Universitat
Autònoma
de Barcelona

etse

PRELIMINARY STUDY OF COOPERATION IN HYBRID AD-HOC NETWORKS

Memòria del Treball Final de Carrera d'Enginyeria Tècnica de Telecomunicació, especialitat Sistemes Electrònics

realitzat per

Federico Fernández Delgado

i dirigit per

María Ángeles Vázquez Castro

Bellaterra, 9 de Juny de 2007

Universitat
Autònoma
de Barcelona

etse
Escola Tècnica Superior d’Enginyeria

El sotasignat, Maria Ángeles Vázquez Castro,

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en Federico Fernández Delgado,

I per tal que consti firma la present.

Signat: ....

Bellaterra, ...de...............................de 200.....

ABSTRACT

In this paper, we present a first approach to evolve a cooperative behavior in ad hoc networks.

Since wireless nodes are energy constrained, it may not be in the best interest of a node to always accept relay requests. On the other hand, if all nodes decide not to expend energy in relaying, then network throughput will drop dramatically. Both these extreme scenarios are unfavorable to the interests of a user. In this paper we deal with the issue of user cooperation in ad hoc networks by developing the algorithm called Generous TIT-FOR-TAT (GTFT).

We assume that nodes are rational, i.e., their actions are strictly determined by self-interest, and that each node is associated with a minimum lifetime constraint. Given these lifetime constraints and the assumption of rational behavior, we study the added behavior of the network.

TABLE OF CONTENTS

TABLE OF FIGURES

1 INTRODUCTION

1.1 OVERVIEW

DVB-RCS is an open standard that provides a return channel via satellite to systems based on DVB standard. Due to its implementation on Ad-Hoc Networks (AHNs), we develop a simple model, based on game theory, analyzing characteristics of AHNs to improve the integrated system.

Since, Ad hoc networks are formed by a collection of wireless nodes without the support of any existing infrastructure. Nodes in an ad hoc network may serve as hosts (end points of communication) or as routers forwarding packets to other hosts. So, we want to optimize the system by developing a protocol that links the satellite with a lost terminal in case of bad weather, infrastructure issues or any condition that reduces the communication capacity between them.

We can reach that goal by using "game theory" and the advance in wireless technologies. Analyzing how to improve the protocol what is handling the different nodes within the network.

In this paper we will assume a scenario with four nodes. We try to reach the best protocol to increase the global throughput in the net making sure that signal can arrive to any terminal from side to side of the AHN.

Paper Supervisor: Maria Angeles Vázquez Castro
Title: Department of Telecommunications and Systems engineering. Signal and Communications Theory.

Paper Committee Member: Josep Parrón Granados
Title: Department of Telecommunications and Systems engineering. Signal and Communications Theory.

Paper Committee Member: Pedro Antonio de Paco Sánchez
Title: Department of Telecommunications and Systems engineering. Signal and Communications Theory.

1.2 MOTIVATION AND OBJECTIVES

The current trend is to achieve information wherever we are by using the wireless technology. DVB-RCS let us get this information but it has its boundaries that are related with satellite signal problems like bad weather or infrastructure problems.

Nowadays, there are a lot of papers and studies about the AHNs traffic balancing but they determine different priorities of the resources. Normally, the energy efficiency is the most common fact to be treated. We want to do a preliminary study of this efficiency in a network that will create external traffic to a satellite.

So we have to study cases focused on the optimization of the global throughput network where the nodes behavior cares. This demands a complete study of the various links possible and development of a mechanism for establishing those links to improve the system.

Ad hoc networks are an emerging networking technology, in which the terminals form a network without any fixed infrastructure. The operation of the network is based on cooperation thanks to the grade of generosity. Each node forwards traffic of the others.

Game theory deals with multiperson decision making, in which each decision maker tries to maximize his utility. Game theory originates from economics, but it has been applied in various fields. In this paper, we introduce the basic concepts of game theory and its applications in telecommunications. The cooperation of the users is crucial to the operation of ad hoc networks, hence game theory provides a good basis to analyze the networks.

We analyze the relationship between a node and the rest of the network from the energy efficiency perspective using game theory. We simulate networks in order to study the characteristics of the nodes that lose energy when rely traffic from the neighbor.

1.3 STRUCTURE OF THE PAPER

CHAPTER 2→ It's a brief introduction about DVB-RCS standard.

CHAPTER 3→ Introduces the Ad-Hoc Networks.

CHAPTER 4→ Introduces the cooperation in wireless Ad-Hoc networks and the Generous Tit-For-Tat algorithm.

CHAPTER 5→ Explanation of the simulation results from four different cases. Study of the grade of generosity in the algorithm, the power constraint and consumption dependence and the behavior for a external traffic throughput.

CHAPTER 6→ Observations about the simulation results.

2 DVB-RCS

2.1 WHAT IS DVB-RCS? [1]

DVB-RCS stands for Digital Video Broadcast - Return Channel Satellite. DVB-RCS is part of the DVB standards for satellite communication, DVB-S and DVB-S2. The purpose of DVB-RCS is to provide a return channel to enable Internet and other data services over satellite.

The DVB standards are maintained by the DVB Project, which is an industry-led consortium of over 260 broadcasters, manufacturers, network operators, software developers, regulatory bodies and others in over 35 countries.

It is officially defined in ETSI EN 301 790: Digital Video Broadcasting (DVB); Interaction channel for satellite distribution systems. The DVB-RCS standard is explained in ETSI EN 101 790: Digital Video Broadcasting (DVB); Interaction channel for Satellite Distribution Systems; Guidelines for the use of EN 301 790.

DVB-RCS is the open standard for bi-directional or two-way transmission of digital data. It employs satellite transmission using combinations of C, Ku and Ka bands with return bandwidth up to 2 Mbit/s. It takes full advantage of the benefits of satellite and is instrumental in efforts to bridge the digital divide.

Interoperability is one of the main advantages of DVB-RCS. Until DVB-RCS came along, customers of two-way broadband access via satellite had no choice except to commit to propriety systems, with all its inherent inflexibility and higher cost. Interoperability gives customers the choice of purchasing from one or several vendors throughout the lifetime of their systems. The result will be improved competition among vendors, reduced costs for users and accelerated enhancement of DVB-RCS equipment.

2.2 UNDERSTANDING DVB-RCS

The first version of DVB-RCS was released in April 2000 [2]. Its main objective is the definition of an interaction channel via satellite for GEO satellite interactive networks. The standard mainly addresses air-interface issues: physical layer aspects such as modulation, coding, synchronization and medium access control procedures for sharing the return link, maintaining MF-TDMA as the reference Radio Transmission Technology (RTT). The system features a star architecture (Fig.2-1): satellite terminals (RCSTs) transmit towards the Hub (gateway station) over the medium access controlled return link, whereas the Hub uses a forward broadcast link for transmitting data towards RCSTs. All communications, including those between the two DVB-RCS terminals, have to be routed via the Hub.

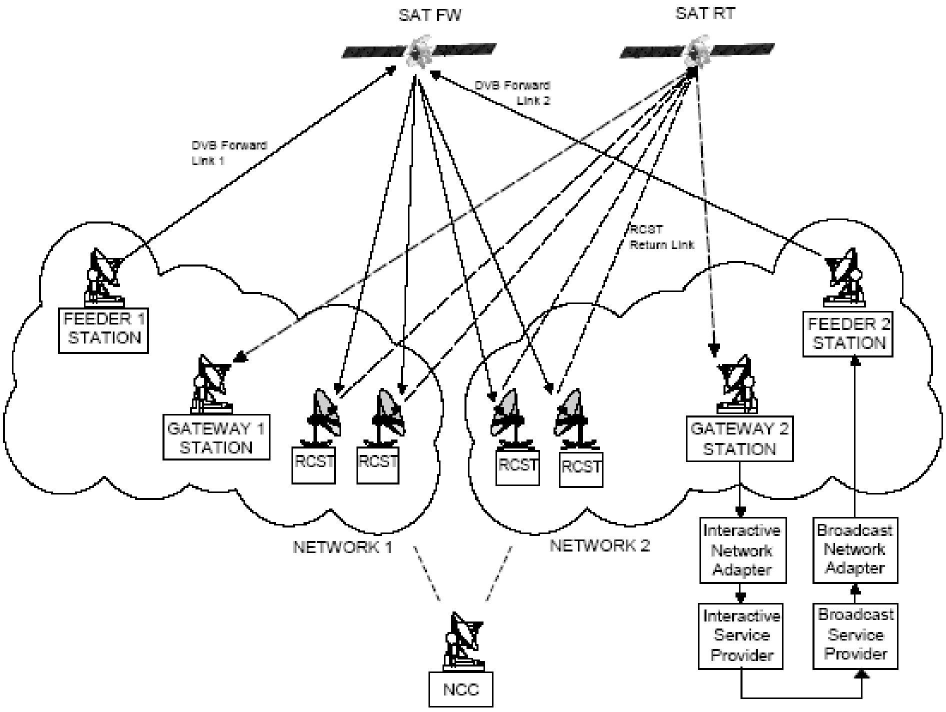

Figure 2-1 Reference architecture for DVB-RCS network [2]

- **Network Control Centre:** a NCC provides Control and Monitoring Functions (CMF). It generates control and timing signals for the operation of the Satellite Interactive Network to be transmitted by one or several Feeder Stations.

- **Traffic Gateway:** a TG receives the RCST return signals, provides accounting functions, interactive services and/or connections to external public, proprietary and private service providers (data bases, pay-per-view TV or video sources, software download, tele-shopping, tele-banking, financial services, stock market access, interactive games etc.) and networks (Internet, ISDN, PSTN, etc.).

- **Feeder:** a Feeder transmits the forward link signal, which is a standard satellite digital video broadcast (DVB-S or DVB-S2) uplink, onto which are multiplexed the user data and/or the control and timing signals needed for the operation of the Satellite Interactive Network.

In general the standard does not go into much detail, leaving great deal of implementation aspects open for individual operators to determine. At medium access control (MAC) level, the standard only describes capacity request categories that can be used as building blocks for implementing elementary MAC transfer capabilities over the satellite network. A separate technical report stands complementary to the standard specification, in providing guidelines for the actual standard implementation [3].

Interestingly, two types of RCS terminals are identified, called type A and B: the former supports IP traffic (Fig. 2-2) and the latter is envisaged for the support of native ATM protocols. User data are carried over two types of bursts, the one carrying ATM cells and the other (optional) MPEG-2 packets. IP traffic is encapsulated into either ATM cells via ATM adaptation layer 5 or, optionally, MPEG-2 bursts via Multi Protocol Encapsulation and is carried by the respective types of traffic bursts. ATM is therefore present in DVB-RCS, although in the case of RCST-A, its functionality is limited to the packetisation of the variable-length IP datagrams. There is no ATM signaling or ATM QoS framework.

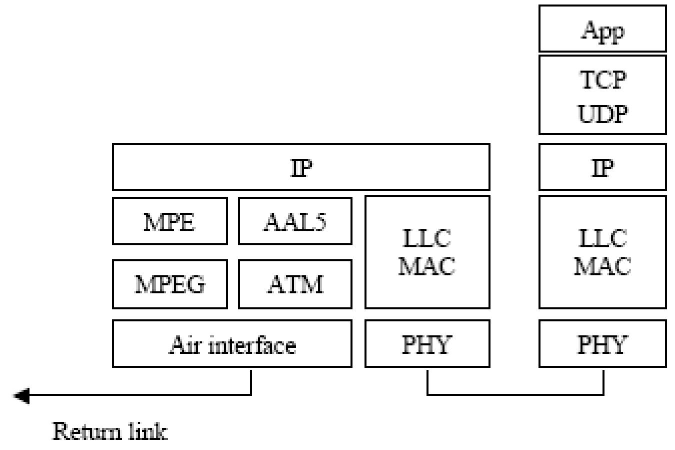

Figure 2-2 User plane protocol stack for type A RCS terminal [2]

The DVB-RCS standard enjoys the strong support of ESA. An expression of this support has been the set-up of two special interest groups, the ad-hoc RSAT group and, more recently, the SatLabs group. The RSAT group consists of major, mainly European, industrial players. It produced a report listing the changes that are necessary to expand the standard applicability to the regenerative satellite scenario[4], thus enabling single-hop mesh connectivity between DVBRCS terminals. Most of the proposed changes have been incorporated in the latest version of the DVB-RCS specification and the group is currently inactive. The SatLabs group introduces itself as an international, non-profit association, whose main objective is the large-scale adoption of the DVB-RCS standard as a platform for system interoperability. The group consists of many significant industrial players, ranging from satellite manufacturers to system integrators and service providers, and has recently produced its first set of recommendations [5].

2.3 CAPACITY REQUEST CATEGORIES [3]

Hereinafter, MSL refers to the minimum scheduler (i.e. the entity which generates the TBTP) latency. For example, the MSL (in frames) can be defined as the minimum time from the beginning of the frame in which a request is sent until the frame in which a corresponding assignment will apply. The MSL corresponds to the worst case round trip propagation delay from any RCST to scheduler and back again, plus any on-board delay at the satellite, plus scheduler processing delays, rounded up to a whole number of frames (figure 2-3) where the bandwidth of a frame is less or equal to the frequency hopping range of RCSTs, which is 20 MHz in this example.

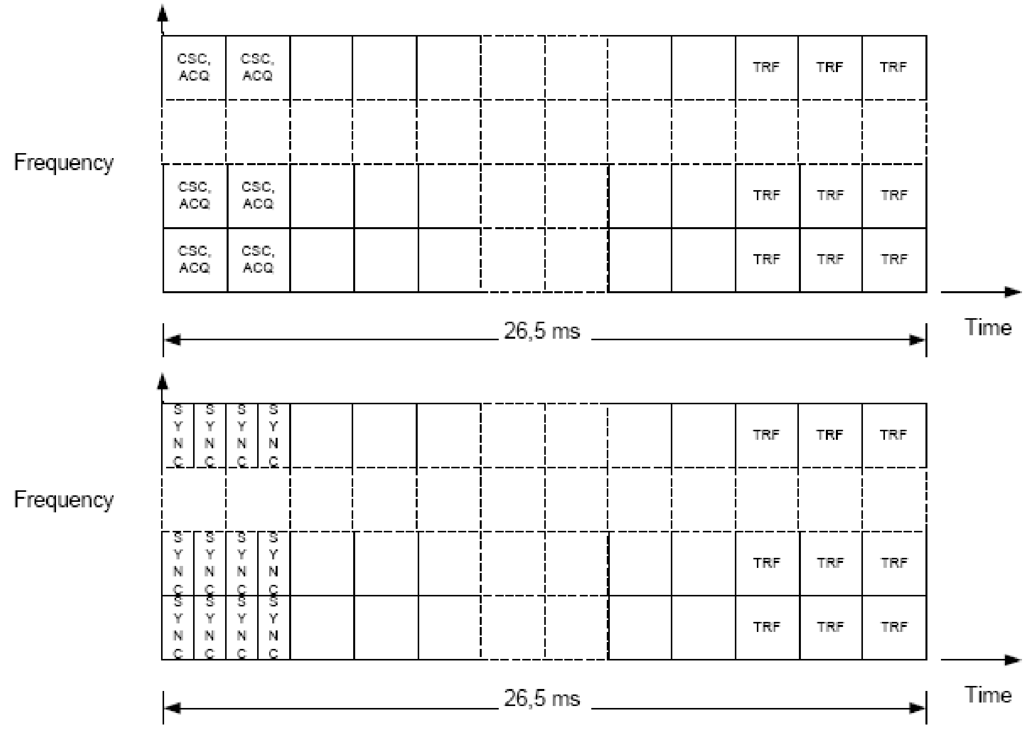

Figure 2-3 Example frame composition principle.

The sum of allocated or requested capacity for any given Return Channel Satellite Terminal (RCST) shall not exceed the maximum transmit capability of that RCST, or the maximum allowed transmit capability whichever is less.

The mapping between source traffic type and capacity category depends on the types of service provided, on the transmission protocols used and on constraints imposed by the satellite orbit. For these reasons, the following suggested mapping is only provided as examples.

In most networks, RCSTs transmit in all assigned time slots, even when they have no actual traffic to send. Some networks may prefer that RCST's generally do not transmit in this case. Such networks can occasionally force a transmission by means of the mechanism described below:

Assignment_type: the meaning of the field values in table 29 of EN 301 790:

• **00: One time assignment**: the slot(s) is (are) assigned only for this superframe.
• **01: Repeating assignment**: the slot(s) is (are) assigned in all superframes after the current one, until released.

• **10: Assignment release**: the slot(s) previously allocated are no longer useable by the RCST.

• **11: Forced_transmission one time assignment**: the RCST is forced to transmit in the burst(s), even if it has no traffic to send.

The assignment_type field is used to specify the type of allocation, which is granted to the RCST. In most networks,RCSTs transmit in all assigned time slots, even when they have no actual traffic to send. In these systems, the reserved value of the assignment_type field can be used exceptionally to force transmission in a burst.

It should be noted that the "forced_transmssion" uses the combination "11", which is currently a "reserved value" in the normative document.

2.3.1 CONTINUOUS RATE ASSIGNMENT (CRA)

CRA is rate capacity, which shall be provided in full for each and every superframe while required. Such capacity shall be negotiated directly between the RCST and the Network Control Center (NCC).

CRA should be used for traffic, which requires a fixed guaranteed rate, with minimum delay and minimum delay jitter, such as the Constant Bit Rate (CBR) class of Asynchronous Transfer Mode (ATM) traffic.

This category is also preferred for variable rate traffic, which cannot tolerate the Minimum Scheduler Latency (MSL) delay. An example of such traffic for a GEO satellite could be the ATM Variable Bit Rate - real time (VBR-rt) class.

2.3.2 RATE BASED DYNAMIC CAPACITY (RBDC)

RBDC should be used for variable rate traffic, which can tolerate the MSL delay. A typical application for RBDC over a GEO satellite could be the ATM Available Bit Rate (ABR) class.

RBDC is rate capacity, which is requested dynamically by the RCST. RBDC capacity shall be provided in response to explicit requests from the RCST to the NCC, such requests being absolute (i.e. corresponding to the full rate currently being requested). Each request shall override all previous RBDC requests from the same RCST, and shall be subject to a maximum rate limit negotiated directly between the RCST and the NCC.

To prevent a terminal anomaly resulting in a hanging capacity assignment, the last RBDC request received by the NCC from a given terminal shall automatically expire after a time-out period whose default value is 2 superframes, such expiry resulting in the RBDC being set to zero rate. The time-out can be configured between 1 and 15 superframes (if set to 0 the time out mechanism is disabled).

CRA and RBDC can be used in combination, with CRA providing a fixed minimum capacity per frame and RBDC giving a dynamic variation component on top of the minimum. A typical application could be the ATM Variable Bit Rate - non real time (VBR-nrt) class.

2.3.3 VOLUME BASED DYNAMIC CAPACITY (VBDC)

VBDC is volume capacity, which is requested dynamically by the RCST. VBDC capacity shall be provided in response to explicit requests from the RCST to the NCC, such requests being cumulative (i.e. each request shall add to all previous requests from the same RCST). The cumulative total per RCST shall be reduced by the amount of this capacity category assigned in each superframe.

VBDC should be used only for traffic that can tolerate delay jitter, such as the Unspecified Bit Rate (UBR) class of ATM traffic or standard IP traffic.

VBDC and RBDC can also be used in combination for ABR traffic, with the VBDC component providing a low priority capacity extension above the guaranteed limit in the RBDC category.

2.3.4 ABSOLUTE VOLUME BASED DYNAMIC CAPACITY (AVBDC)

AVBDC is volume capacity, which is requested dynamically by the RCST. This VBDC capacity shall be provided in response to explicit requests from the RCST to the NCC, such requests being absolute (i.e. this request replaces the previous ones from the same RCST). The AVBDC is used instead of VBDC when the RCST senses that the VBDC request might be lost (for example in the case of contention minislots). This might happen when requests are sent on contention bursts or when the channel conditions (PER, Eb/N0) are degraded. Traffic supported by AVBDC is similar to the VBDC one.

2.3.5 FREE CAPACITY ASSIGNMENT (FCA)

FCA is volume capacity, which shall be assigned to RCSTs from capacity, which would be otherwise unused. Such capacity assignment shall be automatic and shall not involve any signaling from the RCST to the NCC. It shall be possible for the NCC to inhibit FCA for any RCST or RCSTs.

It should be noted that the term "free" in FCA refers to "spare" system capacity and has no bearing on accounting. CRA and FCA can also be viewed as two mechanisms to grant dynamically capacity to a terminal, without requests being made from that terminal. This does not exclude the possibility that requests may have been made at a higher level than the terminal.

FCA should not be mapped to any traffic category, since availability is highly variable. Capacity assigned in this category is intended as bonus capacity, which can be used to reduce delays on any traffic, which can tolerate delay jitter.

2.4 QUEUING STRATEGY

An RCST may queue all traffic arriving from the user interface, using separate queues for traffic, which is subject to different transmission priorities. As an example, one queue shall be provided for each of the following priorities, where implemented:

- Real Time (RT) priority, corresponding to traffic carried using the CRA capacity category. Such traffic typically represents emulated circuit switched operation with tight constraint on end-to-end jitter build-up.

