The reason for consumers not receiving most of the content is a cross-Interest transmission which blocks the forwarding of the Datas further in the network. To illustrate how it works, a reduced case is presented. These four nodes are connected in the following way: $X \iff A \iff B \iff C$. If A sends an Interest to X and B, and later C sends an Interest with the same name to B before B had time to resend the Interest further to C, this one is going to be aggregated in B's PIT. If afterward A gets the content from X, it is not going to forward it further cause there is no entry in the PIT for that and, therefore, B is not going to get the content, and as the Interest coming from C's side was aggregated in the same PIT entry, no content will arrive on that direction. Hence, a deadlock situation

is formed, being the cause of the major part of not returned packets in the flooding approach. Besides, the reason for this number of lost Datas is the duplicity generated by the broadcast system. As each Interest is duplicated and sent to all available interfaces, one of the paths might return the content faster due to finding a cached copy. Then, this copy is forwarded to the incoming interfaces stated in the PIT and then, the entry is dropped. If the rest of Interests return with another copy, retrieved either from the Producer or another node in the network, there is no entry in the PIT satisfying their name and, therefore, they are dropped and counted as wasted packets. Finally, note that the producers over the simulations are able to deliver all the content they possess.

Connected with the information introduced above, Figure 6.13 presents the average data objects received by each consumer. There is a high variation in the amount of content recovered, as suggested by the confidence interval in the figure. This means that the consumers in the same simulation receive a rather disparate number of data objects back. There might be a correlation between the cause behind this circumstance and the high amount of lost packets in the network. As exposed earlier, a lot of Data packets do not arrive at most of the consumers who requested them. This has even worse effects when these Data packets are the ones carrying the meta-data with the names of the chunks composing the content, resulting in the consumer lacking this knowledge and thus, it cannot request them, dropping the retrieve ratio.

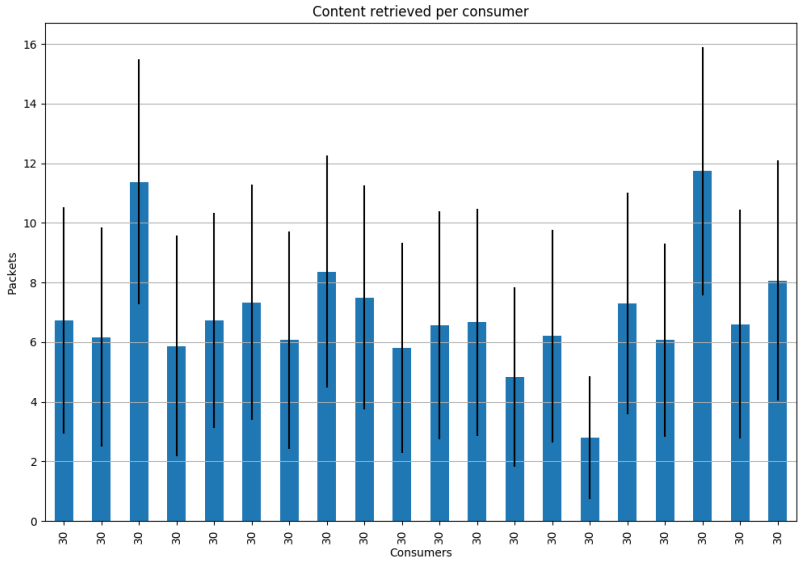


Figure 6.13: Experiment 6. Average data objects received

Figure 6.14 presents the average time to fetch every data object that has been received by a consumer. The confidence interval shows how divergent are the times between consumers. As stated in Section 6.1.5, the varied location of the consumers and producers and the big topology affect to the retrieval times for the packets. Naturally, the difference between data fetched from the producer and a cached copy is another influential factor.

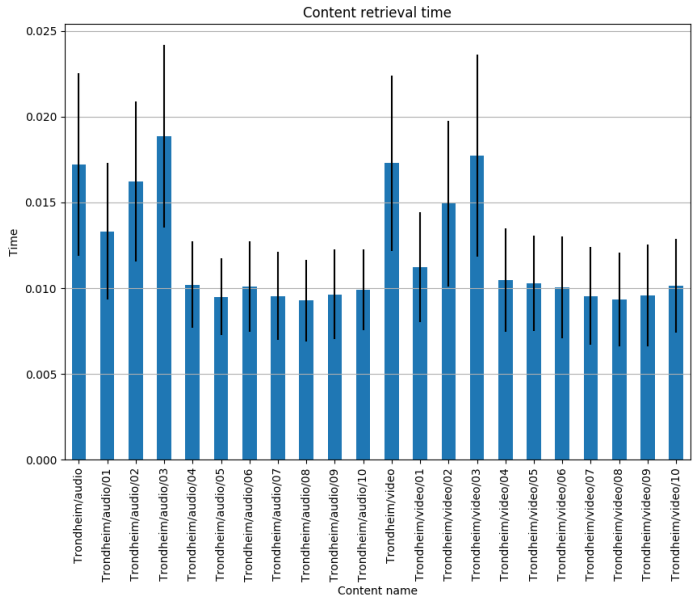


Figure 6.14: Experiment 6. Content retrieval time

The flooding implementation over the Uninett topology presents a poor performance, retrieving less than half of the data objects over the run simulations and experiencing an excessive variation on the content returned to each consumer and the retrieval times of these objects. Moreover, it saturates the network with redundant packets, ending in an unreasonable amount on lost packets. Besides, it seems to be a higher recovery time for the general name and the three first chunks of data. This concurs with the behavior of the requesting process, explained in Section 6.1.4, where the three first chunks are requested as a unit, congesting the network, and thus obtaining worse retrieval times.

