UNIVERSITÀ DI PISA
FACOLTÀ di INGEGNERIA
LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA
ANNO ACCADEMICO 2006-2007

# IEEE 802.11s Mesh Deterministic Access : Design and analysis

Candidato: Soleri Michele

Relatore: Prof. Lenzini Luciano
Relatore: Prof. Mingozzi Enzo
Relatore: Ing. Cicconetti Claudio

*A Stefania*
*e*
*alla mia famiglia,*
*che mi hanno supportato*
*e sopportato durante*
*questi anni di vita*
*universitaria*

Ringrazio il professor Lenzini che mi ha dato l'opportunità di approfondire un argomento di grande interesse, il professor Mingozzi per le osservazioni e i suggerimenti e l'Ing. Cicconetti per la costante e paziente presenza.
Ringrazio, inoltre, gli amici del laboratorio rosso che hanno allietato il lungo lavoro con la loro simpatia.

*IEEE 802.11s is a draft IEEE 802.11 amendment for mesh networking, defining how wireless devices can interconnect to create an ad-hoc network. It includes some mesh-specific optional MAC enhancements like Mesh Deterministic Access, Common Channel Framework, Intra-mesh Congestion Control and Power Management. Mesh Deterministic Access (MDA) is an access method that allows MPs to access the channel at selected times (called MDAOPs) with lower contention than would otherwise be possible. In this work we study Mesh Deterministic Access (MDA) feature. Specifically: we implement 802.11s in ns-2 simulator and evaluate performance comparing results with those obtained with DCF. We also propose an improvement called Dynamic Relocation. Dynamic Relocation permits to overcome MDA limits by reallocating MDAOPs basing on statistics collected during transmission times. The effectiveness of MDA improved with Dynamic Relocation in a scenario with realistic traffic is then confirmed via a simulation analysis.*

# 1. Introduction

Wireless Mesh Networks (WMNs) have emerged as a key technology for next-generation wireless networking. Because of their advantages over other wireless networks, WMNs are undergoing rapid progress and inspiring numerous applications.

Wireless mesh networks (WMNs) are dynamically self-organized and self-configured, with the nodes in the network automatically establishing an ad hoc network and maintaining the mesh connectivity. WMNs are comprised of two types of nodes: mesh routers and mesh clients. Other than the routing capability for gateway/bridge functions as in a conventional wireless router, a mesh router contains additional routing functions to support mesh networking [1]. Through multi-hop communications, the same coverage can be achieved by a mesh router with much lower transmission power. To further improve the flexibility of mesh networking, a mesh router is usually equipped with multiple wireless interfaces built on either the same or different wireless access technologies. In spite of all these differences, mesh and conventional wireless routers are usually built based on a similar hardware platform. Mesh routers have minimal mobility and form the mesh backbone for mesh clients.

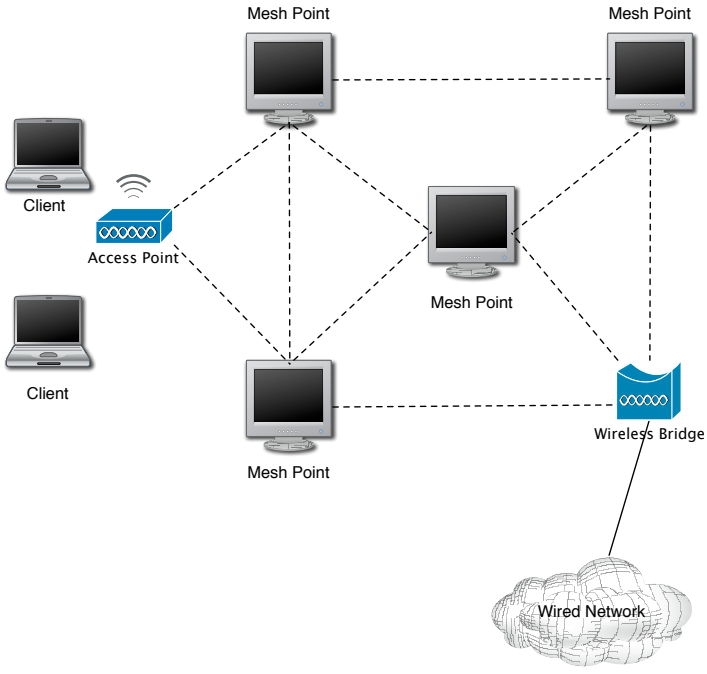


*Figure 1.1 : Example of Mesh Network Architecture*

In addition to mesh networking among mesh routers and mesh clients, the gateway/ bridge functionalities in mesh routers enable the integration of WMNs with various other networks. Conventional nodes equipped with wireless network interface cards (NICs) can connect directly to WMNs through wireless mesh routers. Customers without wireless NICs can access WMNs by connecting to wireless mesh routers through, for example, Ethernet. Thus, WMNs will greatly help users to be always-on-line anywhere, anytime. Consequently, instead of being another type of ad-hoc networking, WMNs diversify the capabilities of ad-hoc networks. This feature brings many advantages to WMNs, such as low up-front cost, easy network maintenance, robustness, reliable service coverage, etc. Therefore, in addition to being widely accepted in the traditional application sectors of ad hoc networks, WMNs are undergoing rapid commercialization in many other application scenarios such as broadband home networking, community networking, building automation, high-speed metropolitan area networks, and enterprise networking products.

Researchers have started to revisit the protocol design of existing wireless networks, especially of IEEE 802.11 networks, ad hoc networks, and wireless sensor networks, from the perspective of wireless mesh networking. Industrial standards groups, such as IEEE 802.11, IEEE 802.15, and IEEE 802.16, are all actively working on new specifications for WMNs.

The 802.11 working group in IEEE has recently started working on mesh networks in the task group identified as TGs, which will produce the 802.11s standard for mesh networks [2]. The main target of TGs is to investigate and design mesh networks consisting of different WLAN devices performing routing at link layer. TGs is currently working on the presentation and selection of different proposals.
The standard is aimed for approval in 2008. Specifically, 802.11 TGs defines an extended service set (ESS) mesh as a collection of WLAN devices interconnected with wireless links that enable automatic topology learning and dynamic path configuration. 802.11 mesh networks will be based on extensions to the IEEE 802.11 MAC standard, based on the definition of a mesh network architecture and new protocol mechanisms. The architecture will provide an IEEE 802.11 Wireless Distribution System (DS) that supports both broadcast/multicast and unicast delivery at the MAC layer using radio-aware metrics over self-configuring multi-hop topologies, thus providing the functional equivalent of a wired DS.