- Variable Rate (VR) priority, corresponding to traffic carried using the RBDC capacity category. Two VR traffic sub-priorities are possible: jitter sensitive (VR-Real Time or VR-RT) or jitter tolerant (VR-Jitter Tolerant or VR-JT). Where an RCST is required to support traffic with separate VR-RT and VR-JT components, then at least one queue shall be provided for each component with the VR-RT queue being the higher priority.

- Jitter Tolerant (JT) priority, corresponding to all other traffic i.e. that carried using the VBDC/AVBDC capacity category.

Queue lengths are a function of several factors including traffic profile, total system loading and congestion control methods. The queuing strategy for traffic classes using a combination of the above categories is not considered in the present document. However, it is likely that it requires a further queue per circuit source to allow context specific transmit processing. More queues may be required to meet network management constraints, such as the congestion control strategy. For example, the ATM explicit rate control for ABR traffic may require one queue per Virtual Circuit (VC).

2.5 REQUESTING STRATEGY

The capacity requesting strategy used by Type B RCSTs shall depend on the traffic priority. These strategies are defined below for the case where no congestion control is applicable.

2.5.1 RT PRIORITY TRAFFIC

The RCST shall not issue any requests for RT priority traffic, or for the RT priority component of mixed priority traffic. The capacity assigned to the RCST will be the CRA capacity.

2.5.2 VR PRIORITY TRAFFIC

VR priority traffic can be sent only where the RCST has negotiated a non-zero RBDC category limit with the NCC. Such traffic requires a request for RBDC capacity to be sent which matches the current demand.

The RCST shall calculate the total VR request required as the sum of the jitter sensitive component (VR-RT) and the jitter tolerant component (VR-JT). The VR-RT component shall be the amount of VR-RT traffic required to be sent in the frame being requested (i.e. one MSL in the future), and corresponds to that traffic of this class which was received during the prior frame period, less any part which is already allowed for in the RT priority traffic (CRA) capacity as a minimum capacity. If the resulting value is negative, then it shall be set to 0. The VR-JT component shall be the size of the current total VR-JT queue, after allowing for assignments in the current frame, less the total of pending VR-JT requests. A pending VR-JT request is defined as a request transmitted to the scheduler (or left active where no VR request update was sent) within the last MSL frames, i.e. the request or associated assignment is either in transit to/from the scheduler or being processed by the scheduler. If the resulting value is negative, then it shall be set to 0.

The total VR request shall be limited to the maximum RBDC rate. The resulting request shall be transmitted if it satisfies any of the following criteria:

- it is not equal to the last RBDC request.

• the time since the last RBDC request was sent is approaching the time-out value.

Since RBDC requests are non-cumulative, a duplicated transmission of each request may be advisable where the probability of a request loss is unacceptable in guaranteeing the QoS of the associated traffic class.

For VR-RT traffic, the assignment strategy must ensure ready availability of at least one CR opportunity for that RCST in each uplink frame. Where this is not implicitly guaranteed by other means, the simplest way of ensuring this is to always use a combined RBDC+CRA approach with the CRA component giving one or two slots per frame minimum assignment. A similar provision may also be needed for VR-JT traffic, where the QoS guarantees a given minimum latency.

To avoid a potential loss of VR communication, the requesting strategy shall ensure that no single loss of an RBDC request will trigger the time-out mechanism.

2.5.3 JT PRIORITY TRAFFIC

For JT priority traffic, the RCST shall calculate the total JT request required as the sum of a JT traffic component and a network management-messaging component.

The JT traffic component shall be the size of the current total JT queue, after allowing for assignments in the current frame, less the pending JT request. The pending JT request is defined as the rolling sum of all JT requests previously transmitted to the scheduler less the JT component of assignments already received i.e. it represents requests and assignments which are either in transit to/from the scheduler or stored in the scheduler. If the resulting JT component value is negative, then it shall be set to 0.

The network management-messaging component is the number of cells required for the network management messaging defined above. If the resulting total JT request is negative or zero, then no JT request shall be transmitted. Otherwise the RCST shall use as many VBDC/AVBDC CR as needed to transmit the total JT request, subject to availability, given that each transmitted request is limited to a maximum size and that such requests are cumulative/absolute. In the event of conflict between a need to transmit both RBDC and VBDC requests, then priority shall be given to RBDC.

2.6 DVB-RCS ON AD HOC NETWORKS

Now, we have an idea of what DVB-RCS is capable to do, but we could come up against some difficulties referred to the signaling transfer process.

For instance, such signal transmission is done at high frequencies ranging Gigahertzs, the frequency has its endemic qualities with relations to magnetic fields and can be interfered by heavy clouds. This explains DVB-RCS signal loss during rain or heavy clouds, with the length of the signal loss being dependent on the extent of rain, and the heaviness of the clouds above the reception dish. We could come up against infrastructure problems, too. Because of a bad situation or position of the antenna which means a bad reception of the signal for the RCST.

Assuming an integrated system where the RCSTs are the nodes of an AHN, we could develop a network system where every node (terminal) is capable to link with the satellite through different relays. This relays are the nodes of the AHN we want to simulate in this paper. Our job will be studying the nodes behavior to improve the global system.

3 AD-HOC NETWORK

3.1 UNDERSTANDING AD-HOC NETWORKS [7]

In this Chapter, we are going to explain how ad-hoc networks work, their pros and cons to consider and some alternatives to correct the arisen problems. All the issues discussed further are related to our project which is find the best way to direct the load traffic through an ad-hoc network over a DVB standard.

Most installed wireless LANs today utilize "infrastructure" mode that requires the use of one or more access points. With this configuration, the access point provides an interface to a distribution system (e.g., Ethernet), which enables wireless users to utilize corporate servers and Internet applications.

As an optional feature, however, the 802.11 standard specifies "ad hoc" mode, which allows the radio network interface card (NIC) to operate in what the standard refers to as an Independent Basic Service Set (IBSS) network configuration. With an IBSS, there are no access points. User devices communicate directly with each other in a peer-to-peer manner.

This way, Ad hoc mode allows users to spontaneously form a wireless LAN. Through ad hoc mode, you can easily transfer the file from one laptop to another. With any of these applications, there's no need to install an access point and run cables.

Some product vendors are beginning to base their solutions on ad hoc mode. As an example, Mesh Networks offers a wireless broadband network system based on 802.11 ad hoc mode and a patented peer-to-peer routing technology. This results in a wireless mesh topology where mobile devices provide the routing mechanisms in order to extend the range of the system. For example, a user on one side of the building can send a packet destined to another user on the far side of the facility, well beyond the point-to-point range of 802.11, by having the signal hope from client device to client device until it gets to its destination. This can extend the range of the wireless LAN from hundreds of feet to miles, depending on the concentration of wireless users.

3.1.1 THE AD-HOC NETWORK OPERATIVE

Much of the 802.11 standard define a common operation whether you're using ad hoc or infrastructure mode. The use of ad hoc mode only affects the protocols, so there is no impact on the Physical Layers (i.e., 802.11a and 802.11b). Within the MAC Layer, all of the carrier sensing and most of the frame types and corresponding usage are the same regardless of which mode you choose. The absence of an access point, however, means that an ad hoc wireless LAN must take on more of the MAC Layer responsibilities.

The first ad hoc station (radio NIC) active establishes an IBSS and starts sending beacons, which are needed to maintain synchronization among the stations. (With infrastructure mode, only the access point sends beacons.) Other ad hoc stations can join the network after receiving a beacon and accepting the IBSS parameters (e.g., beacon interval) found in the beacon frame.

All stations that join the ad hoc network must send a beacon periodically if it doesn't hear a beacon from another station within a very short random delay period after the beacon is supposed to be sent. The random delay minimizes the transmission of beacons from multiple stations by effectively reducing the number of stations that will send a beacon. If a station doesn't hear a beacon within the random delay period, then the station assumes that no other stations are active and a beacon needs to be sent.

After receiving a beacon, each station updates their local internal clock with the timestamp found in the beacon frame, assuming the timestamp value is greater than the local clock. This ensures that the all stations are able to perform operations, such as beacon transmissions and power management functions, at the same time.

As with infrastructure networks, an ad hoc sleeping station (i.e., power management "on") indicates that they're entering sleep state by setting the power management bit in the control field of any frame. All other stations learn of this by monitoring the frame control fields of all frames. Stations will then hold off transmitting to the sleeping station and buffer the corresponding packets locally.

Regularly, all sleeping stations wake up at the same time during the announcement traffic indication map (ATIM) window, which corresponds with each beacon transmission. If a station is holding packets for a sleeping destination, the station will send an ATIM frame to the sleeping station indicating that packets are awaiting transmission. The station that had been asleep then knows to stay awake through the next beacon interval, which is hopefully long enough for the station buffering the packet to send it successfully. After receiving and acknowledging reception of the packet, the station can go back to sleep.

Then, we can state ad hoc mode offers enough advantages to consider when deploying wireless LANs. The thought of saving the cost on access points is certainly a compelling reason to strongly consider this configuration. Unless you implement routing among the wireless users, however, you'll find that ad hoc mode mostly applies to smaller, spontaneous networks when there isn't a strong need for interfacing with a wired network.

3.1.2 PROS AND CONS TO CONSIDER

Rapid setup time. Ad hoc mode only requires the installation of radio NICs in the user devices. As a result, the time to setup the wireless LAN is much less than installing an infrastructure wireless LAN. Obviously this timesaving only applies if the facility you plan to support wireless LAN connectivity doesn't already have a wireless LAN installed.

Better performance possible. The question of performance with ad hoc mode is certainly debatable. For example, performance can be higher with ad hoc mode because of no need for packets to travel through an access point. This assumes a relatively small number of users, however. If you have lots of users, then you'll likely have better performance by using multiple access points to separate users onto non-overlapping channels to reduce medium access contention and collisions. Also because of a need for sleeping stations to wake up during each beacon interval, performance can be lower with ad hoc mode due to additional packet transmissions if you implement power management.

Limited network access. Because there is no distribution system with ad hoc wireless LANs, users don't have effective access to the Internet and other wired network services. Of course you could setup a PC with a radio NIC and configure the PC with a shared connection to the Internet. This won't satisfy a larger group of users very well, though. As a result, ad hoc is not a good way to go for larger enterprise wireless LANs where there's a strong need to access applications and servers on a wired network.

Difficult network management. Network management becomes a headache with ad hoc networks because of the fluidity of the network topology and lack of a centralized device. Without an access point, network managers can't easily monitor performance, perform security audits, etc. Effective network management with ad hoc wireless LANs requires network management at the user device level, which requires a significant amount of overhead packet transmission over the wireless LAN. This again leans ad hoc mode away from larger, enterprise wireless LAN applications.

3.1.3 MESH NETWORKING [8]

Mesh networking is a way to route data, voice and instructions between nodes. It allows for continuous connections and reconfiguration around broken or blocked paths by "hopping" from node to node until the destination is reached. A mesh network whose nodes are all connected to each other is a fully connected network. Mobile ad-hoc networking (MANET), featured in many consumer devices, is a subsection of mesh networking.

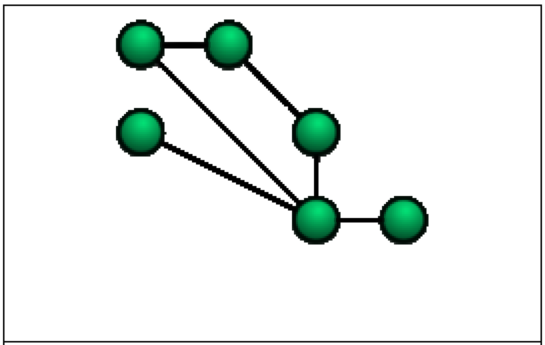

Figure 3-1 Image showing mesh network layout

Mesh networks are self-healing: the network can still operate even when a node breaks down or a connection goes bad. As a result, a very reliable network is formed. This concept is applicable to wireless networks, wired networks, and software interaction.

A mesh network is a networking technique, which allows inexpensive peer network nodes to supply back haul services to other nodes in the same network. It effectively extends a network by sharing access to higher cost network infrastructure.

Mesh networks differ from other networks in that the component parts can all connect to each other via multiple hops, and they generally are not mobile.

3.2 ROUTING PROTOCOLS [9]

An *Ad hoc routing protocol* is a convention or standard that controls how nodes come to agree which way to route packets between computing devices in a mobile ad-hoc network (MANET).

In ad hoc networks, nodes do not have *a priori* knowledge of topology of network around them, they have to discover it. The basic idea is that a new node (optionally) announces its presence and listens to broadcast announcements from its neighbors. The node learns about new near nodes and ways to reach them, and may announce that it can also reach those nodes. As time goes on, each node knows about all other nodes and one or more ways how to reach them.

Routing algorithms have to

- Keep routing table reasonably small
- Choose *best* route for given destination (this can be the fastest, most reliable, highest throughput, or cheapest route)
- Keep table up-to-date when nodes die, move or join
- Require small amount of messages/time to converge

Note that in a wider context, an **ad hoc protocol** can also mean an improvised and often impromptu protocol established for a particular specific purpose.

A MANET consists of wireless hosts that move around, i.e. they have no permanent physical location. In order to facilitate communication within the network, a routing protocol is used to discover routes between nodes before the exchange of IP data packets. Below is a brief overview of IP routing in an Ad Hoc environment.

The routing protocols in Ad Hoc wireless networks are generally categorised as:

3.2.1.1 PROACTIVE

These protocols require each node to maintain one or more tables to store up to date routing information and to propagate updates throughout the network. These protocols try and maintain valid routes to all communication mobile nodes all the time, which means before a route is actually needed. Periodic route updates are exchanged in order to synchronise the tables.

Some examples of table driven ad hoc routing protocols include Dynamic Destination Sequenced Distance-Vector Routing Protocol (DSDV), Optimized Link State Routing Protocol (OLSR) and Fisheye State Routing Protocol (FSR). These protocols differ in the number of routing related tables and how changes are broadcasted in the network structure[12].

The problem with these protocols is the overhead; the protocols propagate and maintain routing information, regardless of whether or not it is needed.

3.2.1.2 REACTIVE

These protocols create routes only when desired by a source node, therefore a route discovery process is required within the network. Once a route has been established, it is maintained by a route maintenance procedure until either the destination becomes inaccessible or until the route isn’t needed any longer.

Some examples of source initiated ad hoc routing protocols include the Dynamic Source Routing Protocol (DSR), Ad Hoc On Demand Distance Vector Routing Protocol (AODV), and Temporally-Ordered Routing Algorithm (TORA). No periodic updates are required for these protocols but routing information is only available when hended[12].

3.2.1.3 HYBRID

These protocols try to incorporate various aspects of proactive and reactive routing protocols. They are generally used to provide hierarchical routing; routing in general can be either flat or hierarchical in a flat approach, the nodes communicate directly with each other. The problem with this is that it does not scale well, it also does not allow for route aggregation of updates

In a hierarchical approach, the nodes are grouped into clusters, within each cluster there is a cluster head, this acts as a gateway to other clusters, it serves as a sort of default route. The advantage of a hierarchical structure is that within a cluster, an on demand routing protocol could be used which is more efficient in small-scale networks. For inter cluster communication then a table driven protocol could be used which, would allow the network to scale better. An example of such a hybrid routing protocol is the Zone Routing Protocol (ZRP) [12].

3.2.1.4 OTHER TYPES OF ROUTING PROTOCOLS

There are many other types of ad hoc routing protocols; one for example LANMAR [12] uses location info, obtained using the Global Positioning System (GPS). By knowing the precise location of a node you can limit the search to a smaller “request zone” of the network.

3.2.2 SOME AD-HOC ROUTING PROTOCOLS

- On Demand:

  DSR (Dynamic Source Routing)

- Vector:

  DSDV (Destination-Sequenced Distance Vector)

- Hierarchy:

  ZRP (Zone Routing Protocol)

- Mixed and Others:

  CBRP (Cluster Based Routing Protocol)

  LANMAR (Landmark Routing Protocol)

  TBRPF (Topology Broadcast Based on Reverse – Path Forwarding)

  AODV (Ad-hoc On-demand Distance Vector Routing)

  TORA (Temporally-Ordered Routing Algorithm)

  LAR (Location Aided Routing)

  OLSR (Optimized Link State Routing)

  FSR (Fisheye State Routing)

Criterion	AODV	DSR	OLSR	FSR	CBRP	LANMAR	TBRPF	ZRP
Without loop	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Many routes possible	No	Yes	No	No	Yes	No	No	Yes
Distributed	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Kind	reactive	reactive	proactive	proactive	hybrid	hybrid	proactive	hybrid
Security	No	No	No	No	No	No	No	No
Periodic messages control	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Unidirectional links	No	Yes	Yes	Yes	Yes	No	No	Yes

Figure 3-2 Routing Protocols Characteristics

The studied metrics to analyze a concrete routing protocol are:

The loss rate: important because the retransmission of the data is managed at transport level and consequently can influence the maximum throughput that the network supports.

The routing overhead: it must be the less possible to optimize the band-width of the network. It is measured as a number of packets.

The relevance of the path: Its the difference between the path taken by the data and the existing shortest path between the source and the destination. That shows the capacity of the protocol to find most efficient paths in terms of a number of intermediate nodes.

3.3 LOAD BALANCING [10]

Computer networks are complex systems, often routing hundreds, thousands, or even millions of data packets every second. Therefore, in order for networks to handle large amounts of data, it is important that the data is routed efficiently. For example, if there are ten routers within a network and two of them are doing 95% of the work, the network is not running very efficiently. The network would run much faster if each router were handling about 10% of the traffic. Likewise, if a website gets thousands of hits every second, it is more efficient to split the traffic between multiple Web servers than to rely on a single server to handle the full load.

Load balancing helps make networks more efficient. It distributes the processing and traffic evenly across a network, making sure no single device is overwhelmed. Web servers, as in the example above, often use load balancing to evenly split the traffic load among several different servers. This allows them to use the available bandwidth more effectively, and therefore provides faster access to the websites they host.

Whether load balancing is done on a local network or a large Web server, it requires hardware or software that divides incoming traffic among the available servers. Networks that receive high amounts of traffic may even have one or more servers dedicated to balancing the load among the other servers and devices in the network. These servers are often called (not surprisingly) load balancers.

Clusters, or multiple computers that work together, also use load balancing to spread out-processing jobs among the available systems (Figure 3-4).

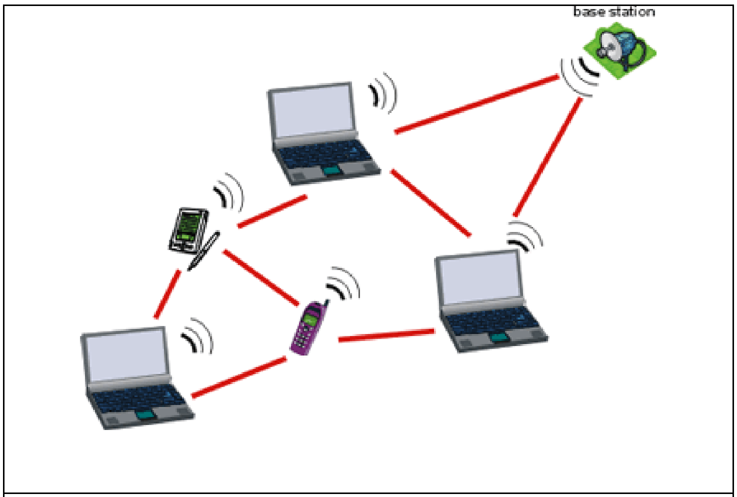

Figure 3-3 Cluster of devices balancing traffic

3.4 GAME THEORY IN TELECOMMUNICATIONS [11]

Game theory has been applied to many fields of telecommunications. It is a good tool when analytical results concerning selfish users are needed. Especially the growth and commercialization of the Internet has required a new point of view. Instead of a homogeneous network where users use the agreed protocols, the Internet is today often modeled to be consisted of selfish users trying to maximize their quality of service.

The term game theory is sometimes used vaguely in the context of telecommunications. Approaches discussing selfish users are called game theoretic, even if they do not have any formal game theoretic analysis. When a telecommunication system is modeled using game theory, there are some properties that are of interest. Is there a Nash equilibrium? Is it unique? Does the system converge to the equilibrium point? Is it also a system wide optimum, i.e. does it maximize the *social welfare*?

We briefly cover some fields of telecommunications in which game theory has been applied. Most importantly, the applications to ad hoc networks are introduced. Also, some game theoretic research of the Internet is discussed in less detail. Finally, we introduce some research in other areas that may give insight into the AHNs. Game theory has been applied to the financial problems of telecommunications but they are not in the scope of this thesis.

Game theoretic research regarding AHNs has been focused on the cooperation of the nodes. While the mechanisms introduced try to provide means to prevent selfishness and to enforce cooperation, the game theoretic research considers the same problem using a more analytical viewpoint.