The second execution for this experiment focuses on the comparison between the swarm and the flooding strategies. They are both executed with the same parameters and seed, and the results displayed below show the same ranges and degrees so it eases their observation.

Figure 6.15 is a combination of Figure 6.9 and Figure 6.12, which is divided into two illustrations. The one on top shows the data objects retrieved on each simulation, where it is established that flooding retrieves on average a third of the content swarm intelligence does. The figure below gives the wasted data in the network, being abysmally higher for the flooding technique.

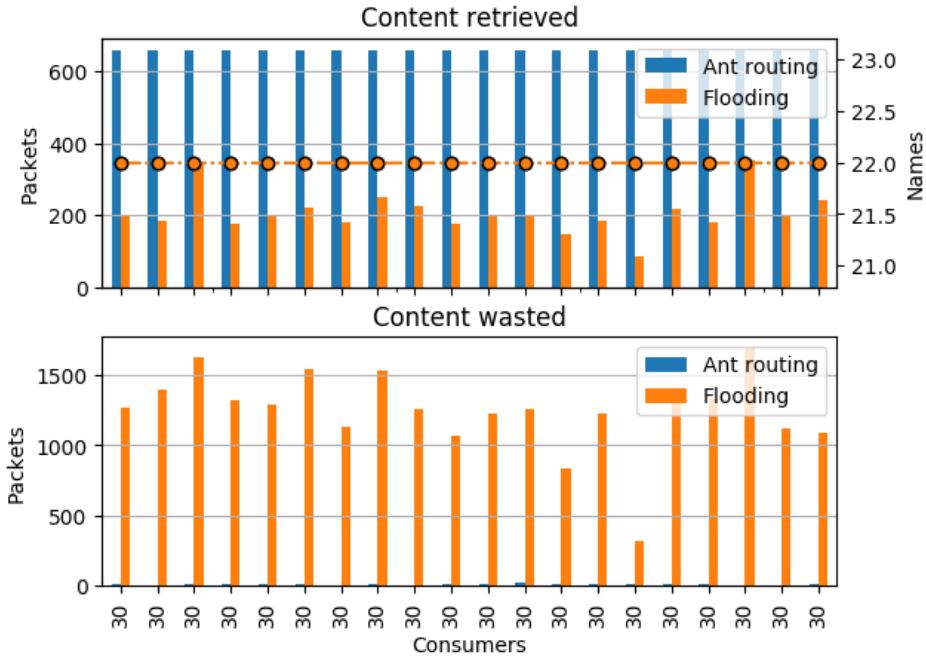


Figure 6.15: Experiment 6. Received by consumers - 20 simulations

Similar to what the previous figure shows, Figure 6.16 presents the comparison between the two techniques on the content received per consumer on each simulation and the divergence on this number between the difference consumers, determined by the confidence interval.

Figure 6.17 presents the comparison between the average stretch for each name for both strategies. Stretch can be defined as the difference between the best (or shortest) possible path through the network and the actual path the packet takes through the network. The flooding system, for its own nature, should have a stretch ratio very close to 1 indicating the use of the shortest path the majority of the times, or even lower, considering the caching capabilities of the network. On the other hand, the swarm approach looks for a good-enough path to the content but providing versatility and adaptability.

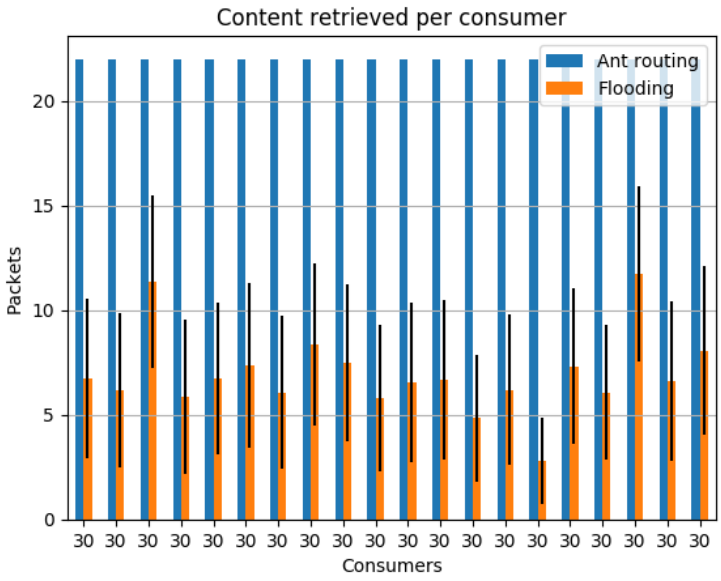


Figure 6.16: Experiment 6. Average data objects received - 20 simulations

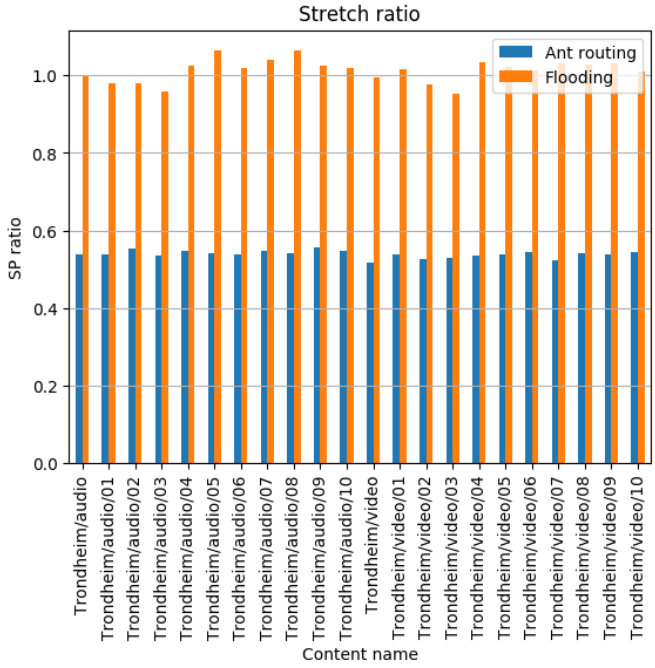


Figure 6.17: Experiment 6. Stretch

The results present how the expected behavior is confirmed by the flood-based strategy which gives a near-to-1 ratio. On the other hand, the stretch ratio offered by the ant-based technique is almost half of the one given by the flooding. The main reason behind this situation is the capacity of the swarm routing to find and retrieve the closest copy from the in-network storage taking advantage of caching.