In this work we start by analyzing 802.11s standard and then we focus on one of its optional MAC enhancements, which is Mesh Deterministic Access. Mesh Deterministic Access (MDA) is an access method that allows MPs to access the channel at selected times (called MDAOPs) with lower contention than would otherwise be possible.

We also propose a new mechanism to improve MDA, which is called Dynamic Relocation.

We finally implement MDA and Dynamic Relocation in ns2 network simulator, where is already present the IEEE 802.11 stack and evaluate performance via a simulation analysis.

# 2. MAC Protocol

This section describes the IEEE 802.11 MAC protocol and the enhancements proposed in the IEEE 802.11s draft.

## 2.1. 802.11

The IEEE 802.11 includes two access mechanisms: the Distributed Coordination Function (DCF) and the Point Coordination Function (PCF).

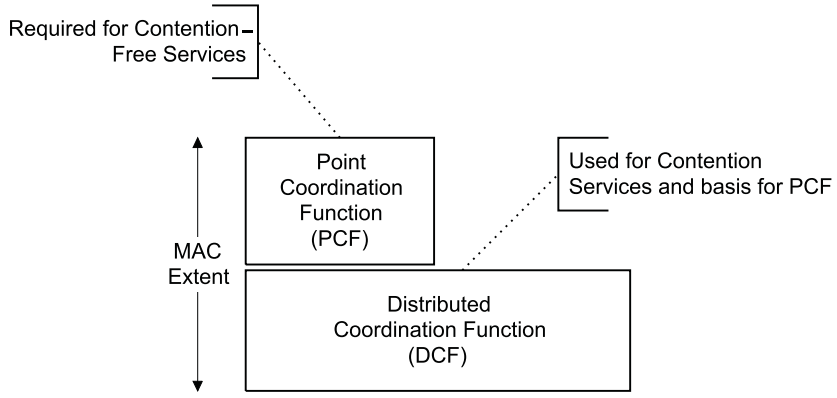


*Figure 2.1 : 802.11 Mac Architecture*

The DCF uses a CSMA/CA access scheme : for a station (STA) to transmit, it shall sense the medium to determine if another STA is transmitting; if the medium is idle, then the transmission may proceed; if the medium is busy, then the STA shall defer until the end of the current transmission. The STA performs a backoff procedure after deferral in order to minimize collisions; the backoff procedure starts after the medium has been sensed idle for a DCF Interframe Space (DIFS). When a unicast frame is correctly received by an STA, the latter transmits an acknowledgment frame after a Short Interframe Space (SIFS), which is shorter than DIFS in order to prevent deferring stations from interrupting ongoing frame exchange sequences.

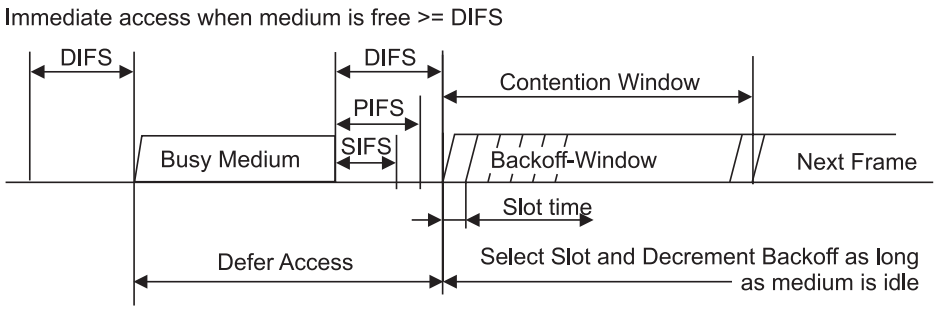


*Figure 2.2 : DCF Inter Frame Spaces*

If the Point Coordination Function is used, then the AP alternates a Contention-Free Period (CFP) with a Contention Period (CP). During the CP, the above DCF transfer rules apply. During the CFP the AP polls the stations that are in the CF poll list, using an implementation-dependent algorithm. When polled, a station transmits an MSDU. The AP may also use the CFP to transmit frames addressed to associated stations. The first frame of the CFP is transmitted by the AP at regular intervals (Target Beacon Transmission Time, TBTT) after the medium has been sensed idle for a duration greater than a PCF Inter-Frame Space (PIFS), which is chosen such that SIFS < PIFS < DIFS.

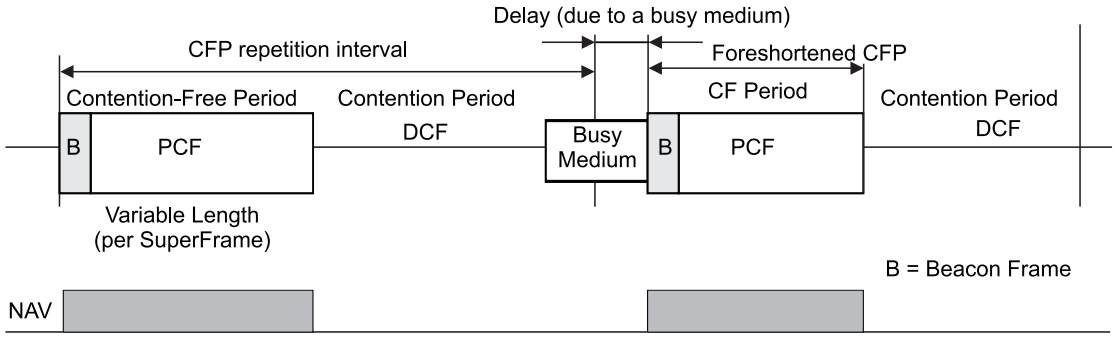


*Figure 2.3 : CFP/CP alternation*

Even though it has been shown that the PCF performs better than the DCF with real-time traffic , it is nevertheless unable to guarantee that associated stations will have a parameterized QoS. The reasons behind this failure are: (i) the lack of knowledge at the AP of the traffic specifications and requirements; (ii) no mechanism to determine if a station actually has data to send, which leads to a high polling overhead; (iii) the

interval between two CFPs is too large and the duration of the CFP is limited; (iv) the TBTT may be delayed due to stations unaware of the PCF procedure; and, finally, (v) no admission control can be performed at the AP.