3.5 STATE OF THE ART [12]

3.5.1 DYNAMIC CONFIGURATION OF IP ADDRESSES

In order to communicate nodes need IP addresses. Since Ad Hoc networks lack any centralised administration these addresses can't be manually configured, and so must be configured dynamically. In a wired network dynamic configuration is achieved using the Dynamic Host Configuration Protocol (DHCP), however this requires the presence of a centralised DHCP server, which maintains the configuration information of all nodes in the network. Since an Ad Hoc network is devoid of any fixed infrastructure such as a central server, this approach can't be used.

The problem in regard to Internet connectivity for IPv4, is that most of the proposals for dynamic address allocation assumes the use of private addresses, due to the difficulty in obtaining global addresses. There is an issue here in regard to connectivity to the Internet, as some sort of network address translation (NAT) process will be required, this is a process that converts a private address into a unique global address. In the wired environment NAT is achieved using "Traditional Network Translation (NAT)", this is a process that converts a non-unique IP address to a unique IP address. A solution is proposed in [13] "Connectivity for IPv4 Mobile Ad Hoc Networks" however it assumes that each node in the MANET is using Mobile IP and as a result already has a globally unique home address. It assumes that a Foreign Agent assigns "care of addresses" to nodes as they arrive into the network and acts as a gateway for them when they want to connect to the Internet.

3.5.2 DYNAMIC ALLOCATION PROPOSALS

When a node is assigned an address, it needs to find out whether or not the address it chooses in unique within its network. In order to determine this, a process known as duplicate address detection (DAD) is performed, it is basically a process that determines whether or not a chosen IP address is unique within a chosen network. An overview of different DAD techniques is presented in . Generally the different proposals differ in the technique they use to perform DAD. The proposals fall into two different categories.

3.5.2.1 HIERARCHICAL APPROACH

In a hierarchical approach, a clustering technique is used with one node (cluster head) assuming the responsibility for the allocation of addresses to new nodes as they arrive, basically when a new node arrives, he registers with the cluster head, who then allocates a new addresses and co ordinates a duplicate address (DAD) process in order to determine whether or not the address chosen is unique within the ad hoc network. The following paper is based on this approach.

A new node entering the network, hereafter called the “requester”, chooses a reachable node as the “initiator”, which performs address allocation on its behalf. All other nodes know the route to the initiator and can forward their responses to it. The initiator chooses an address it perceives as unallocated and attempts to acquire permission from all other nodes in the network to assign the address to the requester. Nodes perceiving this address as unallocated mark the requested address as allocation in progress and reply in affirmative to the initiator. This allocation is made permanent by a second message, which is sent by the initiator if the initiator receives an affirmative response from all nodes in the network. Therefore the IP address allocation is similar to a two-phase commit.

Nodes, which no longer wish to be part of the system, relinquish their address by broadcasting a message to the effect before leaving the network. If a node abruptly leaves the network, i.e. goes down without relinquishing its address, it would fail to respond to the address allocation request by some initiator the next time a requester enter the network. In this case the address of the departed node is cleaned up by the initiator awaiting a reply from the departed node.

Merging works differently, here a unique partition ID is used. When two nodes come into contact and exchange their partition ID’s, they realise that they are different partitions and so merge. To do this they exchange partition ID’s as well as the table, which shows all IP address that have been allocated. If it is found that nodes in the partitions have the same IP address, then one node gives up its IP address and requests another one.

3.5.2.2 FLAT TOPOLOGY APPROACH

In a flat topology approach there is no cluster head, which assumes responsibility for the allocation. Here when a node joins the network and wants an IP address, it chooses an address at random and then performs a duplicate address procedure in order to determine whether or not that address is unique.

The process us based on a proactive routing protocol e.g. Ad Hoc On Demand Distance Vector Routing (AODV) and uses a flat structure. In a proactive routing protocol routing is done on an on demand basis, in order to route to a destination a AREQ message is sent out looking for the destination. When the destination is found an AREP message is sent back indicating that the destination is reachable.

A node performing the auto-configuration process picks two addresses, a temporary address and the actual address to use. The former is used only once in the uniqueness check to minimise the possibility for it to be non unique. The unique check is based on sending as address request (AREQ) and expecting an address reply (AREP) back in case the address is not unique. In case no AREP is received, the uniqueness check is passed.

3.5.3 FUTURE DIRECTIONS

• Merging of Networks

The partitioning and merging of networks has not been addressed in detail. One potential area of research here is in the service discovery, if two networks are to merge together there will presumably be a reason to do so, for example one network may offer printing facilities to another network, issues of service discovery are important here. Other potential areas here include methods for detecting partitioning and merging of networks.

• Security in the auto configuration process has not been addressed, denial of service attacks are one possible security flaw, one node for example may request all the potential IP addresses available.

• Internet Connectivity

This problem is closely related to the routing problem, and the problem differs depending on whether you are using IPv4 or IPv6.

• The applicability of Mobile IP in Ad Hoc networks.

• Routing

Could the IP address assignment process be optimised for different IP address assignment protocols? For example if a hierarchical routing protocol is used, which utilizes clusters and cluster heads. Would it be more efficient for a hierarchical IP address assignment protocol to use the clusters and cluster heads identified by the routing protocol or to create its own. Also could routing information be used in the address assignment protocol, for example, if a node finds that it cant route information to a particular node, can it assume that that node has left the network? This kind of information could be useful for the IP address assignment protocol as it may allow nodes to identify the departure of a node more quickly.

• IPv4 vs. IPv6

Should the IP address assignment solution be independent of IP version in use, i.e. will one solution work for IPv4 and IPv6 or will two different solutions be needed?

4 PROBLEM STATEMENT

4.1 UNDERSTANDING THE COOPERATION IN THE WIRELESS AD-HOC NETWORKS [6]

In wireless ad hoc networks, nodes communicate with far off destinations using intermediate nodes as relays. Since wireless nodes are energy constrained, it may not be in the best interest of a node to always accept relay requests. On the other hand, if all nodes decide not to expend energy in relaying, then network throughput will drop dramatically. Both these extreme scenarios (complete cooperation and complete no cooperation) are inimical to the interests of a user. In this paper we address the issue of user cooperation in ad hoc networks. We assume that nodes are rational, i.e., their actions are strictly determined by self-interest, and that each node is associated with a minimum lifetime constraint. Given these lifetime constraints and the assumption of rational behavior, we are able to determine the optimal throughput that each node should receive. We define this to be the rational Pareto optimal operating point. We then propose a distributed and scalable acceptance algorithm called Generous TIT-FOR-TAT (GTFT). The acceptance algorithm is used by the nodes to decide whether to accept or reject a relay request. We show that GTFT results in a Nash equilibrium and prove that the system converges to the rational and optimal operating point.

The limitation of finite energy supply raises concerns about the traditional belief that nodes in ad hoc networks will always relay packets for each other. Consider a user in a campus environment equipped with a laptop. As part of his daily activity, the user may participate in different ad-hoc networks in classrooms, the library and coffee shops. He might expect that his battery-powered laptop will last without recharging until the end of the day. When he participates in these different ad hoc networks, he will be expected to relay traffic for other users. If he accepts all relay requests, he might run out of energy prematurely. Therefore, to extend his lifetime, he might decide to reject all relay requests. If every user argues in this fashion, then the throughput that each user receives will drop dramatically. We can see that there is a trade-off between an individual user’s lifetime and throughput.

The acceptance algorithm is used to decide whether to accept or reject a packet relay request. The acceptance algorithm at each node attempts to balance the number of packets it has relayed with the number of its packets that have been relayed by others. The problem of this scheme is that it involves for each packet processing which results in large overheads. We propose an algorithm used by the network nodes to decide whether to relay traffic on a per session basis. The goal of this algorithm is to balance the energy consumed by a node in relaying traffic for others with energy consumed by other nodes in relaying traffic and to find an optimal trade-off between energy consumption and session blocking probability. By taking decisions, the packet-processing overhead is eliminated. We emphasize, that the algorithm is based on heuristics and lack a formal framework to analyze the optimal trade-off between lifetime and throughput.

We will consider a finite population of N=4 nodes (e.g., students on a campus). Each node, depending on its type (e.g., laptop, PDA, cell phone), is associated with an average power constraint. This constraint, can be derived by dividing its initial energy allocation by its lifetime expectation. We deal with connection-oriented traffic. At the beginning of each slot, a source, destination and several relays are randomly chosen out of the 4 nodes to form an ad hoc network (e.g., students in a coffee shop). The source requests the relay nodes in the route to forward its traffic to the destination. If any of the relay nodes rejects the request, the traffic connection is blocked.

For each node, we define the *Normalized Acceptance Rate* (NAR) as the ratio of the number of successful relay requests generated by the node, to the number of relay requests made by the node. This quantity is an indication of the throughput experienced by the node. Then, we study the optimal tradeoff between the lifetime and NARs of the nodes. In particular, given the energy constraints and the lifetime expectation of the nodes, we identify the feasible set of NARs. This provides us with a set of Pareto optimal values, i.e., values of NAR such that a node cannot improve its NAR without decreasing some other node’s NAR. By assuming the nodes to be rational, i.e., that their actions are strictly determined by self interest, we are able to identify a unique set of *rational and Pareto optimal* NARs for each user.

Since users are self-interested and rational, there is no guarantee that they will follow a particular strategy unless they are convinced that they cannot do better by following some other strategy. In game theoretic terms, we need to identify a set of strategies, which constitute a Nash equilibrium1. Ideally, we would like the Nash Equilibrium to result in the rational and Pareto optimal operating point. We achieve this by proposing a distributed and scalable acceptance algorithm, called Generous TIT-FOR-TAT (GTFT). We prove that GTFT is a Nash Equilibrium which converges to the rational and Pareto optimal NARs.

4.2 SYSTEM MODEL

We'll consider a finite population of N nodes distributed among K classes. Let ni be the number of nodes in class i ($i = 1, . . . K$). All nodes in class are associated with an energy constraint, denoted by E_i, and an expectation of lifetime, denoted by L_i. Based on these requirements, we contend that nodes in class i have an average power constraint of $\rho_i = E_i/L_i$. We assume that $\rho_1 > \rho_2 > . . . > \rho_K$. The system operates in discrete time. In each slot, any one of the N nodes can be chosen as a source with equal probability. M is the maximum number of relays that the source can use to reach its destination. The probability that the source requires l _ M relays is given by $q(l)$. For the sake of simplicity, in our study we assume $q(0) = 0$, i.e., there is at least one relay in each session. This assumption can be easily relaxed by subtracting the energy spent in direct transmissions from the total energy budget of each node. The l relays are chosen with equal probability from the remaining $N-1$ nodes. We assume that each session lasts for one slot. In this time interval, the source along with the l relays forms an ad hoc network that remains unchanged for the duration of the slot.

The source requests the relay nodes to forward its traffic to the destination. A relay node has the option to either accept or refuse the request. We assume that a relay node communicates its decision to the source by transmitting either a positive or a negative acknowledgment. If a negative acknowledgment is sent, the traffic session is blocked.

A node spends energy in transmitting, receiving and processing traffic. We assume that energy spent in transmit mode is the dominant source of energy consumption; The energy consumed by the nodes in transmitting a session will depend on several factors like the channel conditions, the file size, and the modulation scheme. Here, we assume that the energy required to relay a session is constant and equal to 0.0005 per packet sent, it allows us to capture the salient aspects of the problem. We believe that the ideas presented in this paper can be extended to more realistic settings.

Finally, for a generic node *h*, we denote by B_h the number of relay requests made by node *h*, and by A_h the number of relay requests generated by node *h*. Equivalently, we denote by D_h the number of relay requests made to node *h*, and by C_h the number of relay requests made to node *h*.

It's defined: $\phi_h = A_h/B_h$, and $\psi_h = C_h/D_h$. Observe that ϕ_h is the ratio of the number of relay requests by *h* which have been accepted, to the number of requests made by *h*; thus, ϕ_h is an indication of the throughput experienced by *h*. The Normalized Acceptance Rate (NAR) is defined as NAR = $lim\ \phi_h$. Note that the NAR is defined for each node, however, we have suppressed the indices for the sake of simplicity. From the above definitions it is clear that the throughput of a node is determined by its values of NAR. In the following we will equivalently refer to NARs and throughput.

4.3 THE GTFT ALGORITHM

4.3.1 SIMULATION OBJECTIVES

In this work, our objective is to provide a framework for studying user cooperation in ad hoc networks and to define behavioral strategies that lead the system to the optimal operating point. Several implementation aspects however need to be addressed.

In this section, we present a distributed acceptance algorithm, which propels the nodes to operate at the rational Pareto optimal NARs. This algorithm is called the Generous TIT-FOR-TAT (GTFT) algorithm.

4.3.1.1 SIMULATION DESIGN

In a network of self-interested nodes, each node will decide on those actions, which will provide it maximum benefit. Any strategy that leads such users to the rational optimal NARs should possess certain features. Firstly, it cannot be a randomized stationary policy. If a node in class *i* gets a request, then a possible course of action would be to accept that request with probability τ_h . If all nodes were to use this policy, then the rational optimal τs can be used to achieve the optimal operating point. However, a rational selfish node will exploit the naivete of other nodes by always denying their relay requests thereby increasing its lifetime, while keeping its NAR constant. In other words, in our system, any stationary strategy is dominated by the always deny behavior. Hence, stationary strategies are not sustainable, and behavioral strategies are required in order to stimulate cooperation. By behavioral strategies, we mean that a node bases its decision on the past behavior of the nodes in the system. The second feature, which we would like an acceptance algorithm to have, is protection from exploitation.

Our problem falls in the framework of Non-Cooperative Game Theory. There, the canonical example is that of the Prisoners Dilemma. In this example, two people are accused of a crime. The prosecution promises that, if exactly one confesses, the confessor goes free, while the other goes to prison for ten years. If both confess, then they both go to prison for five years. If neither confesses, they both go to prison for just a year. Table I presents the punishment matrix showing the years of prison that the players get depending on the decision they make. Clearly, the mutually beneficial strategy would be for both not to confess. However, from the perspective of the first prisoner, P1, his punishment is minimized if he confesses, irrespective of what the other prisoner, P2, does. Since the other prisoner argues similarly, the unique Nash Equilibrium is the confess strategy for both prisoners. Nevertheless, if this game were played repeatedly (Iterated Prisoners Dilemma), it has been shown that cooperative behavior can emerge as a Nash equilibrium. By employing behavioral strategies, a user can base his decision on the outcomes of previous games. This allows the emergence of cooperative equilibrium. A well known strategy to achieve this desirable state of affairs is the Generous TIT-FOR-TAT (GTFT) strategy. In the Generous TIT-FOR-TAT strategy, each player mimics the action of the other player in the previous game. Each player, however, is slightly generous and on occasion cooperates by not confessing even if the other player had confessed in the previous game. We have adapted the GTFT algorithm to our problem.

P2 P1	Confess	Not confess
Confess	(5,5)	(0,10)
Not Confess	(10,0)	(1,1)

Table 1. Punishment matrix for the prisoner’s dilema. The fist entry refers to prisoner P1’s prison term and the second one to prisoner P2’s prison term.

In our algorithm, each node maintains a record of its past experience by using the two variables ϕ_h and ψ_h , $h = 1, . . . N$. Each node therefore maintains only information per session type and does not maintain individual records of its experience with every node in the network. The decisions are always taken by the relay nodes based only on their ϕ_h and ψ_h values. We consider the case with N nodes, K classes, $q(1) = 1$ and $M = 1$, i.e., each session uses only one relay. Assume that a generic node h receives a relay request. Let ε be a small positive number. The acceptance algorithm, which we call the GTFT algorithm is as follows.

- If $\psi_h > \tau_h$ *or* $\phi_h < \psi_h - \varepsilon$ →Reject
- Else →Accept .

Thus, a request is refused if either $\phi_h > \tau_h$, where is $\tau_h = {}^{N\rho}\!\big/_{2}$ i.e., node h has relayed more traffic than what it should, or $\phi_h < \psi_h - \varepsilon$ i.e., the amount of traffic relayed by node h is greater than the amount of traffic relayed for node h by others. Since ε is positive, nodes are a little generous by agreeing to relay traffic for others even if they have not received a reciprocal amount of help.

The GTFT algorithm has the following desirable properties:

- It is not a stationary strategy.
- Each node takes its action based solely on locally gathered information; this prevents a node from being exploited.

4.3.1.2 THE FLOW DIAGRAM

In this chapter we try to explain the algorithm to either reject or accept a packet from a source. We will use flow charts to understand the algorithm easier.

The algorithm is called by the function ‘main’ where contains the principle functions to develop the simulation. It will create a vector, which has the time when the nodes will request for the relays. For the sake of organization, in this function we will call two subroutines: initiate_parameters and find_source, which will be explained later.

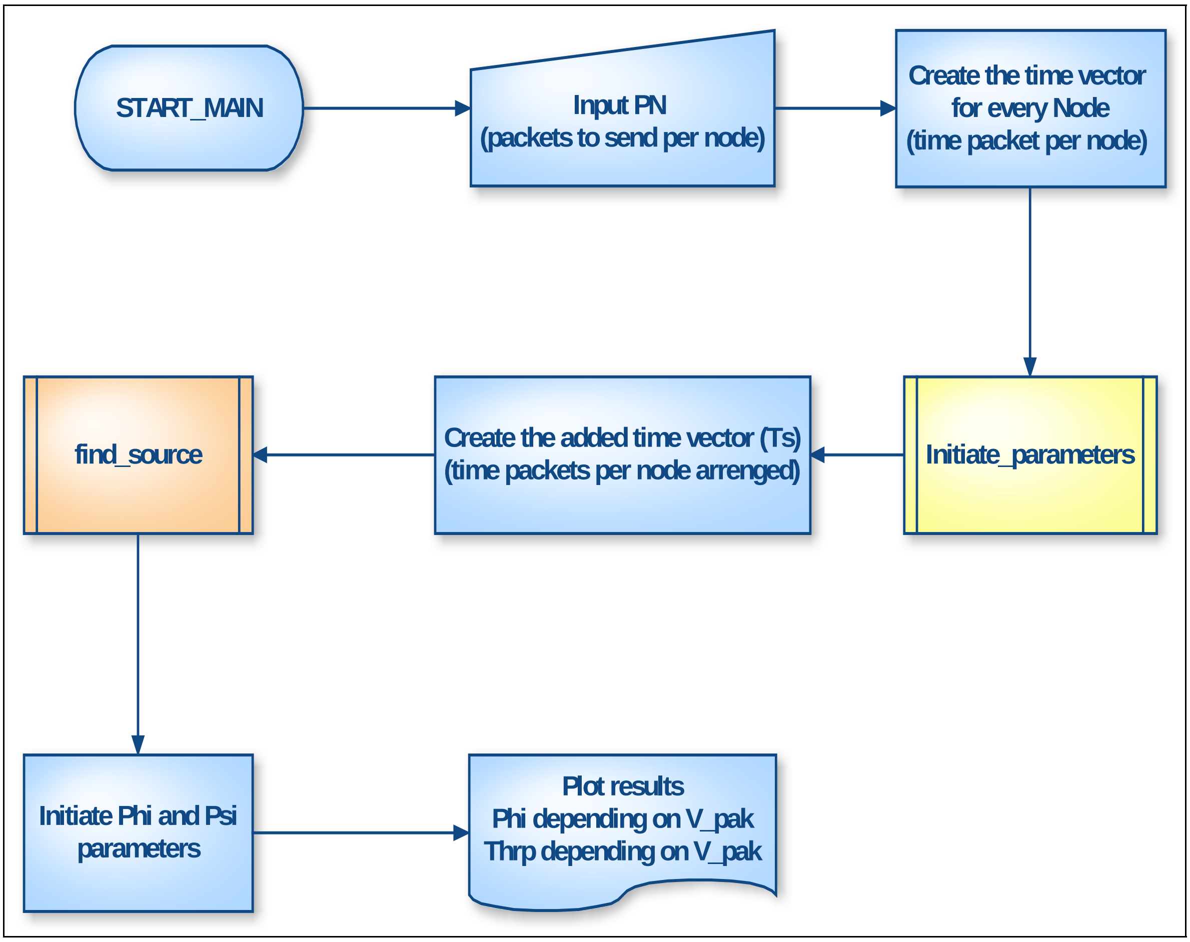

Figure 4-1. Flow chart of the main function

The function ‘initiate_parameters’ defines the main parameters to use in the algorithm as the number of requested packets from each node the phi and psi ratios. It let us introduce the grade of generosity we wish or the energy constraint for each node.