Following the same principle than in the previous figure, Figure 6.18 shows the stretch time. It is the retrieval time divided by the number of hops of the shortest possible path to a Producer. This result tries to provide a measure to compare the performance of the two strategies over the time plane. The ant-based approach has rather stable values for all names, similarly happening with the flood-based strategy except for the general name and the three first chunks, which have fairly higher stretch time than the rest. This phenomenon relates to Figure 6.14 and Figure 6.17, where they present how for these names the retrieval time is higher but the stretch ratio is comparable with the rest. This indicates that the reason these object names take more time to be fetched is the congestion in the network and not the length path followed.

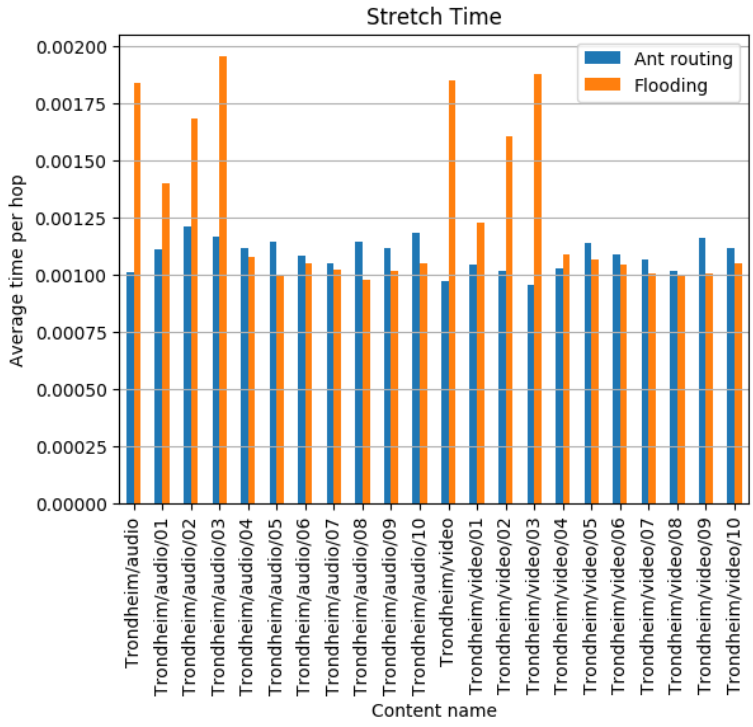


Figure 6.18: Experiment 6. Stretch time

The results obtained by the execution with 200 simulations, as stated in Table 5.10, is completely aligned with the results shown by the execution with 20 simulations. Therefore, and because of their size, they are placed into Appendix A.

The results obtained in this last experiment show an effective routing system, not only is confronted with the flooding system but with general measurements as the stretch. The predicted results based on the features of the systems were to have the flooding system obtaining a clear better outcome on the stretch ratio, although a higher retrievable time due to the congestion of the network. In the end, the swarm approach employed more efficiently the in-network storage to reduce drastically the stretch ratio to almost half of the shortest path length. Moreover, the times between strategies on average per hop are rather close when the network is not collapsed by the flooding technique as with the first four names requested. Finally, the low retrieval of packets by the flooding system was unexpected. While a high loss of content was expected due to the nature of the mechanism, the clashing between Interests was not predicted and enlightening it made possible to understand this issue which was not foreseen.

6.2 Discussion

This section discusses the proposed research questions and how they relate to the previously presented experiments, an evaluation of the methodology followed to achieve these results and, finally, a reflection on the limitations of this project.

6.2.1 The integration of SI in ICN

The assumption of Information Centric Networks being able to adapt a Swarm Intelligence solution was prior to this work. The adaptability shown by swarm algorithms over many different architectures is a clear support to that statement. The first RQ is: "*Can Swarm Intelligence become an alternative to the current routing schemes in an Information Centric Network architecture?*" The final objective for this RQ is to implement a solution to prove whether it is possible to use SI over an ICN class, and how it performs in the routing and forwarding planes when the characteristics of NDN, the ICN scheme chosen, are included.

The starting point to make a design is to understand the basic aspects of the components being used and to have a profound knowledge of the factors influencing them. Thus, the literature review in Chapter 3 helps to comprehend the elements composing an NDN and SI system. This research reinforced the decision to use NDN and ant-based techniques as a foundation for the design because of the similarities shown by both concepts. The comparable behavior between Interest and Data with forward and backward agents can be seen, for example, in how the location-independent naming relates to the nature of swarm systems, like ant colonies, to find content by roaming. This conduct links also to the in-network storage capabilities of the nodes providing high mobility of content, and therefore not having a permanent position.

The design in Section 5.1 offers a proposal for a solution integrating both systems using the resemblances in their components. To validate this proposal, the implementation of the design is performed on a network simulator, as explained in Section 5, including all the main aspects exposed in the scheme along with a set of experiments. These experiments help verify each of the tasks and purposes the system has.