## 2.2. 802.11s

The IEEE 802.11s extended service set (ESS) mesh aims at applying multi-hop mesh techniques to specify a WDS that can be used to build a wireless infrastructure for small-to-large-scale WLANs. Hence, the ESS or WLAN mesh can be considered as an IEEE 802.11-based WDS, a subset of the DS that consists of a set of devices interconnected with each other via wireless links, resulting in a 'mesh of connectivity.' The activities of 802.11s TG comprise the specification of a new protocol suite for the installation, configuration, and operation of WLAN mesh [3]. Its implementation shall be atop the existing PHY layer of IEEE 802.11a/b/g/n operating in the unlicensed spectrum of 2.4- and 5-GHz frequency bands. The specification shall include the extensions in topology formation to make the WLAN mesh self-configure as soon as the devices are powered up. A path selection protocol will be specified in the MAC layer instead of network layer for routing data in the multi-hop mesh topology. This standard is expected to support MAC-layer broadcast and multicast in addition to the unicast transmissions. This standard shall also accommodate devices that are able to support multichannel operations, or are equipped with multiple radios, with an aim to boost the capacity of the overall network. The specification is expected to be adopted as part of the working group standard by March 2008.

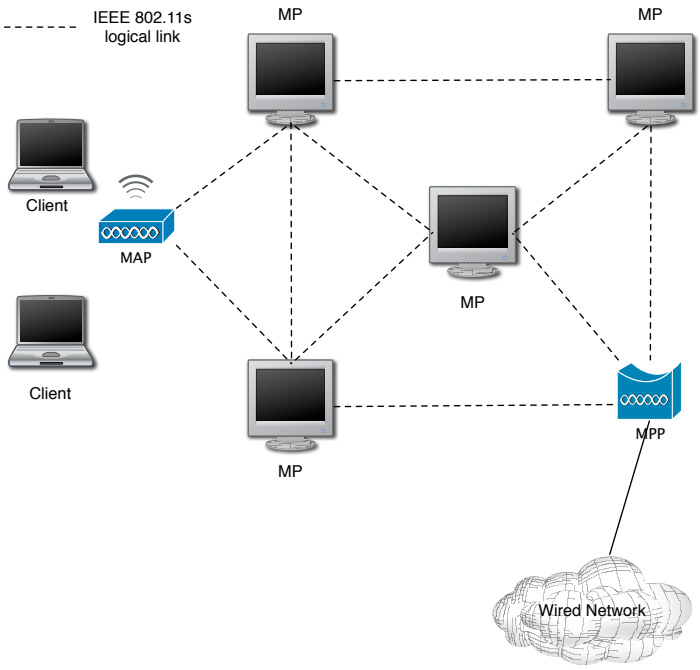


*Figure 2.4 : Example of 802.11s Mesh Network Architecture*

The minimal MP operations include neighbor discovery, channel selection, and forming an association with neighbors. Besides, MPs can directly communicate with their neighbors and forward traffic on behalf of other MPs via bidirectional wireless mesh links. A set of MPs and the mesh links form a WDS, which distinguishes itself from the BSS as defined in the legacy IEEE 802.11. The proposed WLAN mesh also defines a mesh access point (MAP), which is a specific MP but acts as an AP as well. The MAP may operate as a part of the WLAN mesh or in one of the legacy 802.11 modes.

A mesh portal (MPP) is yet another type of MP through which multiple WLAN meshes can be interconnected to construct networks of mesh networks. An MPP can also co-locate with an IEEE 802.11 portal and function as a bridge/gateway between the WLAN mesh and other networks in the DS. To uniquely identify a WLAN mesh, a common mesh ID is assigned to each MP, similar to the use of service set identifier (SSID) to represent an ESS in legacy 802.11 networks.

The major components of the proposed 802.11s Mesh Coordination Function (MCF) are shown in figure.

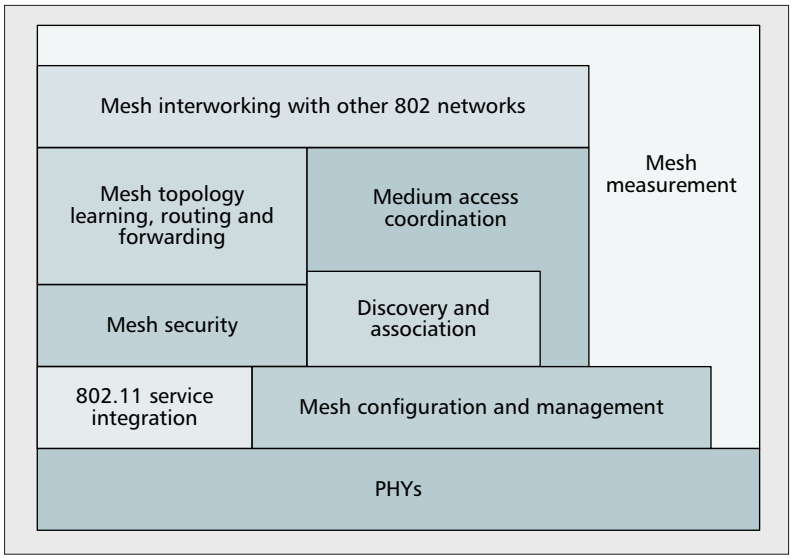


*Figure 2.5 : Components of 802.11s MCF*

Built on top of the legacy physical layer specification, 802.11s shall explicitly provide the WLAN mesh services. These include topology learning, routing and forwarding, medium access coordination, mesh configuration and management, topology discovery and association, mesh measurement, interworking, and security functions.

For interworking of the WLAN mesh with other networks, the IEEE 802.1D standard shall be incorporated in the mesh portals (MPPs), which define the interworking framework and service access interface across all 802 standards. Similarly, security architecture shall be based on the IEEE 802.11i standard, which specifies security features for all WLAN networks.

### 2.2.1. MAC frame formats

The MAC frame format comprises a set of fields that occur in a fixed order in all frames. Figure depicts the general MAC frame format.