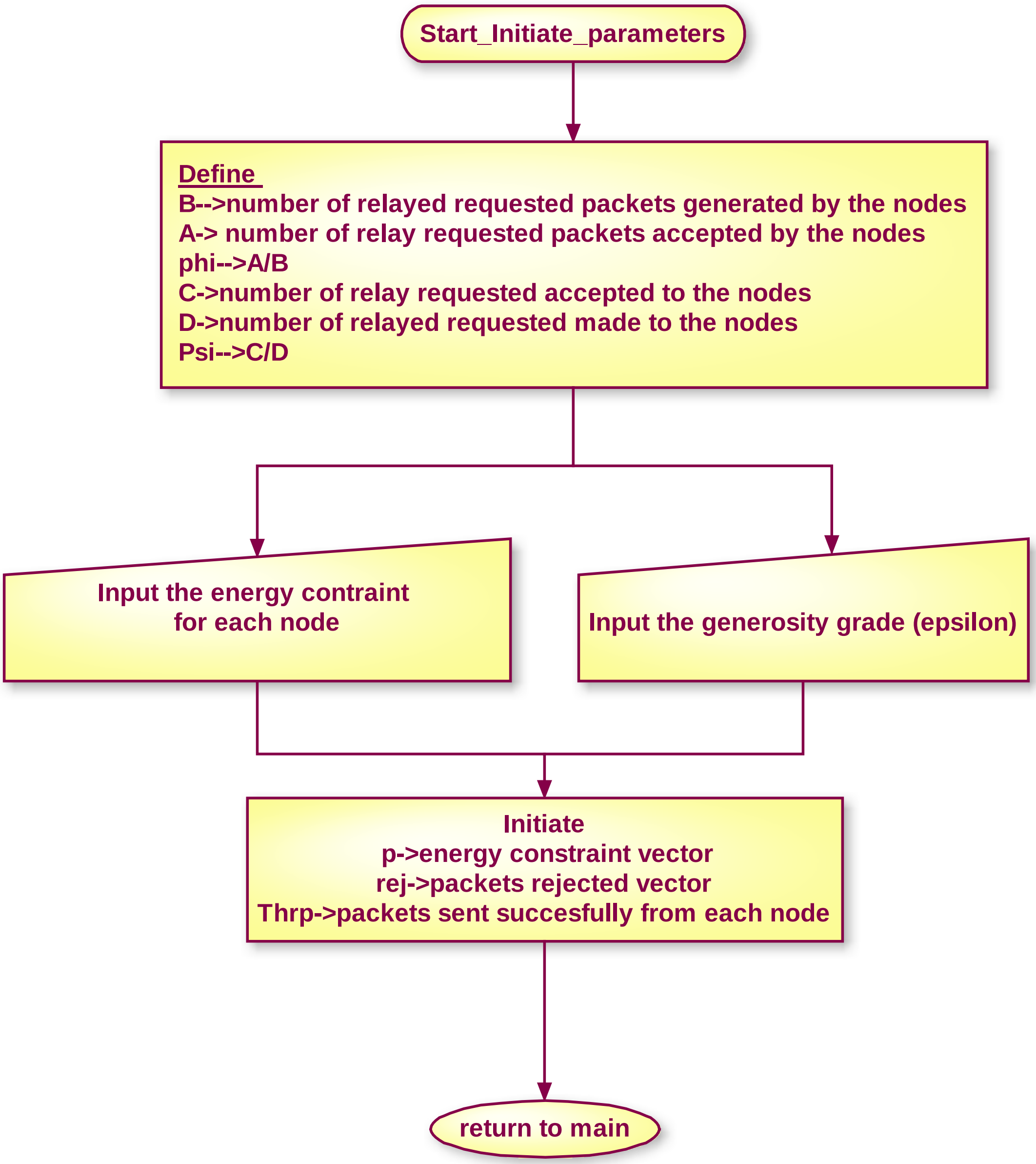
Start_Initiate_parameters
Define
B-->number of relayed requested packets generated by the nodes
A-> number of relay requested packets accepted by the nodes
phi-->A/B
C->number of relay requested accepted to the nodes
D->number of relayed requested made to the nodes
Psi-->C/D
Input the energy contraint for each node
Input the generosity grade (epsilon)
Initiate
p->energy constraint vector
rej->packets rejected vector
Thrp->packets sent succesfully from each node
return to main

Figure 4-2. Flow chart of initiate_parameters

The function ‘find_source’ let us find which is the node what is sending the packet. We can do it by comparing the main time vector with the single vector of each node. Depending on which is the source we will call a subroutine called ‘source_1’ if the source is the node one…to not repeat four times the same chart we will show the generic one.

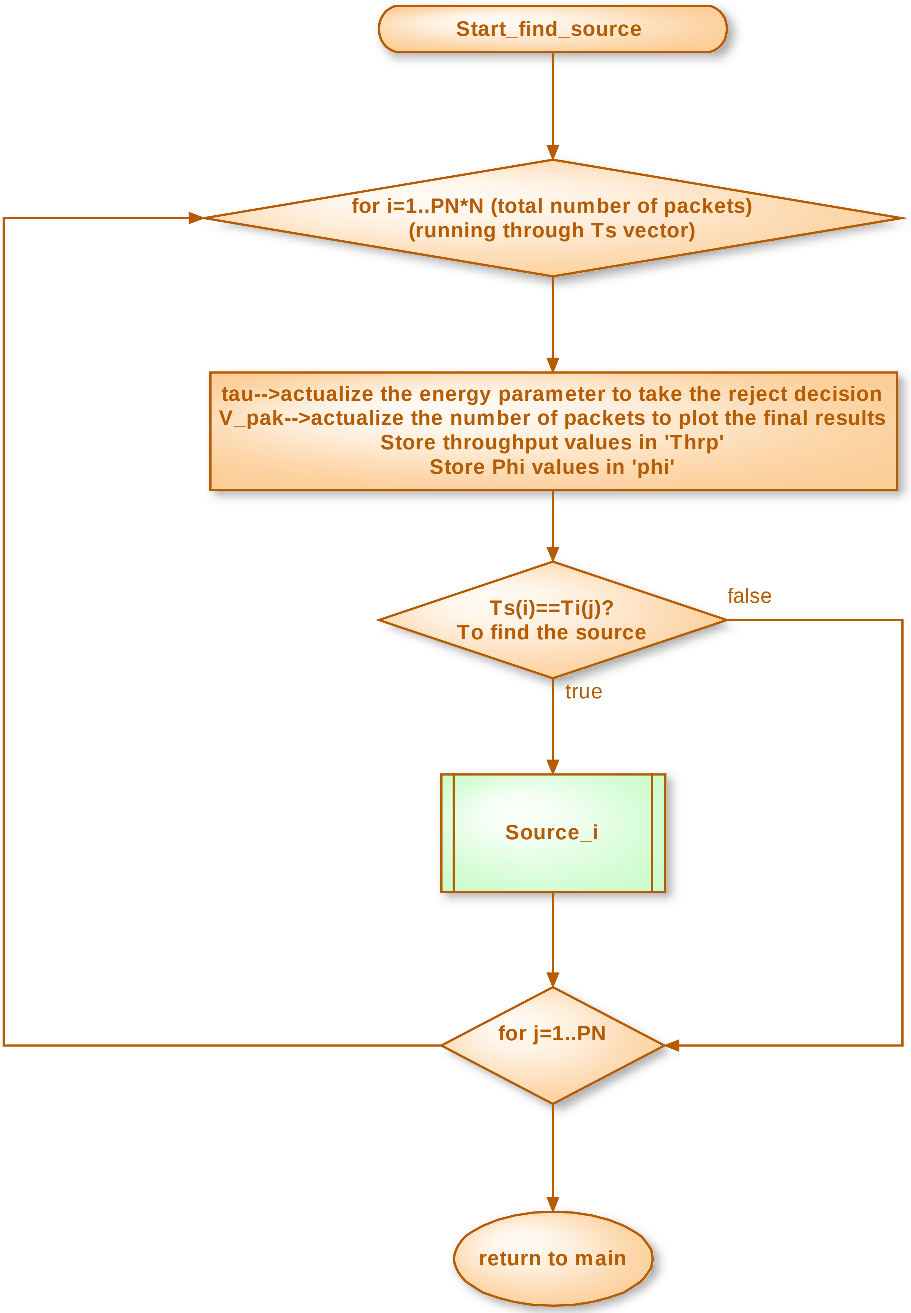
Start_find_source
for i=1..PN*N (total number of packets)
(running through Ts vector)
tau-->actualize the energy parameter to take the reject decision
V_pak-->actualize the number of packets to plot the final results
Store throughput values in 'Thrp'
Store Phi values in 'phi'
Ts(i)==Ti(j)?
To find the source
false
true
Source_i
for j=1..PN
return to main

Figure 4-3. Flow chart of find_source

The ‘source_i’ function is which either rejects or accepts the request from the node ‘i’. First of all it chooses randomly the way to reach the destination, and then it will check the condition parameters to take the decision.

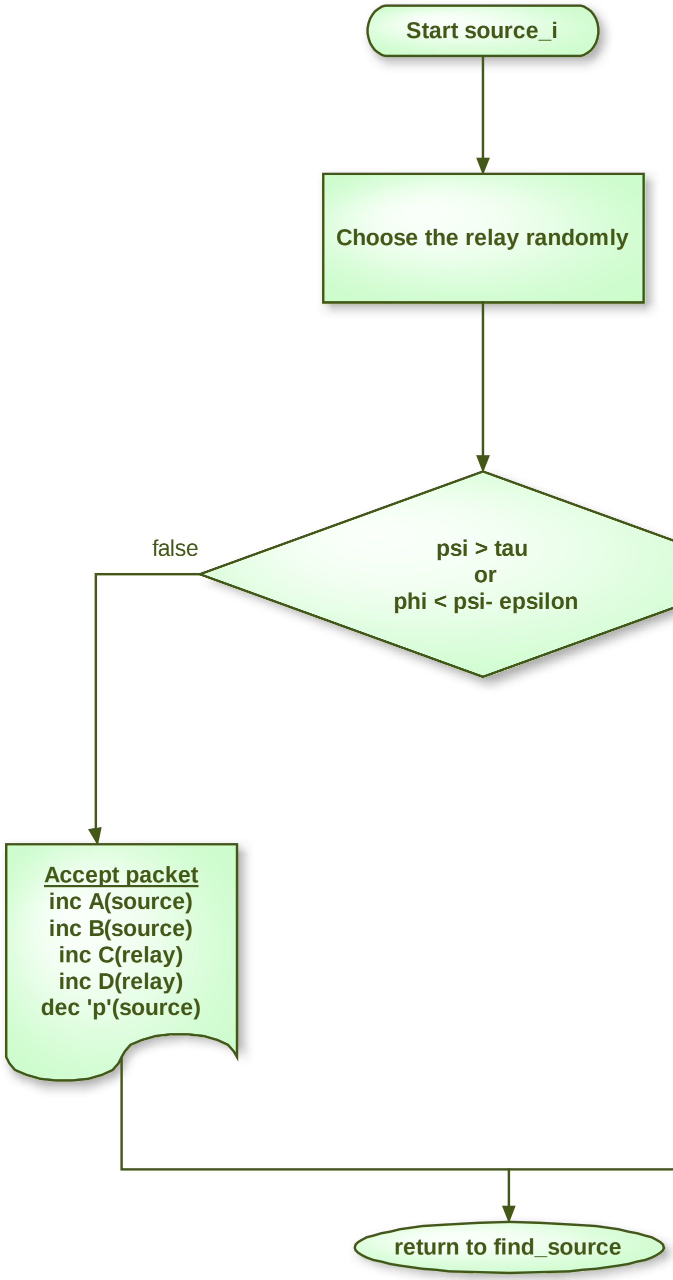

Figure 4-4. Flow chart of ‘source_i’ function

5 SIMULATIONS

SIMULATIONS RESULTS

In case 1, we are going to study the speed saturation and the maximum throughput the nodes can reach. The following case will study the dependency between the final throughput and the energy constraint for every node. In Case 3, we will observe the throughput improvement by incrementing the grade of generosity. Finally, case 4 will show how is the network behavior when nodes have internal and external traffic to treat.

In this section, we investigate the behavior of the GTFT. We focus on the single relay case. We consider a system with four nodes in it (N = 4). The energy constraints are given by ρ_1 , ρ_2, ρ_3 and ρ_4. Also, we asume M = 1 (numbers of relays), i.e., the route between the source and the destination node includes exactly one relay.

We study convergence of the proposed strategy by assuming that all nodes employ GTFT as their acceptance algorithm. The results show that the NAR values increase as far as ε (the grade of generosity) is higher. For the sake of simplicity, in the plot, the evolution of the NARs is shown for just one node per each session type.

5.1 CASE 1. SPEED SATURATION AND MAXIMUM THROUGHPUT

In this case we want to study the general network behavior. Every time a node either relays or sends traffic in the net, we are assigning a value given in tan percent. This value tells us how much energy is consuming each node per packet treated.

By changing the energy consumed by every node we will be able to elaborate a function that shows the maximum rate we can reach in our system and when it gets saturated. This information will be so useful to compare with further cases because it will provide us a point of reference.

In this case we simulate the network with all the nodes fully charged, that is they have all the energy to spend in the session (ρ_i=1). Every node has 1.000 packets to send, in other words, every node will request a relay a thousand times. The power consumption per node is given by ‘c’.

First of all, we are going to explain the main issues about a simulation by creating one with the following parameters: ρ_i=1 (*fully charge for every node*); c=0.0008(*0.08% energy consumption, every packet treated per node*) and ε=0 (*no grade of generosity*).

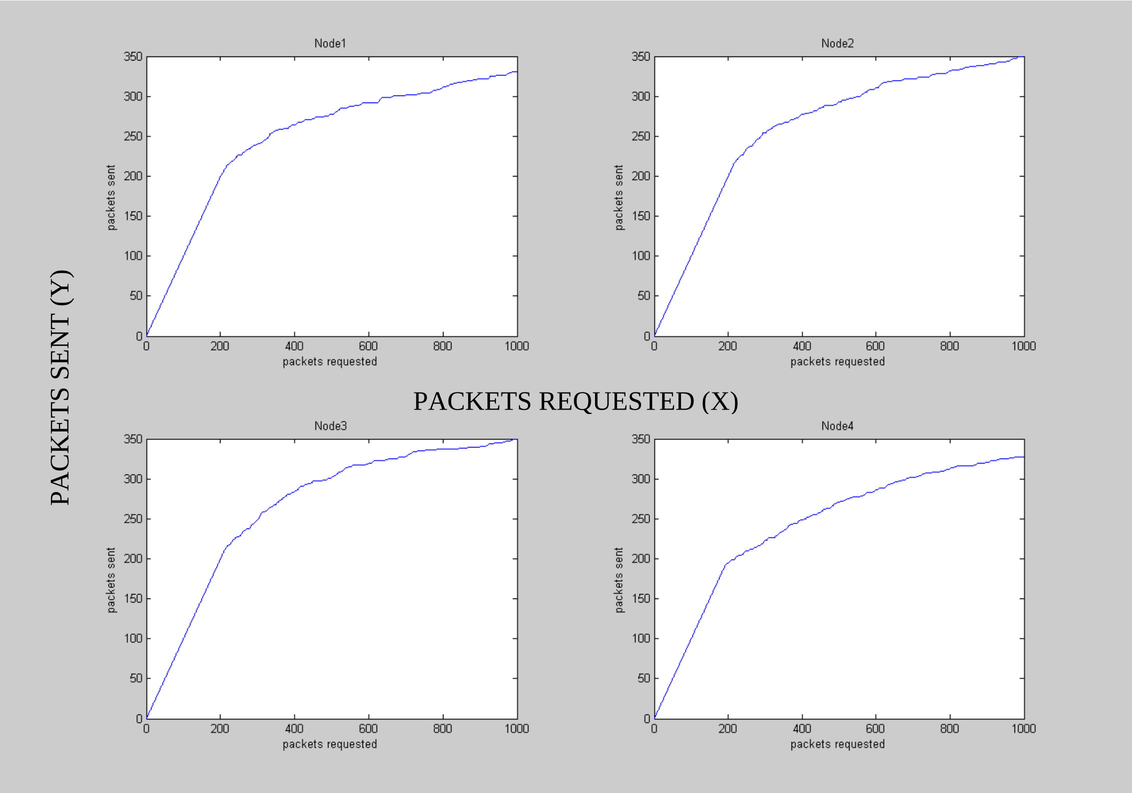

Figure 5-1 Nodes throughput, case 1. Number of packets=1.000; c=0.0008; ρ_i=1; ε=0

As we can see in the *figure 5-1* above, we have to notice that the throughput increases its rate until reaching the saturation. This point is logically explainable because ρ decreases its value meanwhile the nodes are sending traffic packets. This happens until the nodes begin to refuse the requests from their neighbors due the energy limitations.

The next *figure 5-2* shows how the ratio packets sent / packets requested evolves depending on the packets requested. As we have explained before we call this ratio ϕ.

We can notice that the ratio holds on 100% the first packets treated in the net. It occurs until the energy constraint begins to be insufficient, then the ratio falls and it is when the nodes begin to refuse relay petitions from their neighbors.

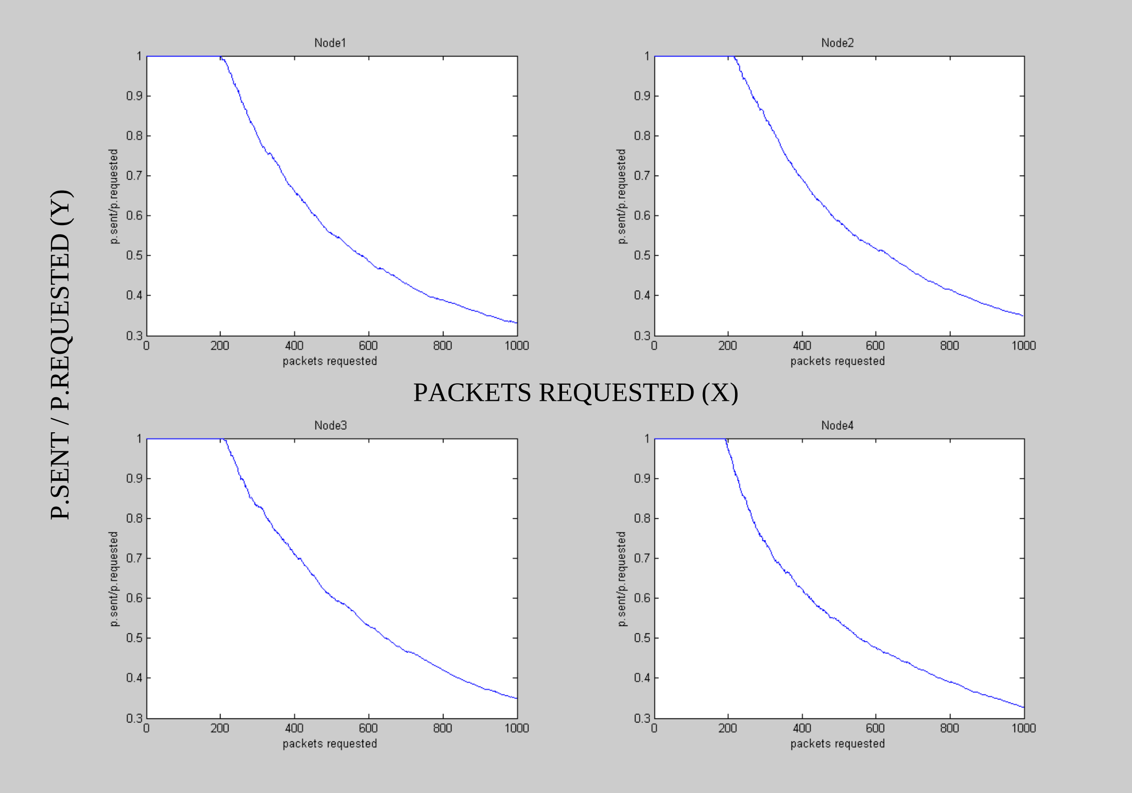

Figure 5-2 Throughput ratio nodes, case 1. Number of packets=1.000; c=0.0008; ρ_i=1; ε=0

The *figure 5-3*, is showing how is the behavior of the system embedded. We can see the added throughput and the ϕ ratio adding the parameters from the four nodes of the net seen before.

In the graphic of the ratio, we have put a line to recognize the $1/e$ level. That will help us to recognize the slump point (SP). This point we have called 'slump point' defines the amount of packets needed to saturate the network. In other words, we take this point when the ratio works below 37% to find out the speed saturation for every further simulation.