The evaluation of the first experiment provides proof of the correct operation of the components within a simple routing scenario. This experiment also offers a verification on the simulator implemented, by means of the operation sequence produced, which is the same as the predicted one. When comparing the two first experiments, the incorporation of another consumer offers a better solution for the forwarding plane as the shortest path is reinforced, the agents from both consumers exploit the same pheromone type and take advantage of this to converge in the best possible path. Moreover, converging to that path is faster, as it can be seen in both

PAT figures. Besides, it corroborates the hypothesis that forward and backward ants can be assimilated as Interest and Data packets.

The forwarding plane has been studied once the experiments incorporated data objects in the executions. The retrieved content from the producer first, and the node store later, confirms the appropriate development of the experiment. The variation in the direction where the pheromones point between the first object retrieved and the second one, from a cached copy, endorse the hypothesis that the flexibility and variability of swarm routing could help with the cached copy discovery.

The behavior when including chunks of data for each content demonstrated some assumptions made in the design. First, it applies the naming structure of NDN with two levels, the general name and the chunk number. This structure is used by the system to provide the needed data to the consumer. At first, when the consumer knows the general name and sends an Interest with it, the producer employs the longest name prefix match technique to determine the data object to be returned. Then, the consumer can continue requesting the actual content. Besides, the network exploits this fact to introduce name aggregation on the node tables, letting the routing state take advantage of it and use previous pheromones, partially matching the requested name, to help discover a path to the content.

The results obtained by these experiments validate the expected behavior of each component of the design, as it has been the objective of the first scenario. The qualitative approach taken to verify the correctness of the operations is satisfactory as every potential necessary event has been studied and the trace followed by the agents, both forward and backward, has been certified. Moreover, the use of Content Stores in the nodes demonstrates how the flexibility of swarm techniques can be effective over NDN by discovering content diverging from the convergent path. Besides, the system fulfills all the tasks when multiple agents carry distinct names, ergo laying different pheromones. Finally, the employment of a general name with several chunk names serves to test the hierarchical name scheme adopted in NDN.

6.2.2 Scalability and routing performance

The other main research question in this thesis is about the routing efficiency and the scalability concerns that ICN generates, reinforced by the operation of swarm systems. It is stated as: "*How does SI over an ICN architecture deal with routing efficiency and scalability?*". As exposed in the previous section, the routing system operates correctly and as expected and the content is retrieved how it is supposed. Nevertheless, this does not mean that the design is able to scale into another topology level with further operational stress. Thus, it must be evaluated how the stochastic approach to path discovery employed by the swarm method, with several agents that

update information in the network nodes, performs and how it compares to flooding, which broadcasts the requested Interest to all available interfaces.

When faced with a real topology and user behavior, the ant-based approach has obtained quite good results. As presented in Section 6.1.5, the system is able to retrieve all content as supposed and with small variations on the retrieval times. The results validate a consistent activity in the system, achieving the desired outcome.

On the other hand, the flooding implementation presents a rather bad development. The amount of received content is not close to half of the requested and, on top of that, the number of wasted packets in the network is several times the retrieved. Although it was not predicted, it was not completely unexpected as in Shi et al.'s work they also compare their design to a basic broadcast solution which produces a poor quality outcome. Nevertheless, the flooding approach is used for its similarities with SI, as stated in Chapter 4. Therefore, some of the reasons why both strategies get such contrasting results are next exposed, focusing on their routing performance and scalability effects.

As presented in Section 6.1.6, the average stretch is a good indicator of the routing performance. In this case, both strategies demonstrate satisfactory routing efficiency as flooding employs paths with a similar length than the shortest possible one, while swarm does it with almost half the length of the best option. Both outcomes are positive results indicating good performance with the received packets, the contrast comes when the number of retrieved objects is in the equation, leaving the flooding system with a rather lower performance. Note that the swarm technique gets these results as a result of exploiting the cached copies, indicating potential as it lowers the load in the network by using the closest possible content copy.

One difference between the two systems is the use of domains to route and forward packets. In the ant-based approach, the nodes implement the domain sharing technique at startup, filling the network with high-level routing information. This is used by the agents during the discovery phase, which helps them direct to the area where the content producer is located. The verification of this process not only benefits the routing process by optimizing the agents sent, but it also has scale implications as it demonstrates the role of domains in the topology and how the system takes advantage of them to improve routing.

Furthermore, the node procedure to aggregate content names is directly related to the previous concept. The domain in the names is not only useful at start-up time but whenever the content is asked for the first time. In the experiment presented in Section 6.1.6, two content names are requested, 'video' and 'audio', with 10 chunks of data composing each of them. The hierarchical name scheme is used in this case every time one of these data objects is wanted, as there is a partial match in the

nodes, of different degrees, which directs the request to the most probable closest copy. Contrasting, the flood system broadcasts the request over the whole network utilizing higher network resources.

Throughout this project, and especially during the experimentation phase, the concerns about an SI approach have been analyzed. This validation provides a positive conclusion to how the swarm technique behaves when the conditions are scaled to a real scenario. All procedures implemented have been verified and have provided with promising results.

6.2.3 Evaluation/Reflection of the methodology followed

The methodology followed in this thesis starts with a collection of knowledge to have enough background to be able to draw the first design. When a design covering all the basic functions of NDN is sketched, its implementation begins. Several iterations of design and implementation take place, where the complexity is increased on each one. This leads to a system able to obtain the results to be used for the evaluation of the Research Questions and to verify whether the design fulfills them or not. Finally, explaining and discussing the results is done and a conclusion is achieved.

Using several iterations to come up with a design has been very useful. On each iteration, new knowledge was acquired and applied to the next one. In the beginning, with the basic routing functions and a small topology, it was possible to follow every single packet flow and understand the behavior of the system and come up with the expected result. From there, each iteration increased the roles of each component, having the security that the previous functions worked as predicted. Moreover, to start with the simpler tasks allowed to detect early failures and understand them, leaving the system error-free for the next iteration and thus, simplifying the debugging process when necessary.