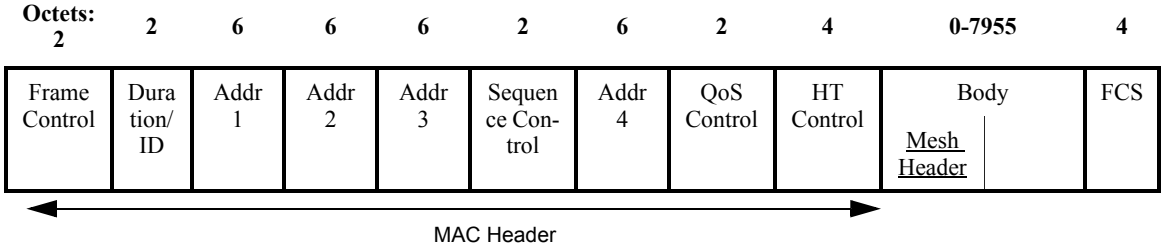


*Figure 2.6 : MAC frame format*

The first three fields (Frame Control, Duration/ID, and Address 1) and the last field (FCS) in figure constitute the minimal frame format and are present in all frames, including reserved types and subtypes. The fields Address 2, Address 3, Sequence Control, Address 4, QoS Control, HT Control, Mesh Header, and Frame Body are present only in certain frame types and subtypes. When present, the Mesh Header field is prepended to the Frame Body. The Mesh Header field is handled identically to the contents of the body with respect to MPDU processing.

The mesh header field is a 5 to 23 octet field that includes:

- an 8-bit Mesh Flags field to control mesh header processing
- a time to live field for use in multi-hop forwarding to aid in limiting the effect of transitory path selection loops
- a mesh sequence number to suppress duplicates in broadcast/multicast forwarding and for other services
- in some cases a 6, 12, or 18-octet mesh address extension field containing extended addresses enabling up to a total of 6 addresses in mesh frames

The Mesh Header field, shown in figure, is present in Data frames if and only if they are transmitted between peer MPs with an established peer link. Data frames including the Mesh Header field are referred to as Mesh Data frames.

| **Octets: 1** | **1** | **3** | **0, 6, 12, or 18** |
|---|---|---|---|
| Mesh Flags | Mesh Time To Live (TTL) | Mesh Sequence Number | Mesh Address Extension (present in some configurations) |

*Figure 2.7 : Mesh Header Field*

The Frame Body is a variable length field that contains information specific to individual frame types and subtypes.

### 2.2.2. Management frame

Management frames are exchanged between neighboring Mesh Points in order to solve important tasks, like beaconing and association. The frame format for a management frame is defined in figure.

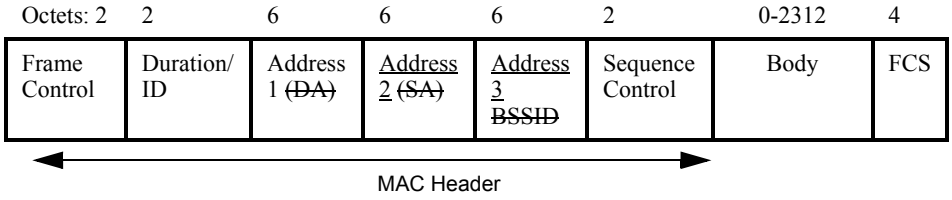


*Figure 2.8 : Management Frame Format*

The body of a management frame contains Information Elements, depending on the subtype. A partial list of IEs is contained in the following table :

| Information Element | Length |
|---|---|
| Mesh Configuration | 17 |
| Mesh ID | 2 to 34 |
| Link Metric Report | 3 to 257 |
| Congestion Notification | 10 |
| Peer Link Management | 5 to 9 |
| Mesh Channel Switch Announcement | 15 |
| Mesh Neighbor List | 4 to 257 |
| Beacon Timing | 7 to 257 |
| MDAOP Setup Request | 7 |
| MDAOP Setup Reply | 4 to 257 |
| MDAOP Advertisements | 3 to 257 |
| MDAOP Set Teardown | 9 |

| Information Element | Length |
|---|---|
| Path Request | 39 to 257 |
| Path Reply | 34 to 257 |
| Path Error | 14 |

*List of Information Elements*

Specific management frame types are :

- Beacon frame
- Probe Request frame
- Probe Response frame
- Multihop Action frame

Other types of Management frame formats, called Action frames, are distinguished by a category and action field inside Management body field.
Categories currently defined are :

- Mesh Peer Link Management
- Mesh Link Metric
- Mesh Path Selection
- Mesh Interworking
- Mesh Resource Coordination

Frames inside categories are differentiated by Action Field.

**Mesh Peer Link Management** category includes :

- Peer Link Open frame
- Peer Link Confirm frame
- Peer Link Close frame

The Peer Link Open frame is used to open a peer link. Peer Link Confirm frame is used by neighbor to confirm a peer link. Peer Link Close frame is used to close a peer link.

**Mesh Link Metric** category includes :

- Link Metric Request
- Link Metric Report

The Link Metric Request frame is transmitted by an MP to a peer MP in a mesh to request metric information. The Link Metric Report frame is transmitted by an MP to a peer MP in a mesh to advertise metric information.

**Mesh Path Selection** category includes :

- Path Request
- Path Reply
- Path Error
- Root Announcement
- RA-OLSR

The Path Request frame is transmitted by a source MP to discover the path to the destination MP using the HWMP protocol. The Path Reply frame is transmitted by a destination MP to a source MP in the mesh to determine the path between the source and destination MP. The Path Error frame is transmitted by an MP that detected a link error on a mesh path to the precursor MP(s). The Root Announcement frame is advertised by a Root MP. These frames may be transmitted using group addresses or individual addresses.

The RA-OLSR frame is transmitted, instead, by MPs using the RA-OLSR path selection protocol.

**Mesh Interworking** category includes :

- Portal Announcement

The Portal Announcement is transmitted by an MPP to announce its presence in the mesh network. This frame is transmitted using group addresses.

**Mesh Resource Coordination** category includes :

- Congestion Control Notification
- MDA Setup Request
- MDA Setup Reply
- MDAOP Advertisement Request
- MDAOP Advertisements
- MDAOP Set Teardown
- Beacon Timing Request
- Beacon Timing Response
- Mesh Channel Switch Announcement
- Connectivity Report

The Congestion Control Notification frame is sent by an MP to its peer MP(s) to indicate its congestion status.

The Mesh Deterministic Access MDA Setup Request frame is used to request the setup of a set of MDAOPs. It is transmitted by an MDA-active MP to a peer MDA-active MP. The Mesh Deterministic Access MDA Setup Reply frame is used to reply to an MDAOP Setup Request. The MDAOP Advertisement Request frame is transmitted by an MDA-active MP to request MDA advertisements from neighbors. The Mesh Deterministic Access MDAOP Advertisements frame is transmitted by an MDA-active MP to one or more peer MDA-active MPs. This frame may be transmitted using group addresses or individual addresses. The Mesh Deterministic Access MDAOP Set Teardown frame is transmitted by an MDA-active MP to one or more peer MDA-active MPs.