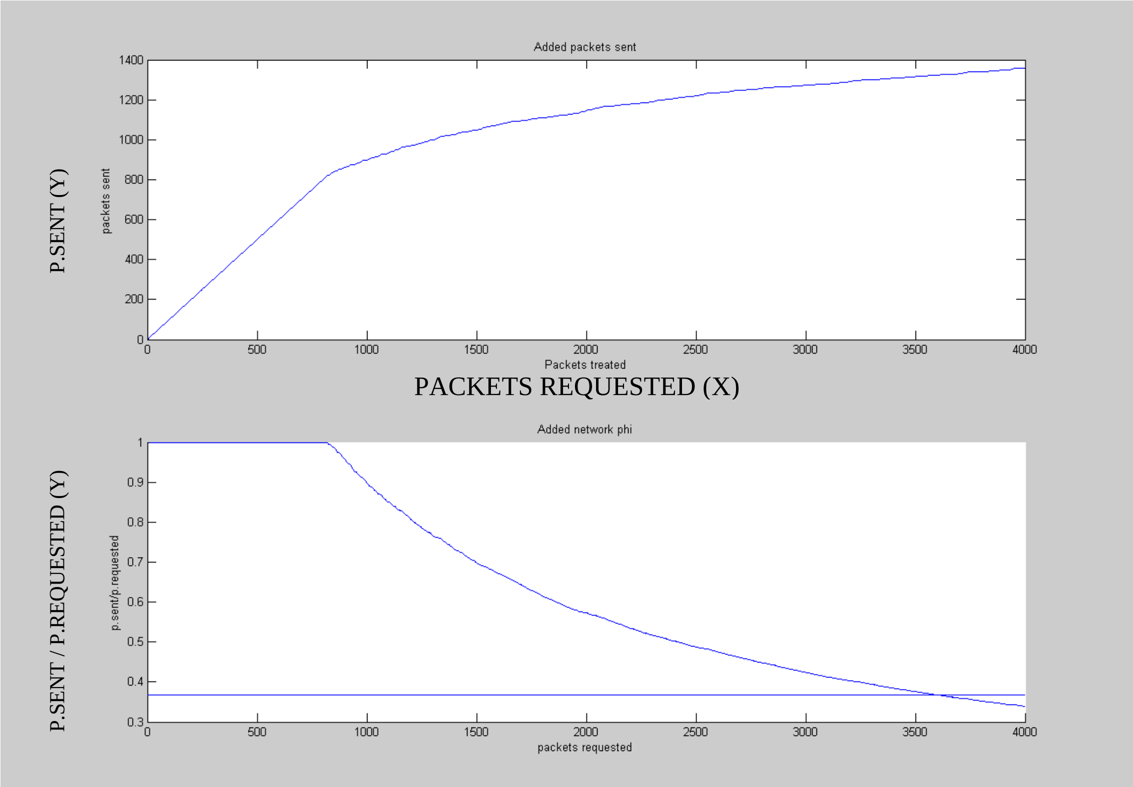

Figure 5-3 Added behavior, case 1. Number of packets=1.000; c=0.0008; ρ_i=1; ε=0

Playing with the consumption parameter, we can make a study to see the Maximum rate we can reach in our model depending on the energy consumption of each node. This let us know how is the net behavior when there is a waste of energy.

Once we have done the simulations, by calling function ‘casezero’ we create a new graphic. Looking at the *figure 5-3*, we figure out logically the Maximum rate decreases as the energy consumption increases its value. We have simulated the net with energy consumption since 0 until 0.001 (a 0.1% of energy consumption per packet treated). It’s showed that the Maximum rate depends on the energy consumption in a logarithmic way.

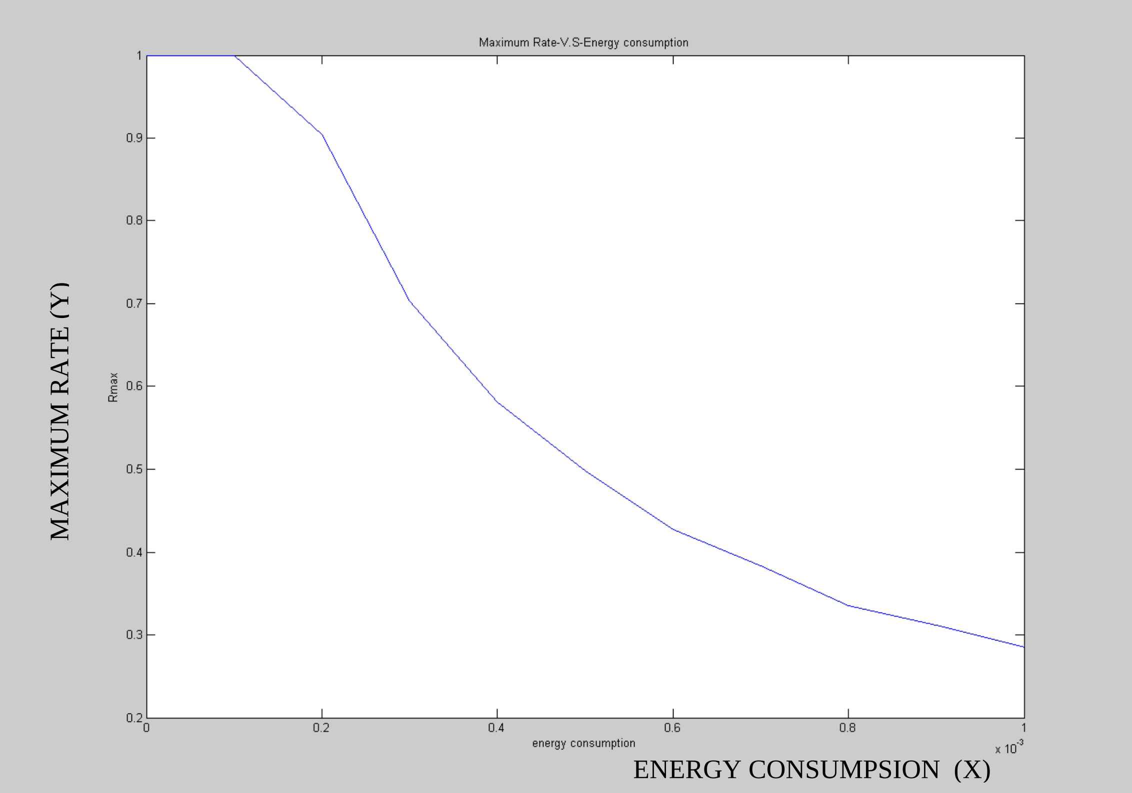

Figure 5-4 Maximum Rate depending on the energy consumption

5.2 CASE 2.THROUGHPUT AND ENERGY CONSTRAINT DEPENDENCE

In our network model, we can set the energy constraint (ρ) for every node. That let us study the behavior of the added net given parameter ρ. By simulating varied scenarios with different ρ we can estimate the throughput behavior. That will be useful to understand the importance of the energy constraint for further simulations.

- The parameters set for these simulations will be:

Energy constraint	(ρ)	→ variable
Energy consumption	(c)	→0.0008 per packet
Packets requested		→1.000 packets
Grade of generosity	(ε)	→0

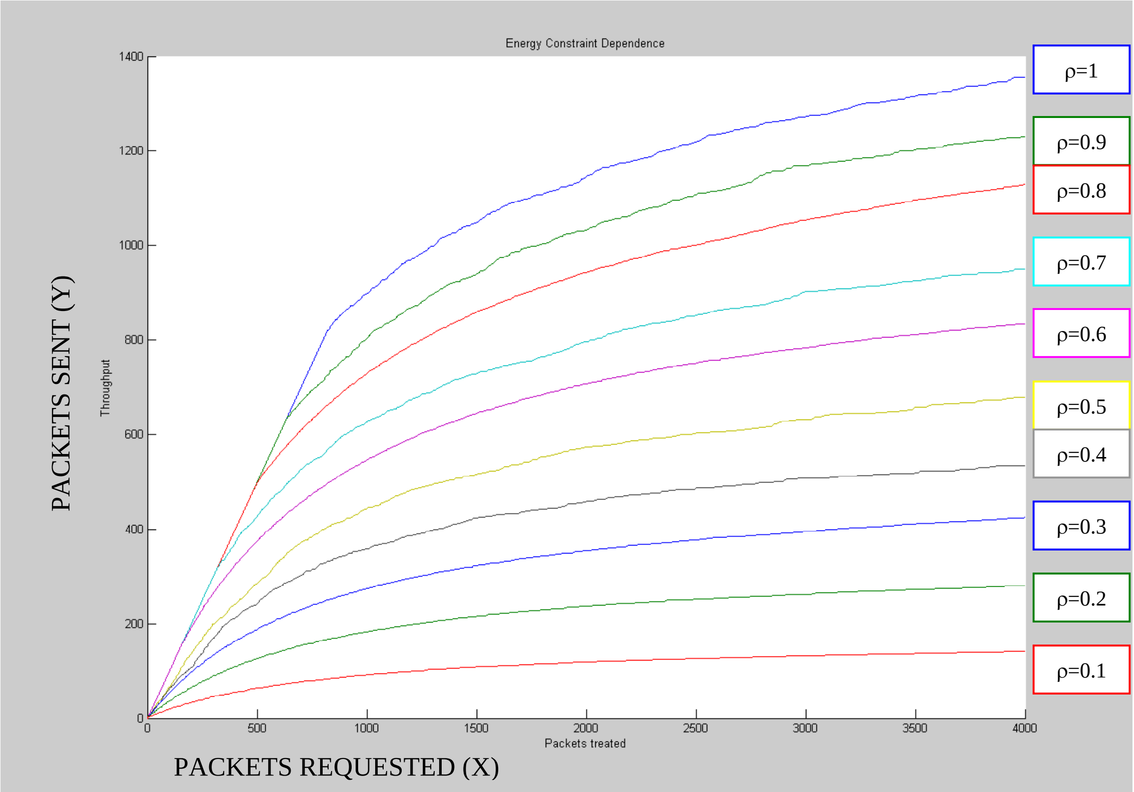

Figure 5-5 Throughput for different power constraint

From the *figure 5-5*, we can notice the way the throughput is restricted by the energy constraint. It's easily perceptible that the throughput depends on the energy constraint in a linear way. But in the next *figure 5-6* we can see it more obvious putting the throughput depending directly on the energy constraint.

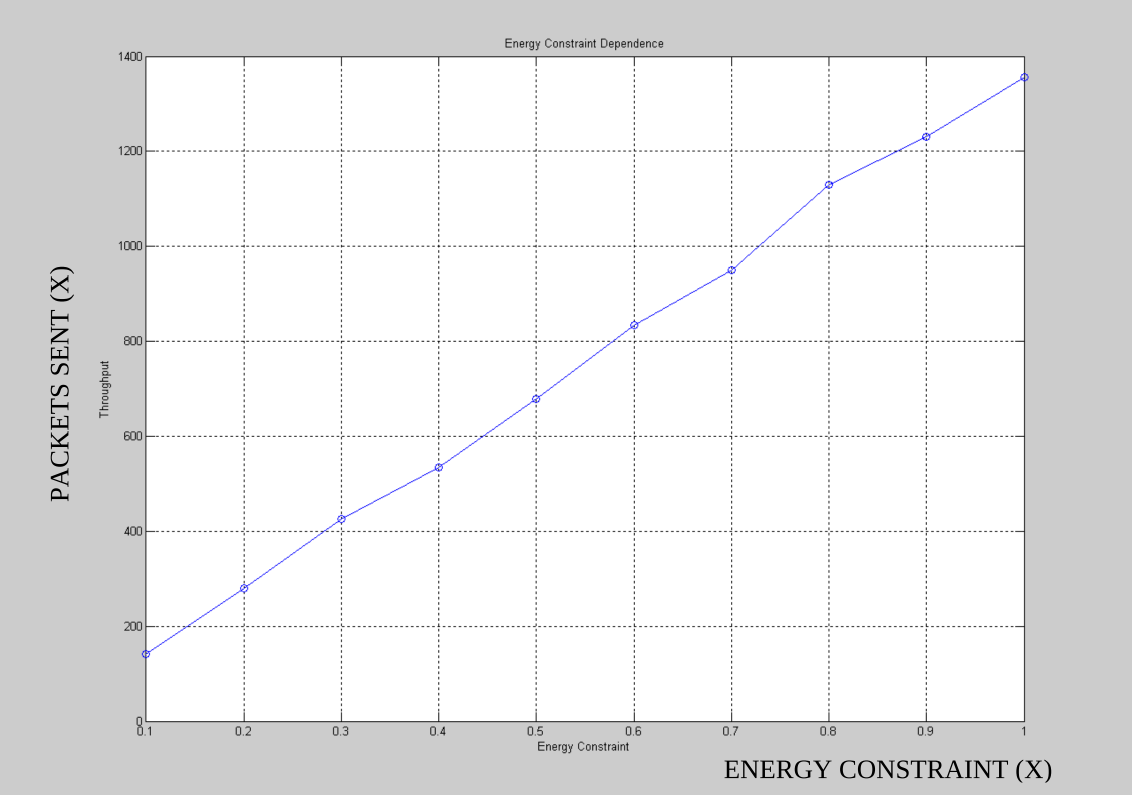

Figure 5-6 Throughput depending on the energy constraint

In other words, in our simulation, it's showed that the throughput increases its level as much as we increase the level of the energy constraint in the network. Compared to the case before, we observe a meaningful difference between the throughput growth and the energy constraint parameter, since in this case is linear. Besides the throughput growth depends on the consumption exponentially as we have seen in the previous case.

5.3 CASE 3. QUANTIFICATION OF THE THROUGHPUT INCREASE DEPENDING ON THE GRADE OF GENEROSITY

In this simulation, we want to study the throughput improvement because of the grade of generosity (GoG). This way we could observe the importance of the grade in our algorithm, GTFT. As the previous cases we will simulate different scenarios changing the grade and observing how is the throughput of the system improving.

To obtain a more realistic simulation this case is set with a different power constraint for each node. Thus, we will make the GoG more important since the rejections will come up because of the limited energy in our system.

The energy constraint set in this simulation will be $\rho_1 = 1$, $\rho_2 = 0.9$, $\rho_3 = 0.7$, $\rho_4 = 0.5$ with a number of request of 1000 packets each node as the previous simulations.

First we study the scenario without GoG to compare with the next scenario, which will be provided with a small increment on its GoG.

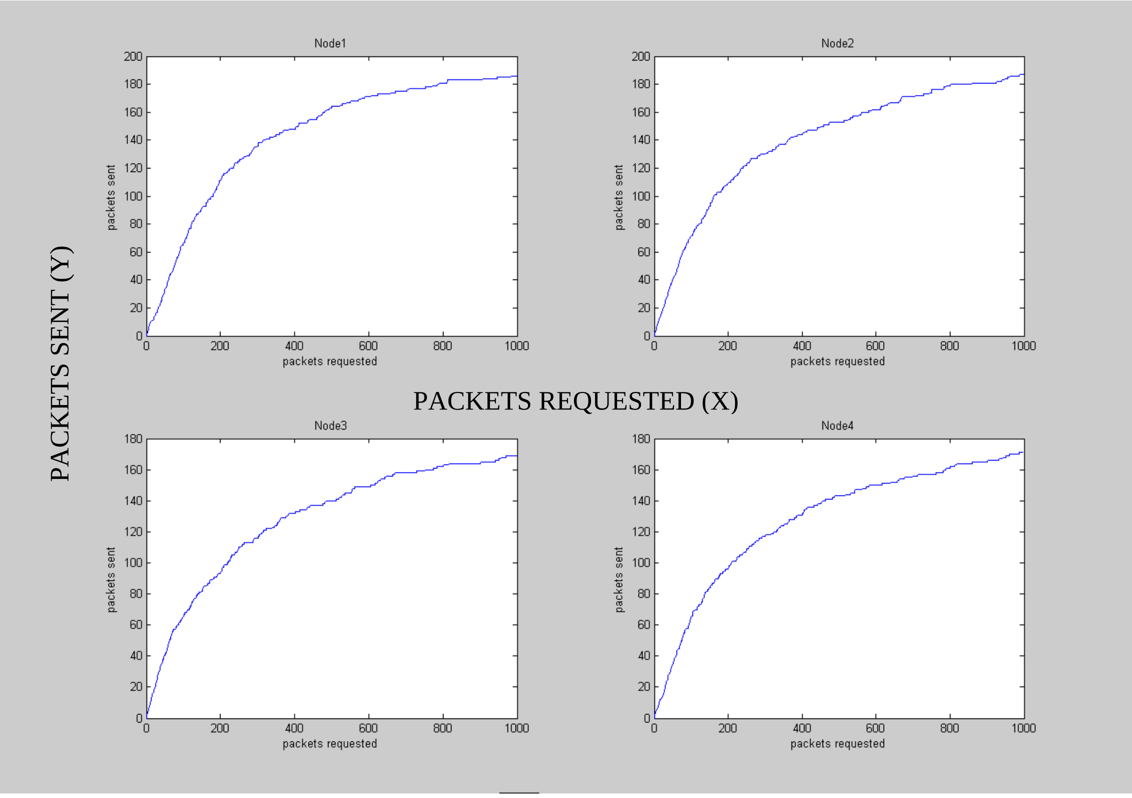

Figure 5-7 Node Throughputs. Simulation with no GoG. Number of packets=1.000; c=0.0008; ρ_1 =1, ρ_2 = 0.9, ρ_3 = 0.7, ρ_4 = 0.5; ε=0

The *figure 5-7* above shows us the throughput behavior from each node. We can notice that the final throughput of every node converges approximately to the same number of packets. This phenomenon occurs since the algorithm considers the energy constraint of the relaying node to either reject or accept the packet from the source node. This means that a node with a high power constraint will send less traffic due the power constraint of its neighbors.

In a long term the nodes will converge its throughput to the same value given that they are relaying the traffic considering their self-interest. We will call this phenomenon the ‘compensating phenomenon’.

The compensating phenomenon occurs because of the rejection condition related to the selfishness of the nodes. Given that one node never relays traffic when it have relayed more packets than the others from him, hence it won’t rely the packet. It’s like saying: ’I don’t help you because you haven’t helped me enough’.

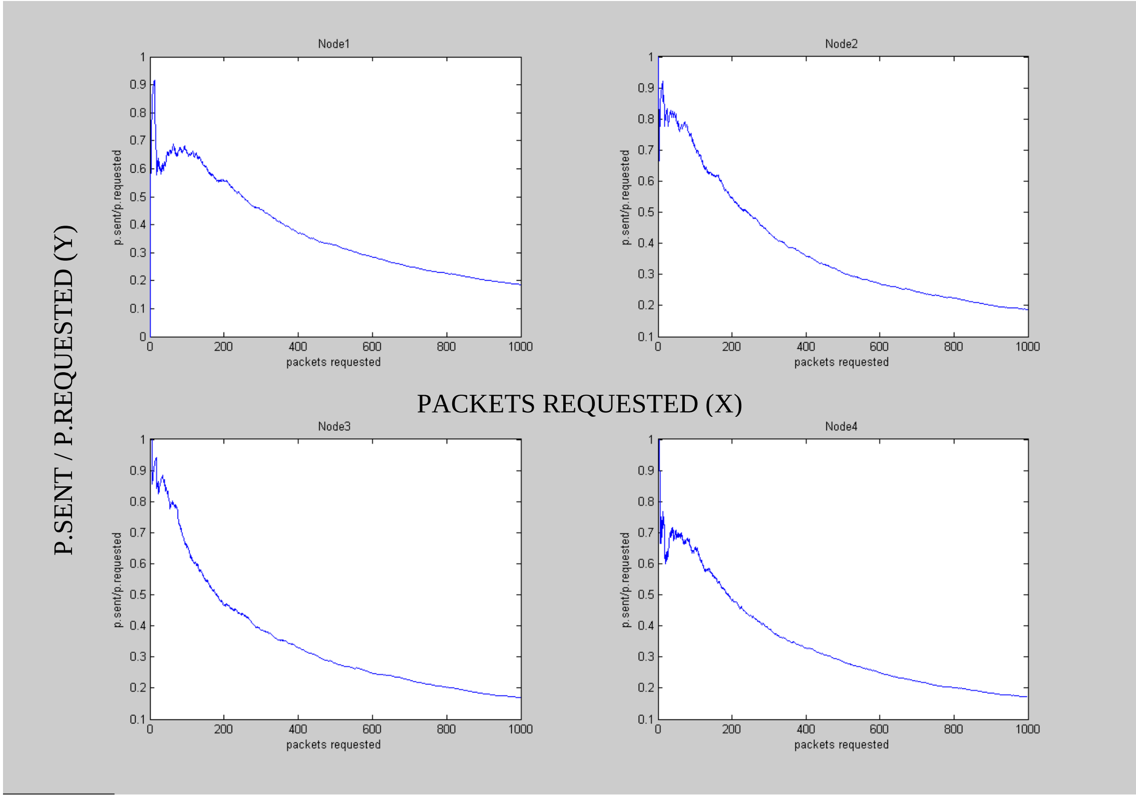

Figure 5-8 Throughput ratio nodes, case 3. Number of packets=1.000; c=0.0008; ρ_1 =1, ρ_2 = 0.9, ρ_3=0.7, ρ_4 = 0.5; ε=0.

The *figure 5-8* above shows us how is the ϕ ratio evolving. As we commented before the value where converges is the same for every node because of the compensating phenomenon.

As we can spot from the graphics, there are some irregularities at the beginning of the simulation. They are attributable to the compensating phenomenon. At the beginning the nodes have different power constraint and because of the algorithm they can reject a packet either they have relayed more traffic compared with their energy ($\psi_h > \tau_h$) or they have relied more packets than have been sent ($\phi_h < \psi_h$).

This fact makes the system compensate the energy for every node until they are balanced. Remembering that a node could reject for two reasons: the first one is related to the energy constraint, and the second is related to a self-interest issue (if the node that have been requested as a relay has sent more traffic from other than its own, then the node rejects the petition).

The straight zone is caused by the increase of the rejections, making slow down the throughput (numerator) meanwhile the node continues requesting to other nodes (denominator). This fact makes the curls less intensive since the throughput magnitude is significantly smaller than the requests.