6.2.4 Limitations

There are several decisions taken in this project that can influence the outcome of the finals results. Most of these choices relate to the design and its implementation, and therefore, they might have an impact on the experiments. Some of these limitations are exposed in this section.

One of the first decisions taken in this thesis has been the use of a self-made simulator to implement and validate the proposed design. Although this choice provides a higher degree of flexibility and adaptability to the system created, it also has some drawbacks. The use of an emulator, or a test-bed system, provides closer results to reality as they introduce the same parameters as it would use a real-life implementation of the system obtaining a more precise outcome. Nevertheless, as

stated in Chapter 4, the steep learning curve of these tools contrasts with the time constraints of this project and therefore, directed the decision to a less complex solution.

Related to the previous point, this project's simulations use a best-case scenario where there are no failures. Therefore, there is no node, link or other network components that can compromise the results with an unplanned behavior. This decision reduces the realism of the simulation, but it allows to study and validate each process as the expected outcome is known.

In the last scenario, introducing a real topology that provided an approach to study scalability, two names (or twenty with the chunks) are used. With this experiment, it is possible to evaluate the behavior of the system and its routing capabilities when scaling up the topology. Nevertheless, the number of content in a real case, with a topology of that size, would be enormously higher. Therefore, the simulation of a scenario where thousands of content names are used could give a better insight into the overall system performance.

Chapter 7
Conclusion

This thesis has focused on providing an alternative routing and forwarding strategy for an ICN variant. Named Data Networking and ant-based techniques have been chosen as ICN and SI variants, respectively. Previous research suggests the versatility of swarm methods, thus providing an encouraging alternative to integrate with NDN. Both concepts have been researched as foundations for the further design of the system.

The design proposed and implemented in this project tries to provide a new approach to routing techniques over NDN. The experiments executed demonstrate the correct behavior on the routing and forwarding planes when using SI. The particularities in the NDN architecture are adopted and integrated into the solution and employed by the swarm agents to provide an efficient system. Moreover, the results confirm that the system benefits from the flexibility of the ants as they are able to retrieve most of the content from the closest copy, even when this one is in a divergent path from the producer.

Furthermore, the results establish that the system can scale. The performance displayed in the experiments with a larger topology endorses the utilization of this design. Not only able to retrieve all the content requested, but it is able to reduce the stretch to almost half by finding the closest cached copy. Thus, certifying a good routing performance.

The contribution done by this thesis is to provide a new concept to the NDN routing field. It might not be an alternative yet, as it requires further work and better optimization, but it has promising results and the integration between the two concepts confirms the potential of this approach. It also establishes a particular opportunity for NDN researchers to widen their alternatives, giving an operational stochastic concept which adapts to the NDN architecture.

Some recommendations for future research are presented next. First, flooding

systems are considered a very basic approach and NLSR is currently considered a much better alternative [15], hence, this design should be better compared to NLSR in the future. Another step should be to move from a simulation-based execution to a real-world implementation, or using a test-bed, to obtain real measurements. Moreover, further experiments with a dynamic topology and a larger number of users and content, with network component failures, could provide more realistic results. Finally, future work should cover security aspects in the design as implementing signature treatment to validate the consumers and producers or propose a mitigation solution for Interest flooding attacks.

To conclude, this thesis answered all the research questions stated at the beginning of the project. It also provided an important insight of NDN and SI, as well as the approaches on how to design and implement a solution. Furthermore, the outcome of this work can be used for other researchers as a foundation to develop new procedures and to demonstrate that different and peculiar approaches can be taken from other research areas to solve novel issues.

References

[1] Bengt Ahlgren, Christian Dannewitz, Claudio Imbrenda, Dirk Kutscher, and Borje Ohlman. A survey of information-centric networking. *IEEE Communications Magazine*, 50(7), 2012.

[2] Md Faizul Bari, Shihabur Rahman Chowdhury, Reaz Ahmed, Raouf Boutaba, and Bertrand Mathieu. A survey of naming and routing in information-centric networks. *IEEE Communications Magazine*, 50(12), 2012.

[3] Eric Bonabeau, Directeur de Recherches Du Fnrs Marco, Marco Dorigo, Guy Théraulaz, Guy Theraulaz, et al. *Swarm intelligence: from natural to artificial systems*. Oxford university press, 1999.

[4] VNI Cisco. Cisco visual networking index: Forecast and methodology 2016–2021.(2017), 2017.

[5] Máté J Csorba, Hein Meling, Poul E Heegaard, and Peter Herrmann. Foraging for better deployment of replicated service components. In *IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 87–101. Springer, 2009.

[6] Christian Dannewitz, Dirk Kutscher, BöRje Ohlman, Stephen Farrell, Bengt Ahlgren, and Holger Karl. Network of information (netinf)–an information-centric networking architecture. *Computer Communications*, 36(7):721–735, 2013.

[7] Gianni Di Caro and Marco Dorigo. Antnet: Distributed stigmergetic control for communications networks. *Journal of Artificial Intelligence Research*, 9:317–365, 1998.

[8] Marco Dorigo and Gianni Di Caro. Ant colony optimization: a new meta-heuristic. In *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*, volume 2, pages 1470–1477. IEEE, 1999.

[9] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1):29–41, 1996.

[10] Frederick Ducatelle, Gianni A Di Caro, and Luca M Gambardella. Principles and applications of swarm intelligence for adaptive routing in telecommunications networks. *Swarm Intelligence*, 4(3):173–198, 2010.

[11] Nikos Fotiou, Pekka Nikander, Dirk Trossen, and George C Polyzos. Developing information networking further: From psirp to pursuit. In *International Conference on Broadband Communications, Networks and Systems*, pages 1–13. Springer, 2010.