The Beacon Timing Request frame is used to request beacon timing information from peer MPs. This frame is transmitted using group addresses or individual addresses.
The Beacon Timing Response frame is used to respond to a Beacon Timing Request frame with neighbor MP beacon timing information.

The Mesh Channel Switch Announcement frame is transmitted by an MP to signal a channel switch.

The Connectivity Report frame is transmitted by an MP to advertise the number of beacon broadcasters during the reporting interval and the peers that transmitted a connectivity report and the Power Management Mode of each peer.

### 2.2.3. Mesh networking procedures.

**Mesh discovery** procedures require that MPs have sufficient informations about themselves and potential neighbors. This process requires detection of potential mesh neighbors through Beacons or through active scanning using Probe Requests.

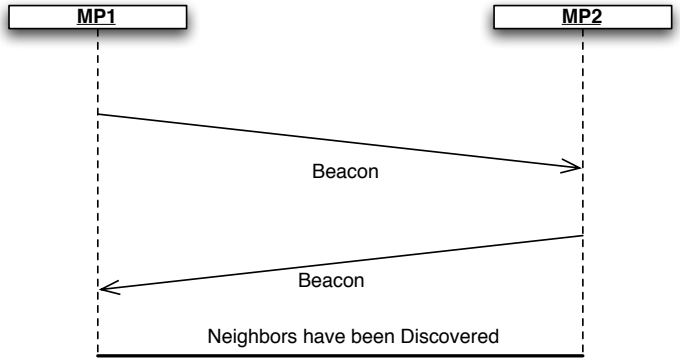


*Figure 2.9 : Neighbor Discovery through beacons*

An MP shall support at least one mesh profile. A mesh profile consists of:

1. A Mesh ID
2. A path selection protocol identifier
3. A path selection metric identifier

The purpose of discovery procedure is to discover candidate peer MPs and their properties, covering cases both before and after an MP is a member of a mesh network.

A configured MP, by definition, has at least one mesh profile. If the MP is a member of a mesh, exactly one mesh profile is active. An MP performs passive or active scanning to discover neighbor MPs.

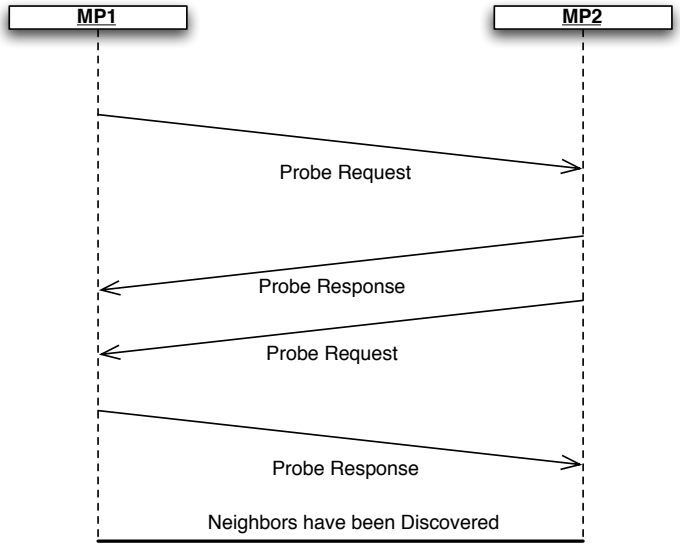


*Figure 2.10 : Neighbor Discovery through active scanning*

The MP attempts to discover all neighbor and candidate peer MPs, and maintains the neighbor MP information in Neighbor Table indicating the MAC address of each MP, the most recently observed link state parameters, the received channel number and state.

If an MP is unable to detect neighbor MPs, it may adopt a Mesh ID from one of its mesh profiles, and proceed to the active state. This may occur, for example, when the MP is the first MP to power on (or multiple MPs power on simultaneously). Peer MP links are established later as part of the continuous mesh peer link management procedures.

The **Mesh Peer Link Management** protocol is used to establish and close peer links between MPs. MPs shall not transmit data frames or management frames other than the ones used for discovery and peer link management until the peer link has been established. An MP shall be able to establish at least one mesh link with a peer MP, and may be able to establish many such links simultaneously, if the maximum number of peer MPs is not reached.

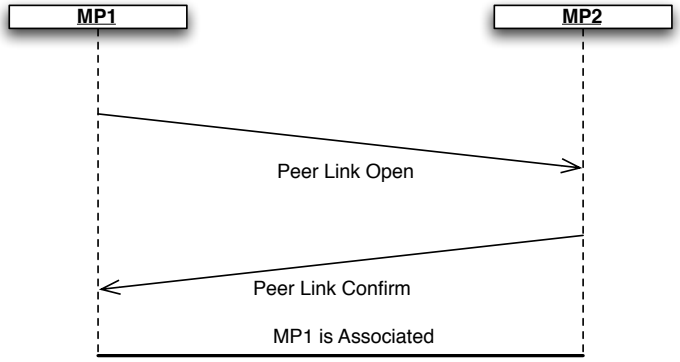


*Figure 2.11 : Peer Link Setup at MP1*

A Peer Link Open frame requests that a mesh link instance be established between the Peer Link Open sender and the receiver. The MP shall send a Peer Link Confirm frame in response to the Peer Link Open frame if the link instance proceeds with the protocol. The Peer Link Close frame is used to inform the receiver to close the mesh peer link. The protocol succeeds in establishing a mesh link when the following requirements are satisfied:

1. both MPs have sent and received (and correctly processed) a Peer Link Open frame regarding this mesh link
2. both MPs have sent and received (and correctly processed) a corresponding Peer Link Confirm frame regarding this mesh link.

An MP PHY shall perform **Network Channel Selection Procedure** in a controlled way such that it enables the formation of a mesh network that coalesces to a unified channel for communication. The MP PHY shall establish links with neighbors that match the Mesh ID and Mesh Profile and select its channel.

An MP PHY shall periodically perform passive or active scanning to discover neighboring MPs. If an MP is unable to detect neighbor MPs, it may adopt a Mesh ID from one of its profiles, select a channel for operation, and select an initial channel precedence value. The initial channel precedence value shall be initialized to a random value. In the event that an MP's PHY discovers a disjoint mesh, that is, the list of candidate peer MPs spans more than one channel, the MP shall select the channel that is indicated by the candidate peer MP that has the numerically highest channel precedence indicator to be the unification channel.