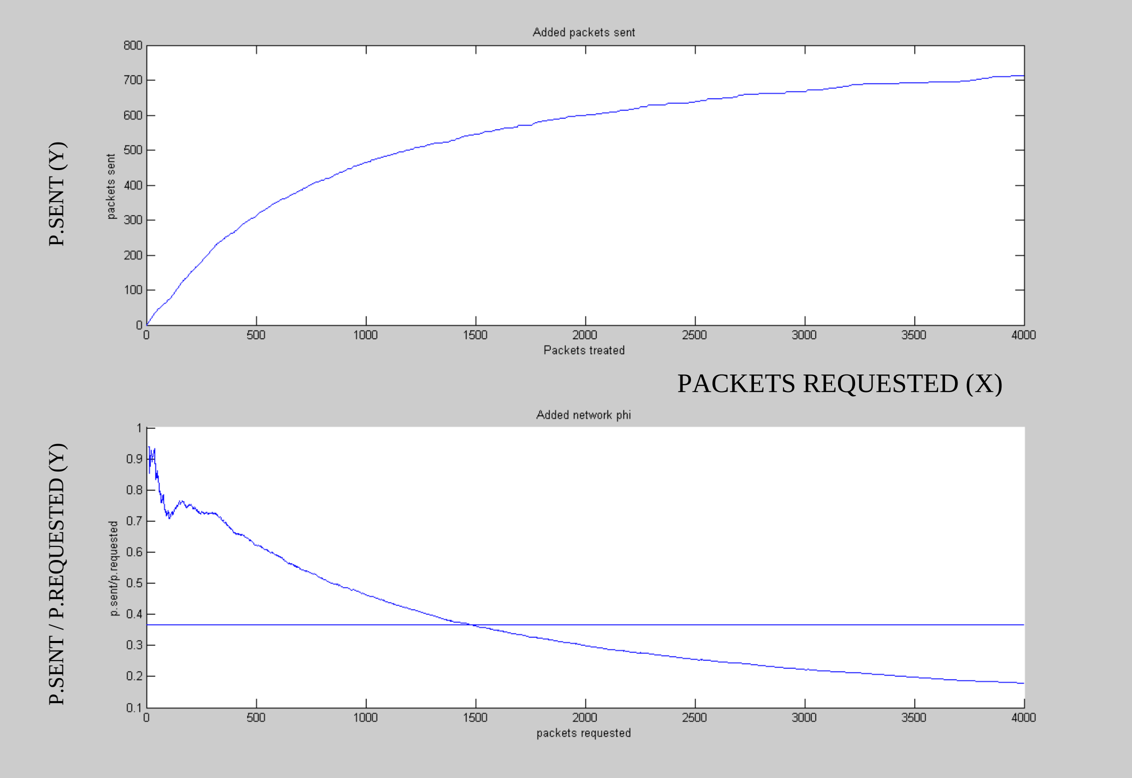

Figure 5-9 Added behavior, case 3. Number of packets=1.000; c=0.0008; ρ_1 =1, ρ_2 = 0.9, ρ_3 = 0.7, ρ_4=0.5; ε=0

As we can observe at the *figure 5-9*, we have the added behavior of the network. Now we want to study the importance of the GoG by looking at the slump point. In our case are 1474 packets to reach the slump point without GoG. For further simulations we can compare this number to confirm the throughput improvement in the network with the same parameters set.

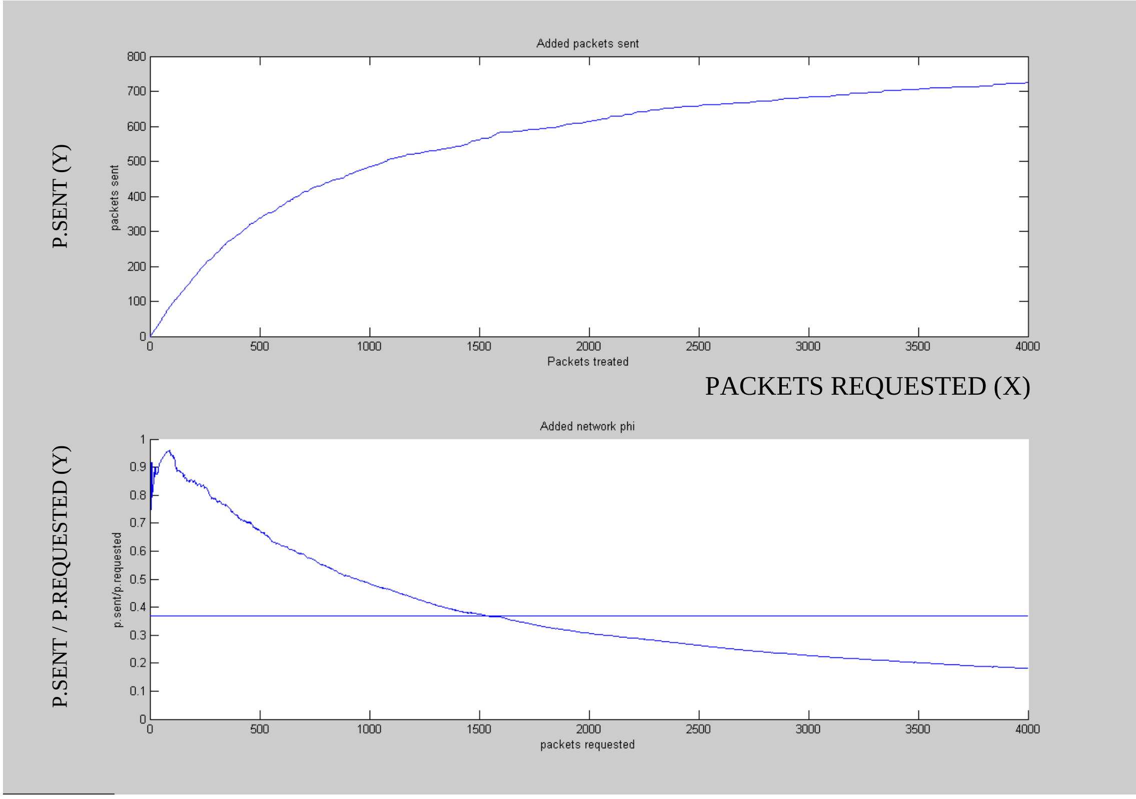

Figure 5-10 Added behavior with grade of generosity, case 3. Number of packets=1.000; c=0.0008; ρ_1 =1, ρ_2 = 0.9, ρ_3 = 0.7, ρ_4 = 0.5; ε=0.01

The *figure 5-10* above shows the behavior of the same scenario but with a GoG=0.01 we can observe the improve since the slump point is establish at 1539 packets, that is 65 packets later than before. In other words, we have hold up the throughput more time than the case before by adding a small grade of generosity for each node. The logical explanation is that the nodes have rejected fewer packets caused by their self-interest making improve the system.

Now, our algorithm to refuse a petition has change, $\phi_h < \psi_h - \varepsilon$, instead of only $\phi_h < \psi_h$, this means that the nodes are less selfishness in view of the fact that they are thinking like before 'I don't help you, if you don't help me' but ε tell us that the other nodes have had to help less than the rely to make it help them.

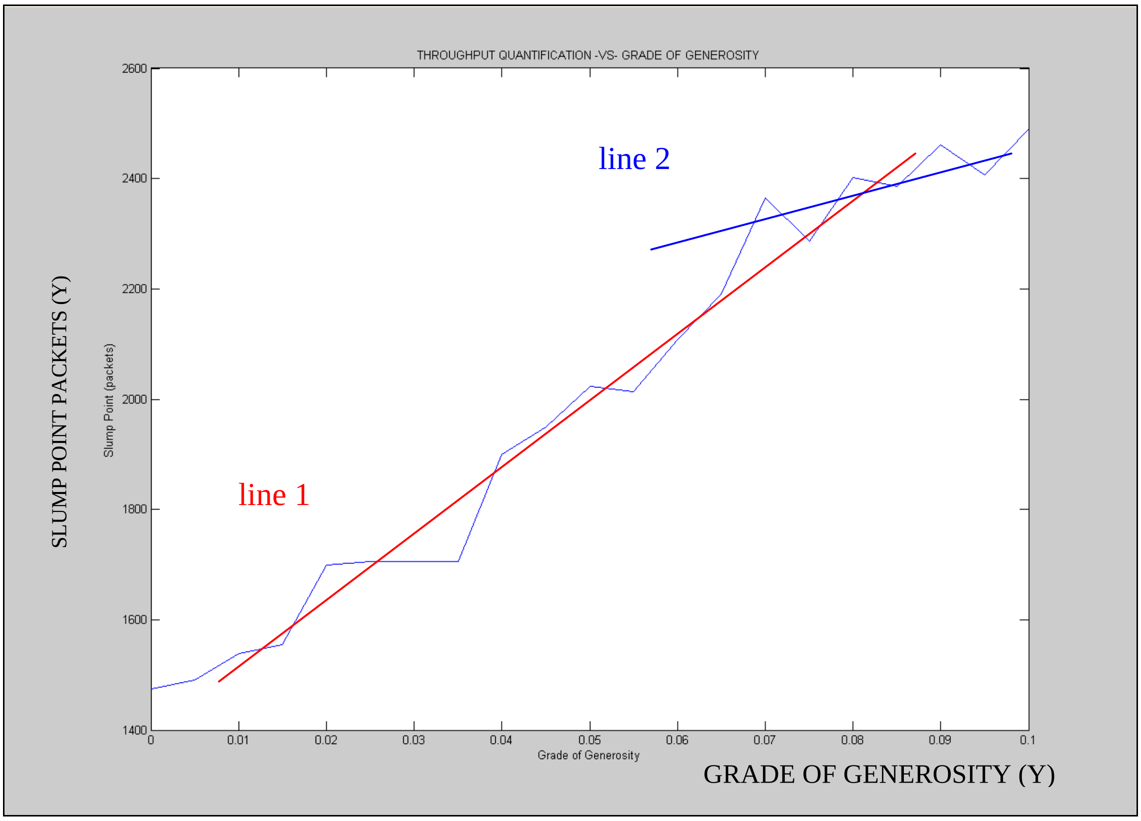

Figure 5-11 Slump Point depending on the grade of generosity

The *figure 5-11* shows how the slump point is delayed as a result of an increase of the GoG. We can observe that the tendency of this function is linear (line 1). This means that the number of packets to reach a throughput rate below 37% increases as long as we enlarge the GoG making a network more efficient. Each node has sent more packets than before with the same power constraint. We can observe that the tendency changes its slope due to this power constraint; it's lower (line 2) . In other words, the power constraint is what sets the ceiling of the throughput, hence the slump point.

5.4 CASE 4. EXTERNAL TRAFFIC NETWORK SIMULATION

Now we want to study the behavior of the system when a node creates packets to send outside the network. We assume that every packet sent out is relayed by one node (it’s the same structure network simulated before). In other words, before the packet reaches the RCST, it will be relayed before being transmitted to the satellite.

Assuming an integrated system where exists a RCSTs , we could develop a network system where every node (terminal) is capable to link with the satellite through different relays. This relays are the nodes of the AHN we want to simulate in this paper. Our job will be studying the nodes behavior to improve the global system.

We choose the free capacity assignment (FCA), as a capacity request category in this simulation. That means that the assignment shall be automatic and shall not involve any signaling from the DVB-RCS terminal to the NCC.

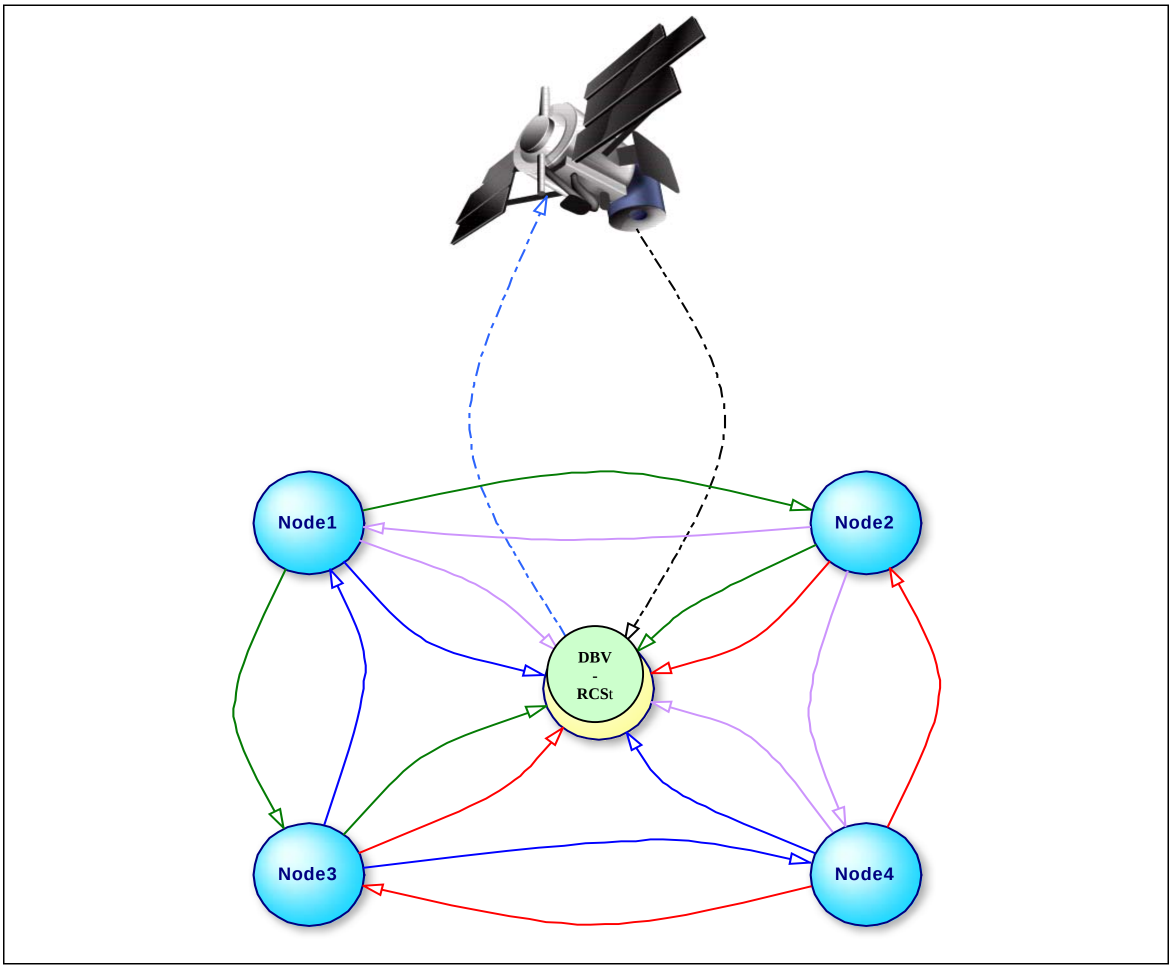

Figure 5-12 External traffic paths

In the simulation we will create external and internal traffic for each node. Being precisely, 50% of the packets will be external. The *figure 5-12* shows us the hops for the packets before reach the satellite, distinguishing a color for each node until it arrives to the RCST.

A relevant issue is ‘when are created the external packets?’ since one packet is produced at the end of the simulation it has more probabilities of being rejected because of the energy constraint boundaries. So our external packets are normally distributed in time. This way we can study the simulations with fewer distortions.

The simulations are set with the nodes fully charged, a power consumption of 0.008 for each packet and a grade of generosity of 0.05. We are going to spot the figures one by one to compare the throughput from the external traffic and compare it with the added traffic behavior.

In the *figure 5-13* we can observe the added throughput from each node. We can observe, with the characteristics of the scenario commented before, all the nodes converge around 35% of the packets requested are sent.

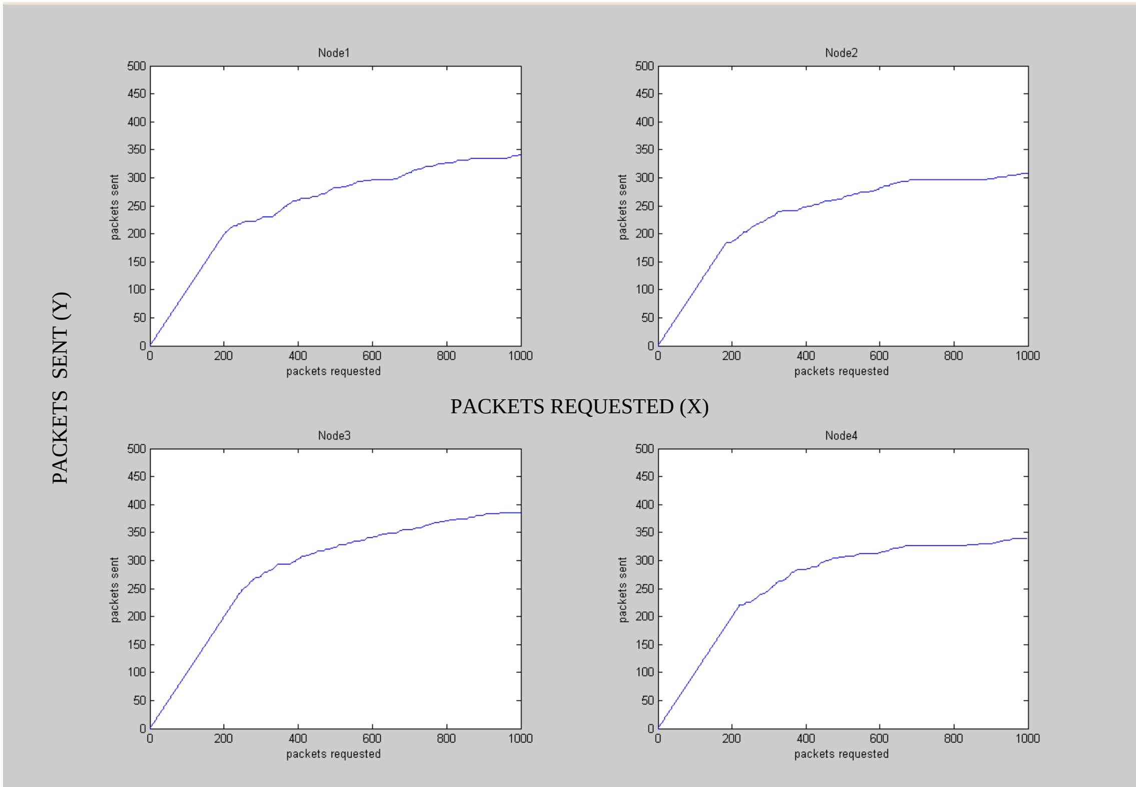

Figure 5-13 Throughput nodes, case 4. Number of packets=1.000; c=0.0008; ρ_i = 1; ε=0.05

In the next *figure 5-14*, not surprisingly, we notice that the nodes converge around the 32%. That is because the network is working with the same structure and a external traffic of 50 %. The packets have to jump one relay as the previous cases.

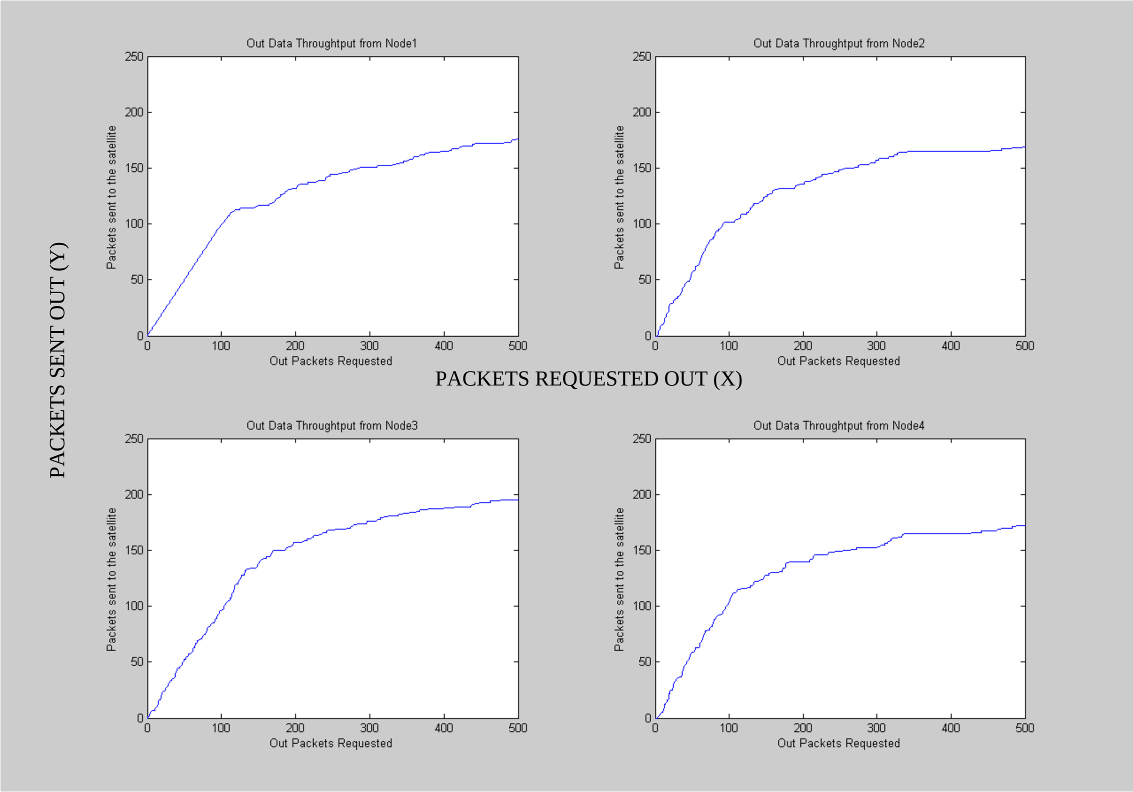

Figure 5-14 External throughput nodes, case4. Number of packets=1.000; c=0.0008; ρ_i =1,; ε=0.05

The interesting thing in this simulation it’s how is the behavior of the packets sent out divided by the packets requested out the network. That is showed in the *figure 5-15* below. The added traffic figure shows straight functions from every node. That is because the nodes are fully charged and the time of compensation for them are shorter since they start with the same energy constraint.