[12] POUL E Heegaard, BJARNE E Helvik, and OTTO J Wittner. The cross entropy ant system for network path management. *Telektronikk*, 104(01):19–40, 2008.

[13] Martin Heusse, Dominique Snyers, Sylvain Guerin, and Pascale Kuntz. Adaptive agent-driven routing and load balancing in communication networks. *Advances in complex systems*, 1(02n03):237–254, 1998.

[14] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. Networking named content. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pages 1–12. ACM, 2009.

[15] Ouassim Karrakchou, Nancy Samaan, and Ahmed Karmouch. A survey of routing mechanisms in icns. In *2018 6th International Conference on Multimedia Computing and Systems (ICMCS)*, pages 1–6. IEEE, 2018.

[16] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. A data-oriented (and beyond) network architecture. In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 181–192. ACM, 2007.

[17] Jianli Pan, Subharthi Paul, and Raj Jain. A survey of the research on future internet architectures. *IEEE Communications Magazine*, 49(7), 2011.

[18] Laurent Paquereau and Bjarne E Helvik. Revisiting the auto-regressive functions of the cross-entropy ant system. In *International Workshop on Self-Organizing Systems*, pages 137–148. Springer, 2009.

[19] Laurent Paquereau and Bjarne E Helvik. Opportunistic ant-based path management for wireless mesh networks. In *International Conference on Swarm Intelligence*, pages 480–487. Springer, 2010.

[20] Laurent Paquereau and Bjarne E Helvik. Ant-based multipath routing for wireless mesh networks. In *European Conference on the Applications of Evolutionary Computation*, pages 31–40. Springer, 2011.

[21] Muhammad Saleem, Gianni A Di Caro, and Muddassar Farooq. Swarm intelligence based routing protocol for wireless sensor networks: Survey and future directions. *Information Sciences*, 181(20):4597–4624, 2011.

[22] Ruud Schoonderwoerd, Owen E Holland, Janet L Bruten, and Leon JM Rothkrantz. Ant-based load balancing in telecommunications networks. *Adaptive behavior*, 5(2):169–207, 1997.

[23] Junxiao Shi, Eric Newberry, and Beichuan Zhang. On broadcast-based self-learning in named data networking. In *2017 IFIP Networking Conference (IFIP Networking) and Workshops*, pages 1–9. IEEE, 2017.

[24] Lan Wang, Vince Lehman, AKM Mahmudul Hoque, Beichuan Zhang, Yingdi Yu, and Lixia Zhang. A secure link state routing protocol for ndn. *IEEE Access*, 6: 10470–10482, 2018.

[25] Horst F Wedde, Muddassar Farooq, and Yue Zhang. Beehive: An efficient fault-tolerant routing algorithm inspired by honey bee behavior. In *International Workshop on Ant Colony Optimization and Swarm Intelligence*, pages 83–94. Springer, 2004.

[26] Horst F Wedde, Muddassar Farooq, Thorsten Pannenbaecker, Bjoern Vogel, Christian Mueller, Johannes Meth, and Rene Jeruschkat. Beeadhoc: an energy efficient routing algorithm for mobile ad hoc networks inspired by bee behavior. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 153–160. ACM, 2005.

[27] Otto Wittner, Poul E Heegaard, and Bjarne E Helvik. Scalable distributed discovery of resource paths in telecommunication networks using cooperative ant-like agents. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, volume 2, pages 1456–1465. IEEE, 2003.

[28] Cheng Yi, Jerald Abraham, Alexander Afanasyev, Lan Wang, Beichuan Zhang, and Lixia Zhang. On the role of routing in named data networking. In *Proceedings of the 1st ACM Conference on Information-Centric Networking*, pages 27–36. ACM, 2014.

[29] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, Patrick Crowley, Christos Papadopoulos, Lan Wang, Beichuan Zhang, et al. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014.