802.11s standard includes an extensible framework to enable flexible implementation of **Path Selection Protocols and Metrics** within the mesh framework. The standard includes a default mandatory path selection protocol (HWMP) and default mandatory path selection metric (Airtime Link Metric) for all implementations, to ensure minimum capabilities for interoperability between devices from different vendors. However, the standard also allows any vendor to implement any path selection protocol and/or path selection metric in the mesh framework to meet special application needs, for instance with high mobility of MPs. The mesh framework allows flexibility to integrate future path selection protocols for wireless mesh networks.
An MP may include multiple protocol implementations as well as multiple metric implementations, but only one path selection protocol and only one path selection metric shall be active in a particular mesh at a time.

The Hybrid Wireless Mesh Protocol (HWMP) is a mesh path selection protocol that combines the flexibility of on-demand path selection with proactive topology tree extensions. The combination of reactive and proactive elements of HWMP enables optimal and efficient path selection in a wide variety of mesh networks.

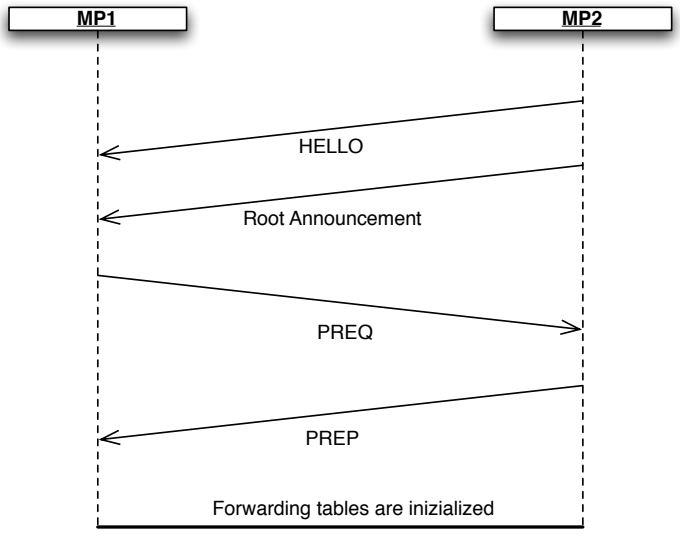


*Figure 2.12 : Path Selection*

HWMP supports two modes of operation depending on the configuration. These modes are:

- On demand mode: this mode allows MPs to communicate using peer-to-peer paths. The mode is used in situations where there is no root MP configured. It is also used in certain circumstances if there is a root MP configured and an on demand path can provide a better path to a given destination in the mesh.
- Proactive tree building mode

On demand and proactive modes may be used concurrently.

It is optional for an MP to support **synchronization**. An MP supporting synchronization may choose to be either synchronizing or non-synchronizing based on either its own requirements or the requirements of its peer MPs.
A synchronizing MP is an MP that updates its TSF timer based on the time stamps and offsets received in Beacon frames and Probe Response frames from other synchronizing MPs. Synchronizing MPs should attempt to maintain a common TSF time called the Mesh TSF time. All Beacon frames and Probe Response frames by synchronizing MAPs carry informations to advertise its self offset value relative to the Mesh TSF time. Synchronizing MPs translate the received time stamps and offsets from Beacon frames and Probe Response frames from other synchronizing MPs to their own timer base.

# 3. Mesh Deterministic Access

In this section we describe an optional Mesh networking procedure called MDA. We start with the version that appears in the 802.11s draft, then we show how we implemented that in network simulator (ns-2). Finally, we propose a new mechanism to improve MDA, which is called Dynamic Relocation.

## 3.1. MDA Standard

Mesh Deterministic Access (MDA) is an optional access method that allows MPs to access the channel at selected times with lower contention than would otherwise be possible. MDA sets up time periods (MDAOPs) in mesh neighborhoods when a number of MDA-supporting MPs that may potentially interfere with each others' transmissions or receptions are set to not initiate transmission sequences. For each such time period, supporting MPs that set up the state for the use of these time periods are allowed to access the channel using MDA access parameters (CWMin, CWMax, and AIFSN). In order to use the MDA method for access, an MP shall be a synchronizing MP.

An MDAOP is a period of time within every Mesh DTIM interval that is set up between the MDAOP owner and the addressed MP. Once an MDAOP is setup :

- The MDAOP owner uses CSMA/CA and backoff to obtain a TXOP and using parameters MDACWmin, MDACWmax, and MDAIFSN. The ranges of values allowed for MDACWMin, MDACWMax, and MDAIFSN parameters are identical to those allowed for EDCA.
- The MDAOP is advertised by both the MDAOP owner and the addressed MP. All MDA supporting MPs other than the MDAOP owner shall defer initiating new transmissions during the TXOP initiated in the MDAOP.

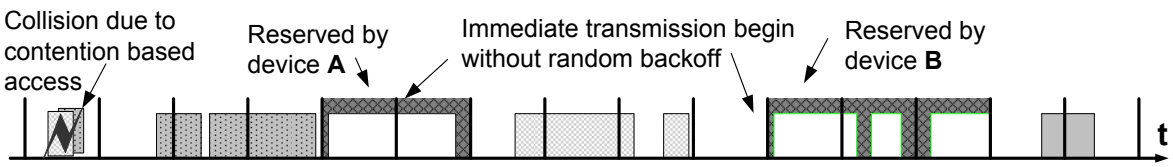


*Figure 3.1 : MDAOPs during DTIM*

A set of MDAOPs may be setup for individually addressed transmissions from a transmitter to a receiver by the transmitter. Such a set is identified by a unique ID called the MDAOP Set ID. The MDAOP Set ID has to be unique for a transmitter, so that the MDAOP set ID and the transmitter (or set owner) MAC address uniquely identifies an MDAOP set in the mesh. The MDAOP set ID is a handle that allows operation such as setup and teardown to be conducted together for the entire set of MDAOPs in an MDAOP set.

A TXOP that is obtained by an MP by using MDA parameters in an MDAOP is called an MDA TXOP. An MDA TXOP is required to end within the MDAOP in which it was obtained. Thus, an MDA TXOP ends either by MDA TXOP limit time after it began or by MDAOP end time, whichever is earlier.

In a MP's mesh neighborhood, all the TX-RX times reported by its neighbors (in their MDAOP advertisements) form a set of MDAOPs that are already being used in the neighborhood. No new MDAOPs may be set up by the MP during these times. These times are referred to as Neighborhood MDAOP times for the MP. In effect, Neighborhood MDAOP Times at an MP include all MDAOPs for which the MP and its neighbors are either the transmitters or receivers.