At the beginning, they all accept the entire requests until they are short of energy. Then the rejections are mostly because of the energy restriction. Their throughputs are compensated, so the reason to reject a source for being more helpful than the others is minimum. These facts make the fall in the throughput ratio straighter.

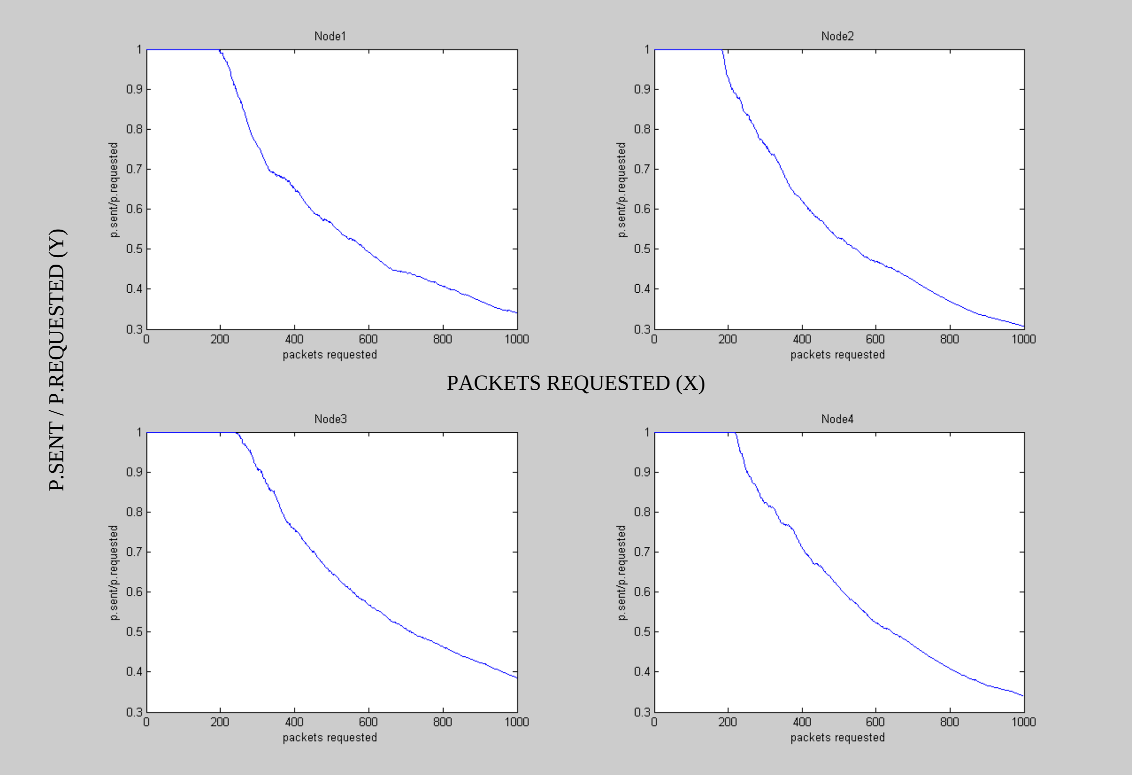

Figure 5-15 Throughput ratio nodes, case4. Number of packets=1.000; c=0.0008; ρ_i =1,; ε=0.05

On the contrary, the *figure 5-16* shows that the packets sent out divided by the packets requested out are choppier than the added traffic ones. The reason is caused by the time of requesting. Now, the packets taken in the functions are requested in a random distribution, this makes the ϕ ratio change suddenly in view of the fact that the rejection criterion is decided in different circumstances for every packets. So the compensation zone is choppier than before, which the requested packets were took in account consecutively.

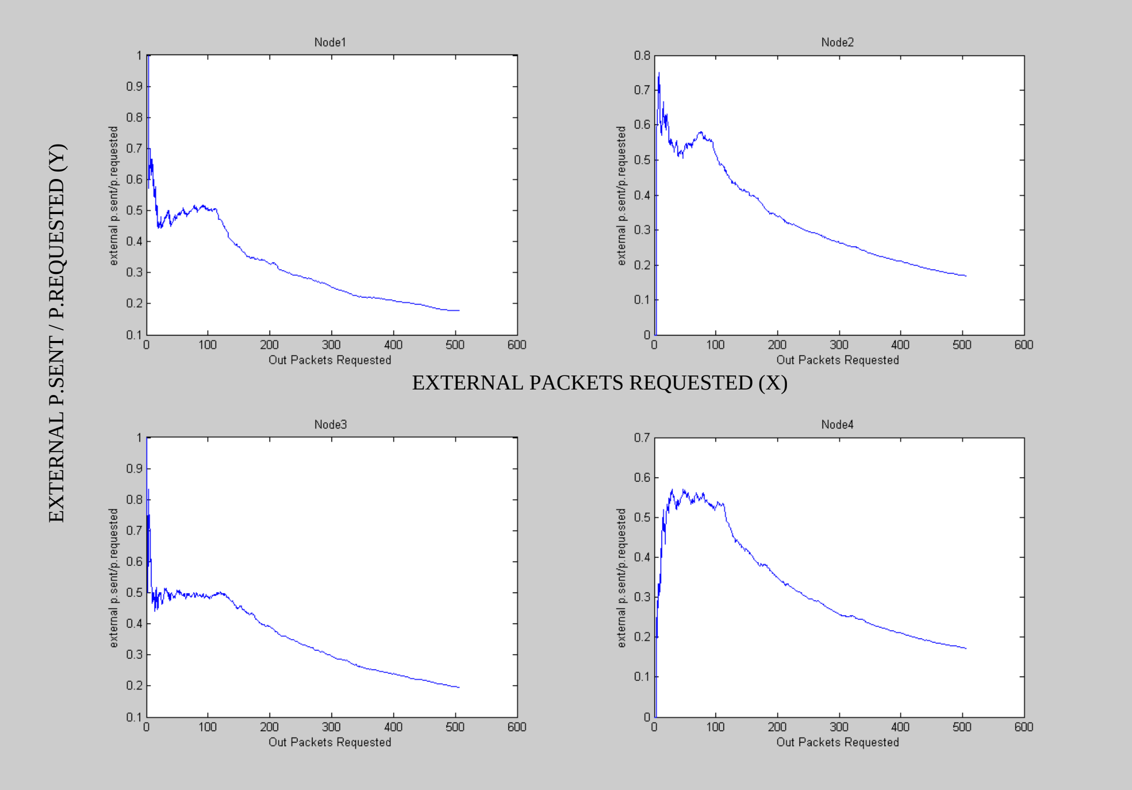

Figure 5-16 External throughput ratio nodes, case 4. Number of packets=1.000; c=0.0008; ρ_i =1,; ε=0.05

In the *figures 5-17* and *5-18* below, we can notice the behavior of the added throughput network. As we have revealed before, the final throughput is around 35% more or less like external throughput. The ϕ ratio throughput function is straight because the simulation starts with the nodes fully charged. This means that as long as the packets are being requested the rejection is for either the relay or energy constraint criterion.

In contrast, the added external ϕ throughput ratio is so irregular at the beginning because we are considering a random distribution of sent packets added. That makes that taking the decision to reject, when nodes are in a different state than the previous decision, the ratio varies excessively generating those curls in the graph.

The compensating phenomenon aims to create those alterations, too. This disturbance longs until the nodes stop of relaying traffic and then the ratio plunges.

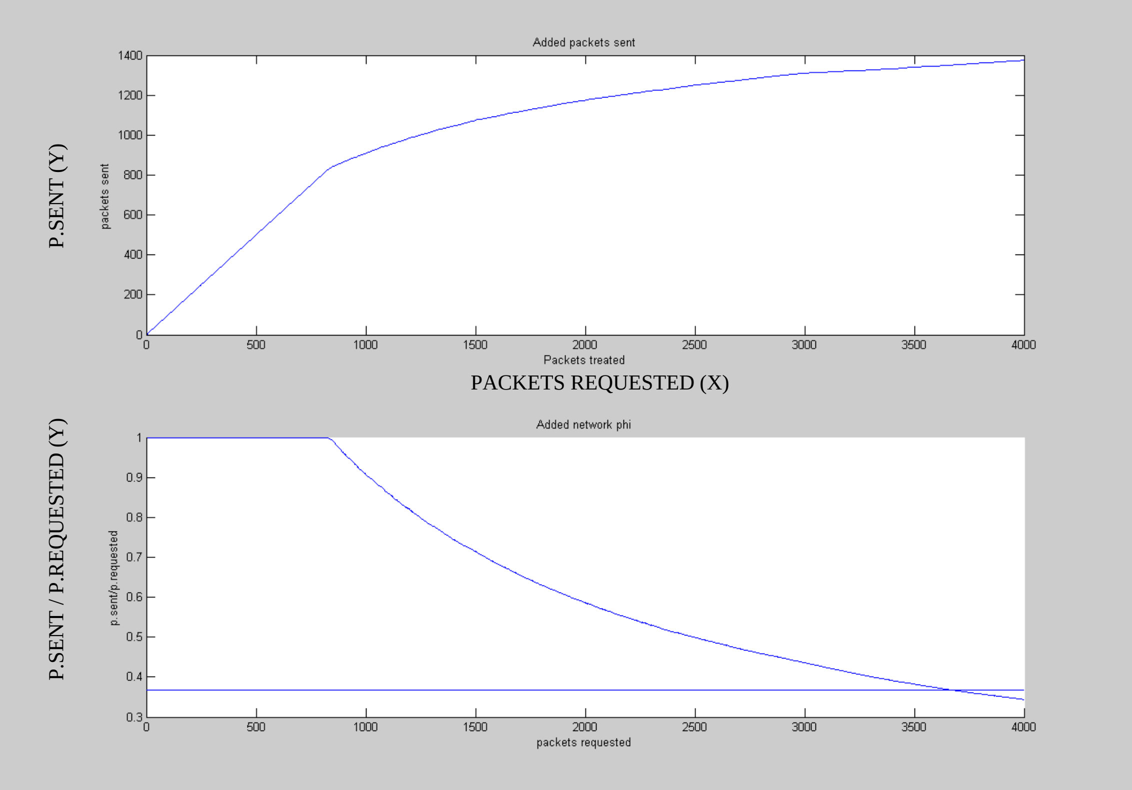

Figure 5-17 Added throughput case 4. Number of packets=1.000; c=0.0008; ρ_i =1,; ε=0.05

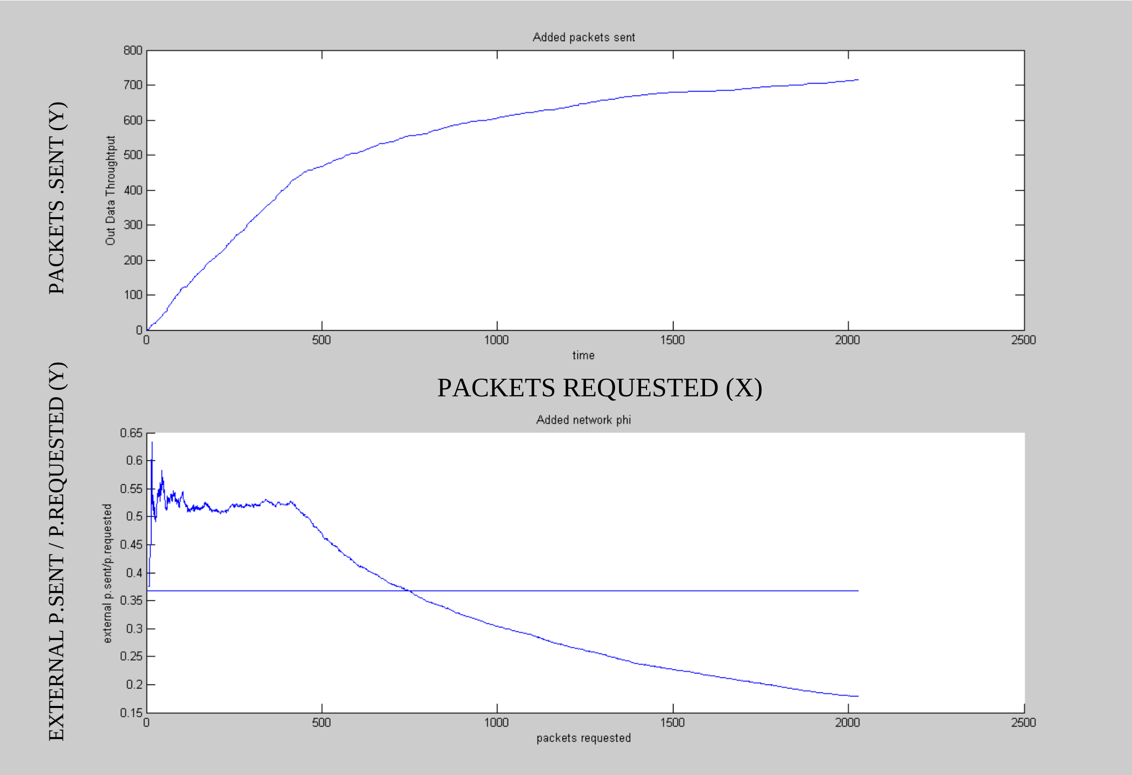

Figure 5-18 External Added throughput 50%, case 4. Number of packets=1.000; c=0.0008; ρ_i =1,; ε=0.05

The *figure 5-19* below shows the results from a 20% external traffic simulation. This helps us to contrast with the previous simulation (50% of external traffic). As we could forward before, the final efficiency is higher since the total traffic to rely is smaller. So we have affectivity around 68%.

Logically, with fewer packets to send outside, the system has more probabilities to increase its external throughput.

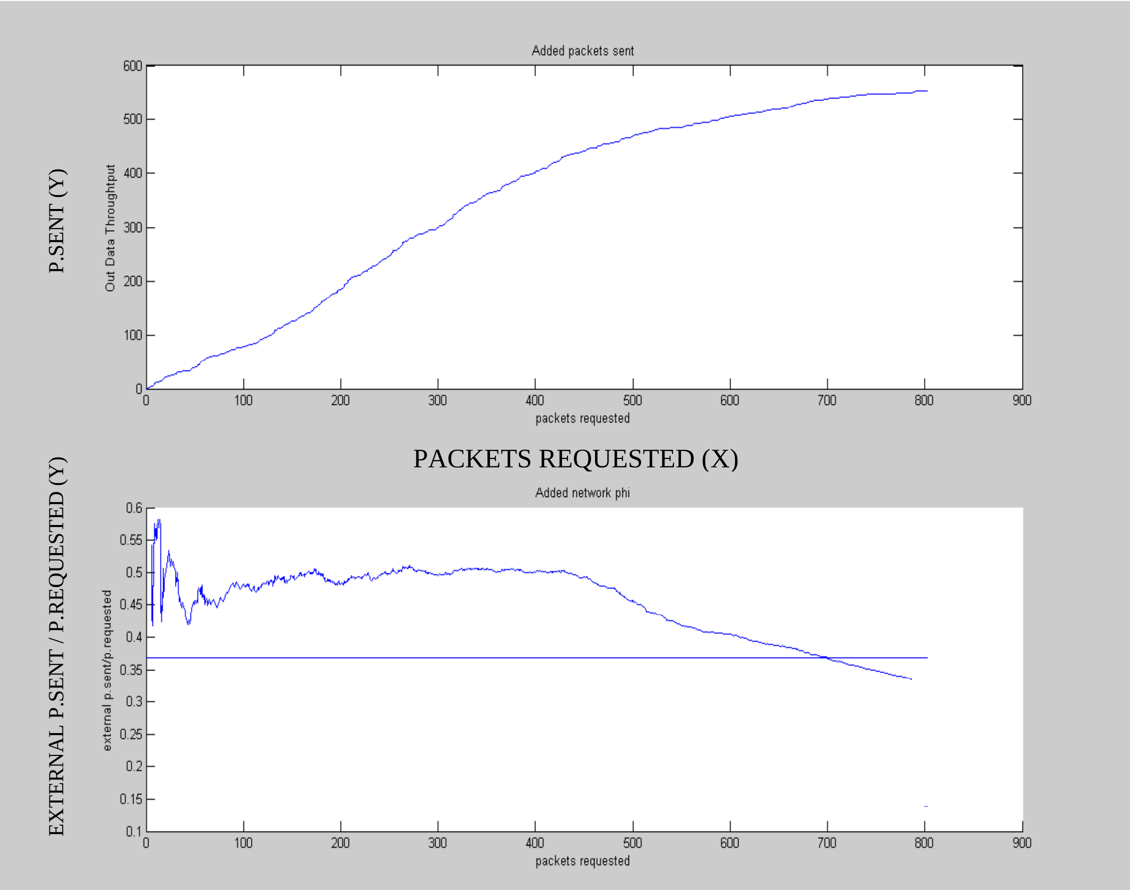

Figure 5-19 External added throughput 20%, case 4. Number of packets=1.000; c=0.0008; ρ_i =1,; ε=0.05.

As we can see looking at the previous simulation the ratio takes similar levels until reaches around 650 packets requested. So we can confirm the energy constraint is the important restricting fact at the end. The curls are attributable to the continuous rejected and accepted packets during the compensating phenomenon, and then the network loses its energy making the throughput to plunge.

6 DISCUSSION

6.1 CONCLUSIONS

Ad hoc networks hold the key to the future of wireless communication, promising adaptive connectivity without the need for expensive infrastructure. In ad hoc networks, the lack of centralized control implies that the behavior of individual users has a profound effect on network performance. For example, by choosing to leave a network or refusing to honor relay requests, a user can severely inhibit communication between other users. This is a severe contrast with fixed wireless systems where a single user has much less influence on other users. The influence of user behavior on network performance, in combination with the fact that nodes in an ad hoc network are constrained by their finite energy capacity, motivates the need for a rational and efficient resource allocation scheme. We addressed the problem of cooperation among energy-constrained nodes in wireless ad hoc networks. We assumed that users are rational and showed that as a consequence users will not always be willing to expend their energy resources to relay traffic generated by other users. By using elementary game theory, we were able to show the existence of an operating point that trades off throughput with lifetime. We also proved that these algorithms lead the system towards a better operating point by increasing a little bit the grade of generosity.

We use a simple model with four nodes as sources. A more complex model would make it harder to achieve analytical results. In ad hoc networks, game theory has been used to analyze the cooperation of the nodes. There exist various mechanisms designed to prevent selfishness and to enforce cooperation. In this paper we have studied the grade of generosity on the GTFT.

Our simulation model pretends to point out the importance and benefits of the GTFT algorithm that prevents selfishness in the network. Our approach determines that this way of acting for a node is good for the system sake. Hence, the added throughput in the network will be higher with the same power constraint. On the contrary, the network would saturate itself before.

CASE 1→ We have pointed out how the maximum rate responds to the energy consumed per packet. The dependence is exponentially decreasing as long as we increase the energy consume by the node in our algorithm.

CASE 2→ We explained how the throughput responds to the nodes power constraint. We noticed that this dependence is linear. That is, the throughput boosts on condition that we increase the total power constraint for the nodes.

CASE 3→ It's proved how the grade of generosity in our algorithm makes soar the added throughput in a linear way. The GoG makes drop off the rejections for selfishness reasons since the nodes can forgive a cheating node in previous requests.

CASE 4→ We have simulated the network with external traffic, which is treated for the relaying nodes alternatively with the internal traffic. That makes the throughput divided by the requests be unstable although the throughput reaches a stable response when the nodes begin to reject packets and the requests are higher.

6.2 FURTHER WORK

There is potential future work in both the game theoretic and simulation part of this paper. In the games, some restrictive assumptions were made. We studied only the behavior connection.

The algorithm has to be scalable. Since in practice, the user probably communicates with several nodes during the connection time instead only three nodes. In a more realistic model the terminal spends energy for requesting and rejecting, facts that we have omitted in this paper.

The external traffic in our network has been simulated in a very simple way. As we show in chapter 2, the connection to the satellite by the DVB-RCS protocol is more complicated than the simple petition we have assumed. For example, not all the traffic that arrives to the DVB-RCS terminal is sent forward to the satellite. Hence, it's important to see how the algorithm works over other standards and the different capacity request categories.