Appendix A

Extra results

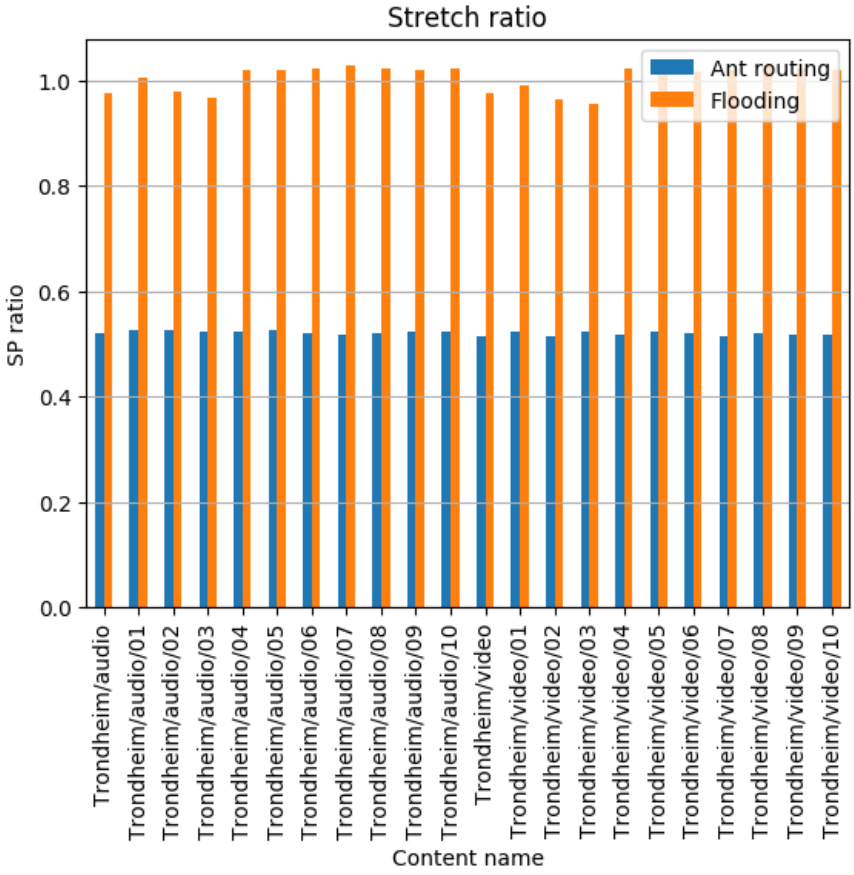


Figure A.1: Experiment 6. Stretch ratio compared to SP - 200 simulations

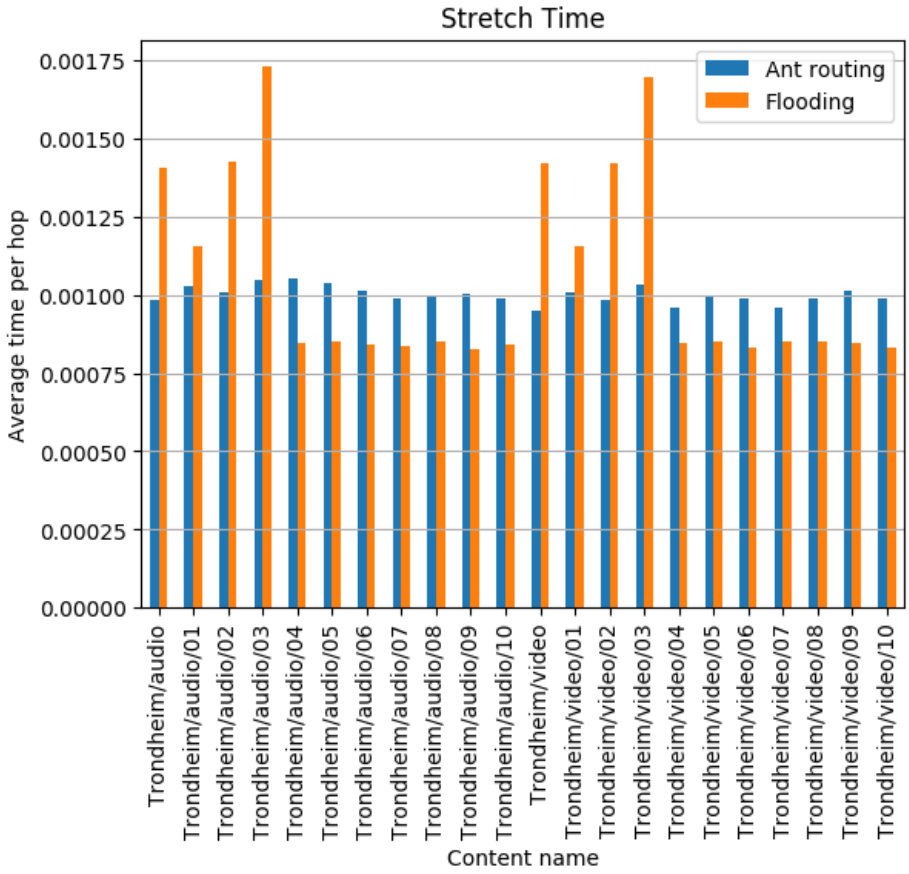


Figure A.2: Experiment 6. Stretch time per hop of SP - 200 simulations