The Interfering times reported by an MP in its MDAOP advertisements are times that may not be used for a new MDAOP with that specific MP. While the MP itself is not the transmitter or receiver in an MDAOP during these times, one of its neighbors is. New MDAOPs to the MP during these times may experience interference. However, new MDAOPs may be setup with another MP during these interfering times. Thus, for every neighbor, there is a set of times that are interfering. These times are referred to as Neighbor MDAOP interfering times for that neighbor.

The MDA access fraction at an MP is the ratio of the total duration of its 'Neighborhood MDAOP Times' in a Mesh DTIM interval to the duration of the Mesh DTIM interval. This parameter may be used to limit the use of MDA in an MP's mesh neighborhood to a certain fraction of the total channel time. Before attempting to set up an MDAOP Set with a neighbor, an MP is required to ensure that the new MDAOP set does not cause the MAF of its neighbors to exceed their MAF Limit. An MDAOP setup request shall be refused by the intended receiver if the MAF limit of its own neighbors is exceeded due to the new setup.

The setup of an MDAOP set is initiated by the intended transmitter, and is accepted/ rejected by the intended receiver. Once accepted, the transmitter is referred to as the owner of the MDAOP. The setup procedure for an MDAOP set is as follows:

I. The MP that intends to be the transmitter in a new MDAOP set builds a map of Neighborhood MDAOP times in the Mesh DTIM interval after hearing Advertisements from all of its neighbors that have MDA active.
II. Based on traffic characteristics, it then chooses MDAOP starting times and durations in the Mesh DTIM interval that do not overlap with either its Neighborhood MDAOP Times or the Neighbor MDAOP Interfering Times of the intended receiver. It also avoids using times that are known to it as being used by itself or one of its neighbors for other activities such as beacon transmissions.
III. It then verifies that the new MDAOP Set will not cause the MAF limit to be crossed for its neighbors. If MAF limit would be crossed for its neighbors, due to the new MDAOP Set, it suspends the setup process.
IV. If the MAF limits at all neighbors are respected despite the new MDAOP set, it transmits an MDAOP Setup request information element to the intended receiver with chosen MDAOP locations and durations.
V. The receiver of the MDAOP Setup Request information element checks to see if the MDAOP times have overlap with its Neighborhood MDAOP Times. The receiver also checks if the new MDAOP Set will cause the MAF limit to be crossed for its neighbors. The MDAOP Setup Reply information element is used to reply to a setup request.
VI. The receiver rejects the setup request if there are overlaps of the requested MDAOP set with its Neighborhood MDAOP Times, or other times that it knows are set to be used by itself or its neighbors for activities such as beacon

transmissions. It may suggest alternate times by including the optional field Alternate Suggested Request information element in the MDAOP Setup Reply element.

VII. The receiver also rejects the setup request if the MAF limit of itself or its neighbors will be exceeded due to the new setup.

VIII. If suitable, the receiver accepts the setup.

| Octets: 1 | 1 | 1 | 1 | 1 | 2 |
|---|---|---|---|---|---|
| Element ID | Length | MDAOP Set ID | MDAOP Duration | MDAOP Periodicity | MDAOP Offset |

*Figure 3.2 : MDAOP Setup Request Element*

| Octets: 1 | 1 | 1 | 1 | variable |
|---|---|---|---|---|
| Element ID | Length | MDAOP Set ID | Reply Code | Alternate suggested Request element |

*Figure 3.3 : MDAOP Setup Reply Element*

Every MP that has MDA active is required to advertise TX-RX and Interfering times using the MDAOP Advertisements information element. These advertisements are always transmitted in group addressed frames; either in Beacon frames or MDA action frames. The advertised times include:

- TX-RX times report
- Interfering times report

| B0 B7 | B8 B15 | B16 B19 | B20 B23 | | |
|---|---|---|---|---|---|
| Element ID | Length | MDA Access fraction | MDA Access fraction limit | TX-RX times report | Interfering times report |
| Bits: 8 | 8 | 4 | 4 | variable | variable |

*Figure 3.4 : MDAOP Advertisement Element*

An MDAOP set is successfully torn down once both the transmitter and the receiver stop advertising the set in their TX-RX times. Either the transmitter or the receiver may indicate a teardown by transmitting the MDAOP Set Teardown information element to the other communicating end (transmitter or the receiver).
The teardown is assumed successful once the ACK is received, or maximum retry attempts are exceeded.

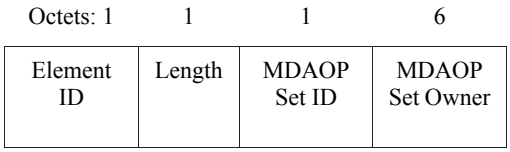


*Figure 3.5 : MDAOP Teardown Element*

The transmitter assumes a successful teardown and stops using or advertising (in TX-RX times report) an MDAOP set if any of the following happens:

- Its MDAOP Set Teardown information element is successfully Acked.
- The maximum retries for the teardown information element it is transmitting are exceeded.
- The receiver's advertisement does not include the MDAOP set
- The receiver is unreachable for greater than a given time

The receiver assumes a successful teardown and stops advertising an MDAOP set if any of the following happens:

- Its MDAOP Set Teardown information element is successfully Acked.
- The maximum retries for the teardown information element it is transmitting are exceeded.
- The transmitter's advertisement does not include the MDAOP set.
- The transmitter is inactive for greater than a specified time

The access behavior for MPs during the Neighborhood MDAOP Times is described as below.

Access by MDAOP owners :

If the MP is the owner of the MDAOP, it attempts to access the channel using CSMA/CA and backoff using MDACWmax, MDACWmin, and MDAIFSN parameters. If the MP successfully captures an MDA TXOP, before the end of its MDAOP, it may transmit until the end of the MDAOP or until a time less than MDA TXOP limit from the beginning of the MDA TXOP, whichever is earlier. The retransmission rules for access in an MDAOP are the same as that of EDCA. Specifically, if there is loss inferred during the MDA TXOP, retransmissions require that a new TXOP be obtained using the MDA access rules in the MDAOP. No MDA TXOPs may cross MDAOP boundaries.

Access by non-owners of MDAOP:

MDA MPs that are not the owner of the current MDAOP can start transmitting after the owner of the MDAOP has finished its transmission.