REFERENCES

[1] Tech FAQ, What is DVB-RCS, http://www.tech-faq.com/dvb-rcs.shtml

[2] ETSI EN 301 790, ‘Digital Video Broadcasting; Interaction channel for satellite distribution systems,’ v1.3.1, March 2003

[3] ETSI TR 101 790, ‘Digital Video broadcasting; Interaction channel for satellite distribution systems; Guidelines for the use of EN 301 790,’ v1.2.1, January 2003

[4] ESA ad-hoc RSAT WG, ‘Harmonization of terminals for regenerative satellite multimedia systems,’ Final report, January 2001

[5] ESA SatLabs group, ‘SatLabs system recommendation,’ v1.1, July 2003

[6] Cooperation in Wireless Ad-Hoc Networks (IEEE,2003). Vikram Srinivassan, Pavan Nuggehalli, CarlaF. Chiasserini, Ramesh R. Rao.[0-7803-7753-2/03]

[7] By Jim Geier, source from wi-fiplanet.com

[8] Mesh Networling from en.wikipedia.org/wiki/Mesh_network

[9] Performance issues of routing in ad hoc networks and ip over geo satellite links, Riadh DHAOU, Julien Fasson, Farid Jaddi, Mahamadou Issoufou Tiado IRIT Lab. INPT-ENSEEIHT – Toulouse, France

[10] Robert J. Shimonski: Windows Server 2003 Clustering & Load Balancing, Osborne McGraw-Hill, ISBN 0-07-222622-6

[11] Ph.D. Applications in game theory Ad-hoc networks, Juha Leino. Supervisor: Professor Jorma Virtamo, Samuli Aalto.

[12] State of the Art: Ad Hoc Networking. John Paul O Grady/ Aidan McDonald Adaptive Wireless Systems Group Cork Institute of Technology.

[13] S. Thomson and T.Narten 1998 “IPv6 Stateless Address Auto Configuration” (Draft Standard) 2462. Internet Engineering Task Force.

APPENDIX A: ACRONYMS AND ABBREVIATIONS

ABR	→ Available Bit Rate.
AHNs	→Ad-Hoc Networks
ATM	→ Asynchronous Transfer Mode.
AVBDC	→Absolute Volume Based Dynamic Capacity.
CBR	→ Constant Bit Rate.
CR	→ Continuous Rate.
CRA	→Continuous Rate Assignment.
Eb/N0	→ The ratio between total power used for transmission divided by the number of information bits per second and the noise power density.
FCA	→Free Capacity Assignment.
GoG	→ Grade of generosity
MAC	→ Medium Access Control
MSL	→ Minimum Scheduler Latency.
NCC	→ Network Control Centre
PER	→ Packet Error Ratio.
QoS	→ Quality of Service
RBDC	→Rate Based Dynamic Capacity.
RCSTs	→ Remote Communities Services Telecentre
RT	→Real Time.
RTT	→ Radio Transmission Technology
SP	→Slump point
TBTP	→ Terminal Burst Time Plan.
TG	→ Traffic Gateway
UBR	→Unspecified Bit Rate.
VBDC	→Volume Based Dynamic Capacity.
VBR-nrt	→Variable Bit Rate - non real time.
VBR-rt	→Variable Bit Rate - real time.
VR	→Variable Rate.
JT	→ Jitter Tolerant

APPENDIX B: CD CONTENTS

The CD contains a copy of the paper zipped in PDF format. Furthermore, there is the file with the matlab code used to simulate the results.

APPENDIX C: ALGORITHM MATLAB CODE

Function ‘main’

```
%Generating Time of request by nodes
N= input ('How many packets per Node Do you want to send')
N1=exprnd(1,1,N);
N2=exprnd(1,1,N);
N3=exprnd(1,1,N);
N4=exprnd(1,1,N);
Initiate_parameters;
%time vector arrangement
T1=sort (N1);
T2=sort (N2);
T3=sort (N3);
T4=sort (N4);
%Bit masks to define the external packets 50%
mask1 = round(rand(1,N));
mask2 = round(rand(1,N));
mask3 = round(rand(1,N));
mask4 = round(rand(1,N));
%Time vector
T=[T1,T2,T3,T4];
%Time vector arrenged
Ts=sort(T);

%finding the node source
find_source
phi=A./B;
psi=C./D;

figure%Added system
THRP=Thrp1+Thrp2+Thrp3+Thrp4;
subplot(2,1,1), plot(V_pak,THRP)
title('Added packets sent')
ylabel('packets sent')
xlabel('Packets treated')

PHI=phi1+phi2+phi3+phi4;
subplot(2,1,2), line([V_pak,V_pak],[k,PHI/4])
title('Added network phi')
ylabel('p.sent/p.requested')
xlabel('packets requested')

figure%THrpt 4 nodes
subplot(2,2,1), plot(request1,Thrp1)
axis ([0 1000 0 500])
title('Node1')
ylabel('packets sent')
xlabel('packets requested')
subplot(2,2,2), plot(request2,Thrp2)
axis ([0 1000 0 500])
title('Node2')
ylabel('packets sent')
xlabel('packets requested')
subplot(2,2,3), plot(request3,Thrp3)
axis ([0 1000 0 500])
```

```
title('Node3')
ylabel('packets sent')
xlabel('packets requested')
subplot(2,2,4), plot(request4,Thrp4)
axis ([0 1000 0 500])
title('Node4')
ylabel('packets sent')
xlabel('packets requested')

figure%phi 4 nodes
subplot(2,2,1), plot(request1,phi1)
title('Node1')
ylabel('p.sent/p.requested')
xlabel('packets requested')
subplot(2,2,2), plot(request2,phi2)
title('Node2')
ylabel('p.sent/p.requested')
xlabel('packets requested')
subplot(2,2,3), plot(request3,phi3)
title('Node3')
ylabel('p.sent/p.requested')
xlabel('packets requested')
subplot(2,2,4), plot(request4,phi4)
title('Node4')
ylabel('p.sent/p.requested')
xlabel('packets requested')
%
figure%external added system
OutT=OutThrp1+OutThrp2+OutThrp3+OutThrp4;
PHI_ext=phi_ext1+phi_ext2+phi_ext3+phi_ext4;
REQEXT=req_ext1+req_ext2+req_ext3+req_ext4
%
subplot(2,1,1), plot(REQEXT,OutT)
title('Added packets sent')
ylabel('Out Data Throughtput')
xlabel('time')

subplot(2,1,2), line([REQEXT,REQEXT],[k,PHI_ext/4])
title('Added network phi')
ylabel('external p.sent/p.requested')
xlabel('packets requested')
%
figure%External Thgroughput
subplot(2,2,1),plot(req_ext1,OutThrp1)
axis([0 500 0 250])
title('Out Data Throughtput from Node1')
ylabel('Packets sent to the satellite')
xlabel('Out Packets Requested')
subplot(2,2,2),plot(req_ext2,OutThrp2)
axis([0 500 0 250])
title('Out Data Throughtput from Node2')
ylabel('Packets sent to the satellite')
xlabel('Out Packets Requested')
subplot(2,2,3),plot(req_ext3,OutThrp3)
axis([0 500 0 250])
title('Out Data Throughtput from Node3')
ylabel('Packets sent to the satellite')
xlabel('Out Packets Requested')
subplot(2,2,4),plot(req_ext4,OutThrp4)
axis([0 500 0 250])
```

```
title('Out Data Throughtput from Node4')
ylabel('Packets sent to the satellite')
xlabel('Out Packets Requested')
%
figure%External PHI
subplot(2,2,1), plot(req_ext1,phi_ext1)
title('Node1')
ylabel('external p.sent/p.requested')
xlabel('Out Packets Requested')
subplot(2,2,2), plot(req_ext2,phi_ext2)
title('Node2')
ylabel('external p.sent/p.requested')
xlabel('Out Packets Requested')
subplot(2,2,3), plot(req_ext3,phi_ext3)
title('Node3')
ylabel('external p.sent/p.requested')
xlabel('Out Packets Requested')
subplot(2,2,4), plot(req_ext4,phi_ext4)
title('Node4')
ylabel('external p.sent/p.requested')
xlabel('Out Packets Requested')
```

Function ‘initiate_parameters’

```
%this function 'initiate_parameters' all the parameters needed

A=[0 0 0 0];%number of relayed requested generated by the nodes
B=[0 0 0 0];%number of relay requested accepted by the nodes

phi=A./B
C=[0 0 0 0];%number of relayed requested made to the nodes
D=[0 0 0 0];%number of relay requested accepted to the nodes
psi=C./D

en1=input('Put the energy constraint per node1')
en2=input('Put the energy constraint per node2')
en3=input('Put the energy constraint per node3')
en4=input('Put the energy constraint per node4')
c=input('put the power concumption')
p=[en1 en2 en3 en4]  ;%energy constraint
Nn=4

n=[Nn/2];%for parameter tau


epsilon=input ('choose the level of generosity')%the level of generosity

rej=[0 0 0 0]%the total rejects from each node
Thrp=[0 0 0 0]%the throuput of each node

request=[0 0 0 0]%number of nodes requeted until time j
OutThrp=[0 0 0 0]%the external throughput to the satellite grom each node
req_ext=[0 0 0 0]%requested relays for external satellite
phi_ext=OutThrp./req_ext
```

Function ‘find_source’

```
%find the source
for i=1:N*4,
   tau=n*p;
   phi=A./B;
   psi=C./D;
   phi_ext=OutThrp./request;
   V_pak=[1:N*4];%the number of packets to set the plot
   k(i)=1/2.72;

   OutThrp1(i)=OutThrp(1);
   OutThrp2(i)=OutThrp(2);
   OutThrp3(i)=OutThrp(3);
   OutThrp4(i)=OutThrp(4);

   Thrp1(i)=Thrp(1);
   Thrp2(i)=Thrp(2);
   Thrp3(i)=Thrp(3);
   Thrp4(i)=Thrp(4);

   phi1(i)=phi(1);%phi vectors to set the plot
   phi2(i)=phi(2);
   phi3(i)=phi(3);
   phi4(i)=phi(4);

   phi_ext1(i)=phi_ext(1);
   phi_ext2(i)=phi_ext(2);
   phi_ext3(i)=phi_ext(3);
   phi_ext4(i)=phi_ext(4);

   request1(i)=request(1);
   request2(i)=request(2);
   request3(i)=request(3);
   request4(i)=request(4);

   req_ext1(i)=req_ext(1);
   req_ext2(i)=req_ext(2);
   req_ext3(i)=req_ext(3);
   req_ext4(i)=req_ext(4);

   for j=1:N,
   if Ts(i)==T1(j)

      source_1%the node one will be the source
   end
   if Ts(i)==T2(j)
      source_2
   end
   if Ts(i)==T3(j)
      source_3
   end
   if Ts(i)==T4(j)
      source_4
   end
end
end    if Ts(i)==T4(j)
      source_4
```

```
    end
end
end
```

Function ‘source1’

```
%this function accepts or rejects a packet from node 1
%source_1-->treats the packet from node 1, choosing the way to node3, randomly

%'way' will be 1 if it chooses N2 as a relay and 2 if is N4 as a realy
way=random('unid',2,1,1);
request(1)=request(1)+1;
if way==1;
    D(2)=D(2)+1;
    B(1)=B(1)+1;
     if mask1(j)~=0
           req_ext(1)=req_ext(1)+1;
        end
    if psi(2)>tau(2) | phi(2)<psi(2)-epsilon
        rej(2)=rej(2)+1;
    else
        A(1)=A(1)+1;
        C(2)=C(2)+1;
        p(1)=p(1)-c;%power consumption
        p(2)=p(2)-c;
        p(3)=p(3)-c;
        Thrp(1)=Thrp(1)+1;%increases the source throughput
        if mask1(j)~=0
           OutThrp(1)=OutThrp(1)+1;
        end
end
end
if way==2;
    D(4)=D(4)+1;
    B(1)=B(1)+1;
     if mask1(j)~=0
           req_ext(1)=req_ext(1)+1;
        end
    if psi(4)>tau(4) | phi(4)<psi(4)-epsilon
        rej(4)=rej(4)+1;
    else
        A(1)=A(1)+1;
        C(4)=C(4)+1;
        p(1)=p(1)-c;%power consumption
        p(3)=p(3)-c;
        p(4)=p(4)-c;
        Thrp(1)=Thrp(1)+1%increases the source throughput
           if mask1(j)~=0
           OutThrp(1)=OutThrp(1)+1;
        end
end
end
```

Function ‘source2’

```
%Source_2-->this function accepts or rejects a packet from node 2
%source_2-->treats the packet from node 2, choosing the way to node4, randomly

%'way' will be 1 if it chooses N1 as a relay and 2 if is N3 as a realy
way=random('unid',2,1,1);
request(2)=request(2)+1;
if way==1;
    D(1)=D(1)+1;
    B(2)=B(2)+1;
     if mask1(j)~=0
           req_ext(2)=req_ext(2)+1;
        end
    if psi(1)>tau(1) | phi(1)<psi(1)-epsilon
        rej(1)=rej(1)+1;
    else
        A(2)=A(2)+1;
        C(1)=C(1)+1;
        p(1)=p(1)-c;%power consumption
        p(2)=p(2)-c;
        p(4)=p(4)-c;
        Thrp(2)=Thrp(2)+1;%increases the source throughput
        if mask2(j)~=0
           OutThrp(2)=OutThrp(2)+1;
        end
end
end
if way==2;
    D(3)=D(3)+1;
    B(2)=B(2)+1;
     if mask1(j)~=0
           req_ext(2)=req_ext(2)+1;
        end
    if psi(3)>tau(3) | phi(3)<psi(3)-epsilon
        rej(3)=rej(3)+1;
    else
        A(2)=A(2)+1;
        C(3)=C(3)+1;
        p(2)=p(2)-c;%power consumption
        p(3)=p(3)-c;
        p(4)=p(4)-c;
        Thrp(2)=Thrp(2)+1;%increases the source throughput
        if mask2(j)~=0
           OutThrp(2)=OutThrp(2)+1;
        end
end
end
```

Function ‘source3’

```
%Source_3-->this function accepts or rejects a packet from node 3
```

```
%source_3-->treats the packet from node 3, choosing the way to node1, randomly

%'way' will be 1 if it chooses N1 as a relay and 2 if is N3 as a realy
way=random('unid',2,1,1);
request(3)=request(3)+1;
if way==1;
    D(2)=D(2)+1 ;
    B(3)=B(3)+1;
     if mask1(j)~=0
          req_ext(3)=req_ext(3)+1;
        end
    if psi(2)>tau(2) | phi(2)<psi(2)-epsilon
        rej(2)=rej(2)+1;
    else
        A(3)=A(3)+1;
        C(2)=C(2)+1;
        p(3)=p(3)-c;%power consumption
        p(2)=p(2)-c;
        p(1)=p(1)-c;
        Thrp(3)=Thrp(3)+1;%increases the source throughput
        if mask3(j)~=0
           OutThrp(3)=OutThrp(3)+1;
        end
end
end
if way==2;
    D(4)=D(4)+1;
    B(3)=B(3)+1;
     if mask1(j)~=0
          req_ext(3)=req_ext(3)+1;
        end
    if psi(4)>tau(4) | phi(4)<psi(4)-epsilon
        rej(4)=rej(4)+1;
    else
        A(3)=A(3)+1;
        C(4)=C(4)+1;
        p(3)=p(3)-c;%power consumption
        p(4)=p(4)-c;
        p(1)=p(1)-c;
        Thrp(3)=Thrp(3)+1;%increases the source throughput
          if mask3(j)~=0
           OutThrp(3)=OutThrp(3)+1;
        end
end
end
```

Function ‘source4’

```
%Source_4-->this function accepts or rejects a packet from node 2
%source_4-->treats the packet from node 4, choosing the way to node2, randomly

%'way' will be 1 if it chooses N1 as a relay and 2 if is N3 as a realy
way=random('unid',2,1,1);
request(4)=request(4)+1;
```

```
if way==1;
   D(1)=D(1)+1 ;
   B(4)=B(4)+1;
    if mask1(j)~=0
        req_ext(4)=req_ext(4)+1;
      end
   if psi(1)>tau(1) | phi(1)<psi(1)-epsilon
      rej(1)=rej(1)+1;
   else
      A(4)=A(4)+1;
      C(1)=C(1)+1;
      p(1)=p(1)-c;%power consumption
      p(4)=p(4)-c;
      p(2)=p(2)-c;
      Thrp(4)=Thrp(4)+1;%increases the source throughput
      if mask4(j)~=0
        OutThrp(4)=OutThrp(4)+1;
      end
   end
end
if way==2;
   D(3)=D(3)+1;
   B(4)=B(4)+1;
    if mask1(j)~=0
        req_ext(4)=req_ext(4)+1;
      end
   if psi(3)>tau(3) | phi(3)<psi(3)-epsilon
      rej(3)=rej(3)+1;
   else
      A(4)=A(4)+1;
      C(3)=C(3)+1;
      p(3)=p(3)-c;%power consumption
      p(4)=p(4)-c;
      p(2)=p(2)-c;
      Thrp(4)=Thrp(4)+1;%increases the source throughput

      if mask4(j)~=0
       OutThrp(4)=OutThrp(4)+1;
   end
end
end
```

function ‘casezero’

```
C=[0 0.0001 0.0002 0.0003 0.0004 0.0005 0.0006 0.0007 0.0008 0.0009 0.001]
Rmax=[1 1 3619/4000 2813/4000 2327/4000 1992/4000 1710/4000 1534/4000 1340/4000 1245/4000
1140/4000] %the values registered from the simulations
figure
plot(C,Rmax)
title('Maximum Rate-V.S-Energy consumption')
ylabel('Rmax')
xlabel('energy consumption')
```

function ‘caseone’

```
figure
TH=[THRP1; THRP2; THRP3; THRP4; THRP5; THRP6; THRP7; THRP8; THRP9; THRP10];
PR=[V_pak; V_pak; V_pak; V_pak; V_pak; V_pak; V_pak; V_pak; V_pak; V_pak;];
line([PR]',[TH]')
title('Energy Constraint Dependence')
ylabel('Throughput')
xlabel('Packets treated')

figure
ro=[0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1];
THP=[141 280 425 534 679 834 949 1129 1230 1356];
plot (ro, THP)
title('Energy Constraint Dependence')
ylabel('Throughput')
xlabel('Energy Constraint')
```

function ‘casetwo’

```
sp=[1474 1490 1539 1555 1700 1705 1705 1706 1900 1950 2024 2013 2108 2190 2365 2287 2402 2386
2462 2407 2490]
eps=[0 0.005 0.01 0.015 0.02 0.025 0.03 0.035 0.04 0.045 0.05 0.055 0.06 0.065 0.07 0.075 0.08 0.085
0.09 0.095 0.1]
figure
plot(eps,sp)
title('THROUGHPUT QUANTIFICATION -VS- GRADE OF GENEROSITY')
ylabel('Slump Point (packets)')
xlabel('Grade of Generosity')
```

function ‘vzero’

```
function [mask]=vzero; %creates a binary array to create the external traffic
z=0;
for i=1:1000;
    if z<200;
        a=random('unid',2,1,1);
        if a==1;
        mask(i)=1;
        z=z+1;
        else a==2;
        mask(i)=0;
        end
    end
end
 mask(i)=1;
end
```

En aquest treball mostrem una primera aproximació a l'evolució de les xarxes Ad-Hoc cooperatives.

Donat que els nodes wireless disposen d'energia finita, poden no estar interessats en transmetre tràfic d'altres nodes. Per altra banda, si cap node decideix gastar energia en passar tràfic d'altres, llavors la tassa de transferència a la xarxa cau críticament. Aquests casos extrems son desfavorables per l'usuari. En aquest treball tractem aquestes qüestions gràcies al desenvolupament d'un algoritme anomenat "Generous Tit-For-Tat"

Assumirem que els nodes son egoistes y tenen energia finit, així que les decisions es determinaran pel seu propi interès y cada node s'associarà amb un temps limitat d'energia. Donades aquestes limitacions y la suposició del comportament racional, estudiarem el comportament agregat de la xarxa.

En este proyecto mostramos un primer acercamiento a la evolución de las redes Ad-Hoc cooperativas.

Puesto que los nodos wireless disponen de energía finita, puede que no estén interesados en aceptar transmitir tráfico de otros nodos. Por otra parte, si ningún nodo decide gastar energía en retransmitir tráfico de otros, entonces la tasa de transferencia en la red cae críticamente. Estos casos extremos son desfavorables para el usuario. En este trabajo tratamos estas cuestiones gracias al desarrollo de un algoritmo llamado "Generous Tit-For Tat".

Asumiremos que los nodos son egoístas y tienen energía finita, así que las decisiones se determinarán por propio interés y cada nodo será asociado con un tiempo limitado de energía. Dadas esas limitaciones y la suposición del comportamiento racional estudiaremos el comportamiento agregado de la red.

In this paper, we present a first approach to evolve a cooperative behavior in ad hoc networks.

Since wireless nodes are energy constrained, it may not be in the best interest of a node to always accept relay requests. On the other hand, if all nodes decide not to expend energy in relaying, then network throughput will drop dramatically. Both these extreme scenarios are unfavorable to the interests of a user. In this paper we deal with the issue of user cooperation in ad hoc networks by developing the algorithm called Generous Tit-For-Tat.

We assume that nodes are rational, i.e., their actions are strictly determined by self-interest, and that each node is associated with a minimum lifetime constraint. Given these lifetime constraints and the assumption of rational behavior, we study the added behavior of the network.