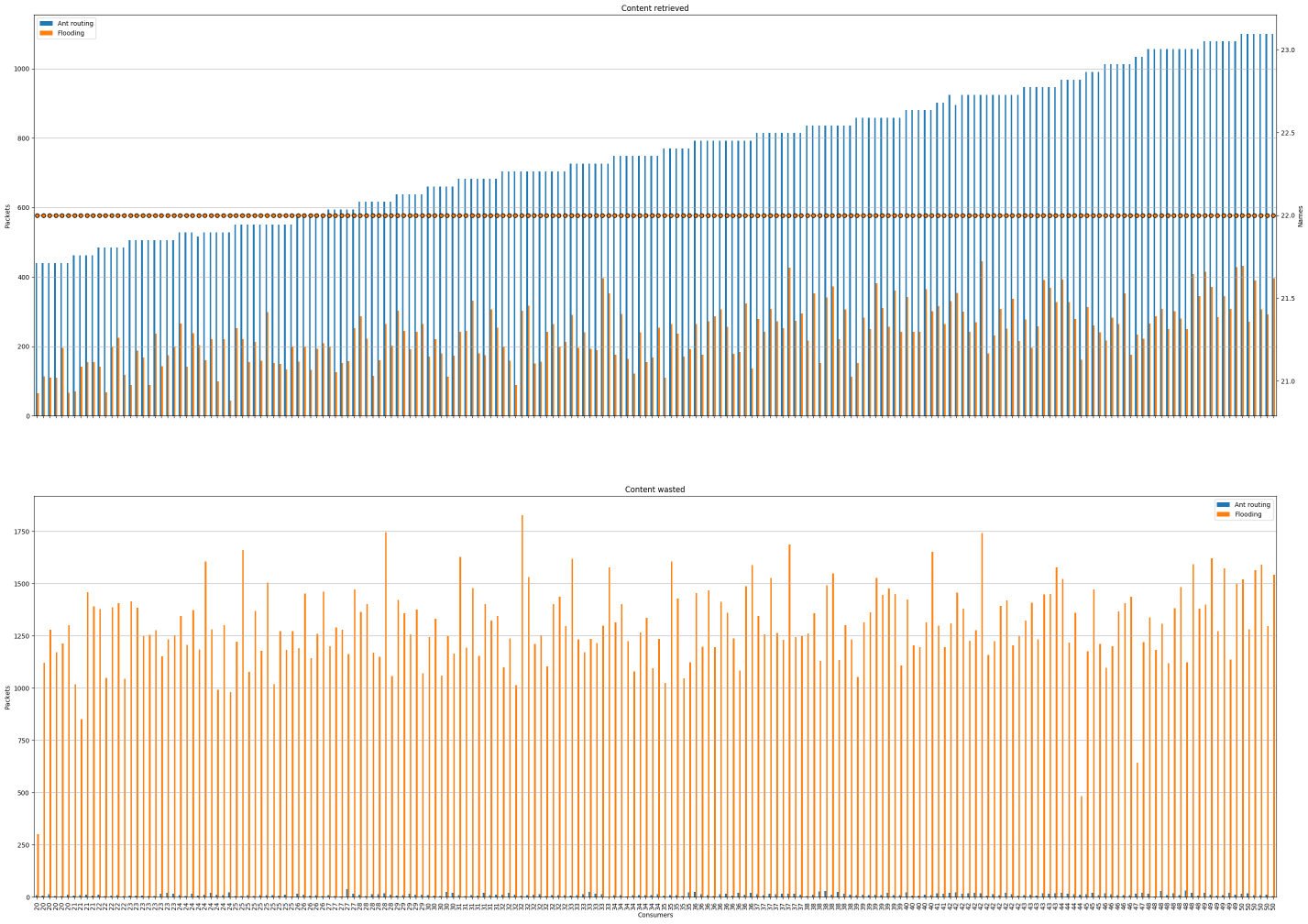


Figure A.3: Experiment 6. Received by consumers - 200 simulations

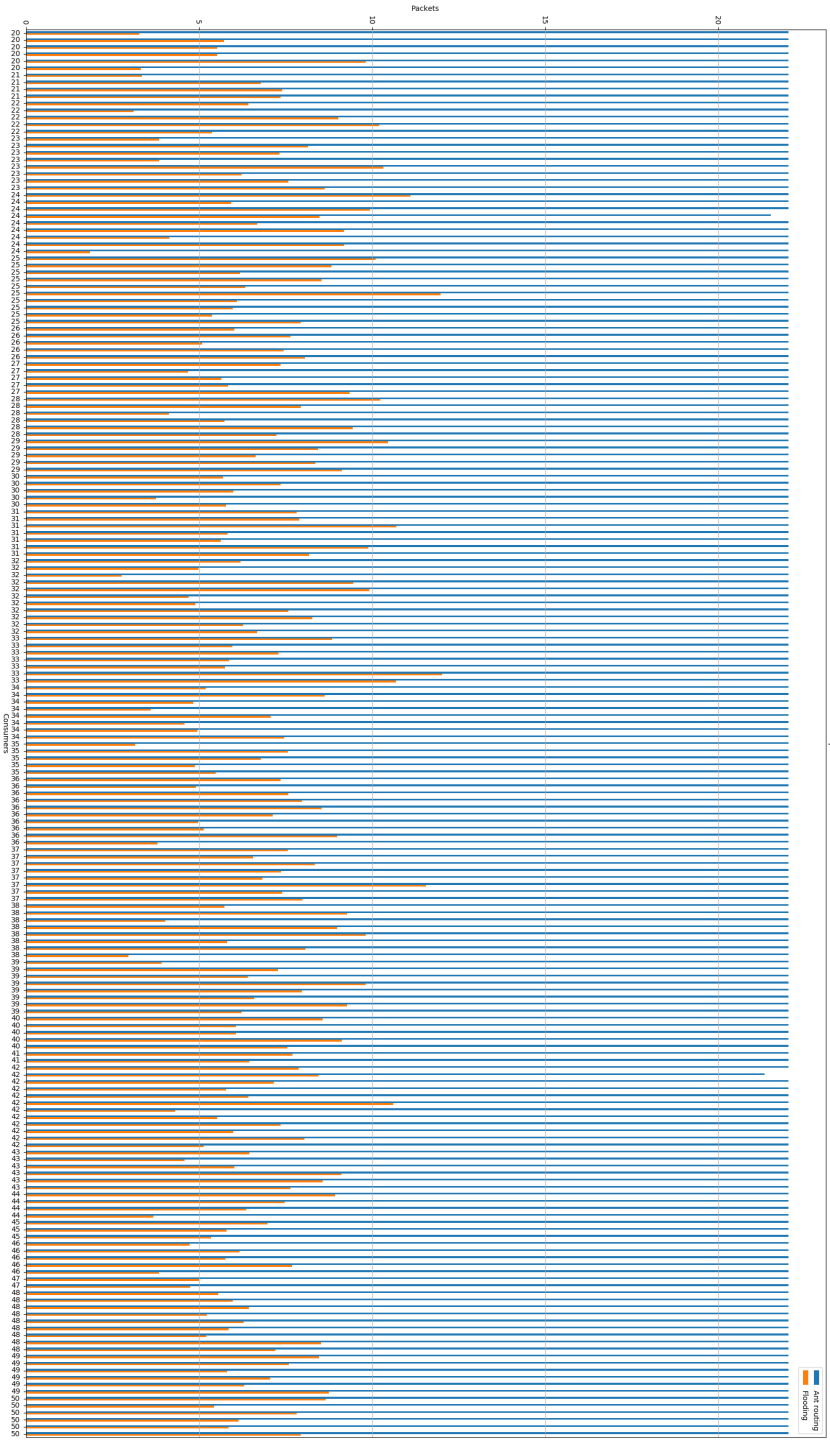


Figure A.4: Experiment 6. Average data objects received - 200 simulations