## 3.2. MDA in ns-2

We implemented IEEE 802.11s Mac in ns-2 as an extension of the IEEE 802.11 Mac. We included in this implementation MDA as an optional feature.

In order to simulate MDA, we also implemented

- a network model for IEEE 802.11s for creating logical links among MPs
- a entity to manage end-to-end flows.

### 3.2.1. MAC Data Structures

We add a new buffer for Management packets, as this was not present in IEEE 802.11 implementation and a data structure that contains Neighbor table, which permits to maintain logical links with neighbors.

Neighbor table contains the following informations about peers of the node :

- Mac Address
- State of the link
- Directionality, which tells which of the two node is responsible for measuring link quality
- Channel used
- Channel precedence
- Rate of logical link
- Packet error rate of logical link
- Signal quality of logical link

In addition to those fields, Neighbor table contains the last MDAOP Advertisement received from peer, as this is necessary, for example, when node evaluates Interfering Times before setting up a MDAOP.

We add data structures to maintain informations about DTIM and synchronization between nodes. This involves an internal timer that elapses every time a DTIM starts and fields with informations about current and next DTIM.

New structures for MDA State were added as well. This involve :

- a buffer for MDA packets : in fact, MDA packets do not pass through link layer queue, but they arrive directly at MAC and are temporarily stored in a buffer with the other packets with the same setid
- a random generator to be used with Slot Selection Procedure (see 3.2.3)
- state variable to indicate if node is actually inside a MDAOP and its role
- a switch table that enable forwarding of MDA packets along the path suggested by MDA Manager (see 3.3)
- Times and Interfering Times structures, used when invoking Slot Selection Procedure and when checking if there are overlapping MDAOPs

Finally, we add data structures for Dynamic Relocation. This structures contain the statistics obtained during transmission and informations about current state.

### 3.2.2. MDAOP Setup Procedure

MDAOP Setup procedure is called when a node wants to reserve resources with its neighbor using MDA. The request is expressed in terms of number of slots and periodicity of MDAOPs inside the DTIM. By expressing a periodicity greater than one, a node can divide slots among more MDAOPs issuing only one Setup Procedure.

Node firstly checks if there are available locations in the DTIM, by analyzing its Times and Neighborhood Times data structures, and then finds groups of contiguous slots. Now node can build up a new data structure which contains offset and length of all available locations. Depending on the slot-selection algorithm used (see 3.2.3.), node chooses one location.

Slot selection is repeated until periodicity is reached. If one of MDAOPs cannot be allocated, Setup Procedure fails.

If all MDAOPs can be allocated, node associates a Mesh-level unique number (setid) to the set of MDAOPs and sends a unicast MDAOP Request message to its peer, where lists MDAOPs previously selected.
MDAOP Request message is sent using CSMA/CA.

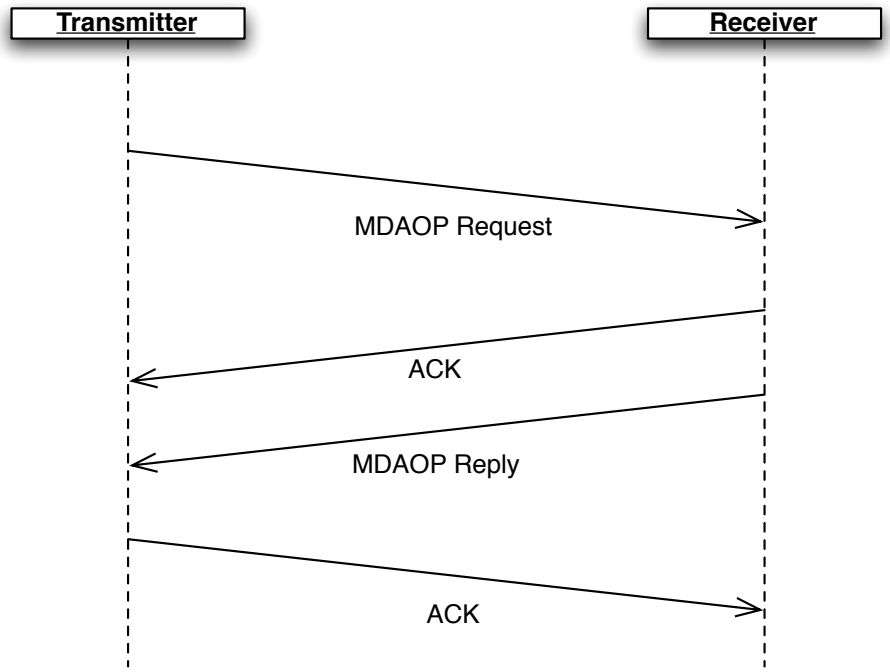


*Figure 3.6 : Reservation through MDA*

A node that receives a MDAOP Request message firstly checks if the request can be accepted or the slots requested are already in use by another MDAOP, so analyzes its Times and Neighborhood Times data structures. Node then sends a unicast MDAOP Reply message to the node that sent the request and specifies a code :

- Request accepted
- Request rejected
- Request rejected, with suggestion

Request can be accepted only if all slots for all MDAOPs are available.

If node finds out that slots specified by the neighbor are already in use, starts a Slot Selection procedure using its own data structures. If it finds a set of suitable locations for all MDAOPs, sets the code of MDAOP Reply message to “Request rejected, with suggestion”, and includes in the message the list of location suggested; if node cannot find locations to suggest, sets the code of MDAOP Reply message to “Request rejected”. Node then sends a MDAOP Advertisement broadcast message.

A node that receives a MDAOP Reply message from a node, to whom has previously sent a MDAOP Request message, checks the code.
If request has been rejected, MDAOP Setup procedure fails.
If code means “Request rejected, with suggestion”, node checks if the suggested slots can be used or these are already in use by another MDAOP, so analyzes its Times and Neighborhood Times data structures. If locations are available, node sends a new MDAOP Request message.
If code means “Request accepted”, node updates its internal state and sends a MDAOP Advertisement broadcast message.

MDAOP request and Reply messages are sent using DCF, so they could be subjected to loss. We can encounter the following problems in a MDAOP Setup Procedure :

1. A MDAOP Request message is lost
2. An ack for a MDAOP Request message is lost
3. A MDAOP Reply message is lost
4. An ack for a MDAOP Reply message is lost

Cases 1 and 3 lead only to retransmission of lost message. Case 2 and 4, instead, need to be considered.
This has been accomplished by tracing state of the MDAOP Setup Procedure for both transmitter and receiver.

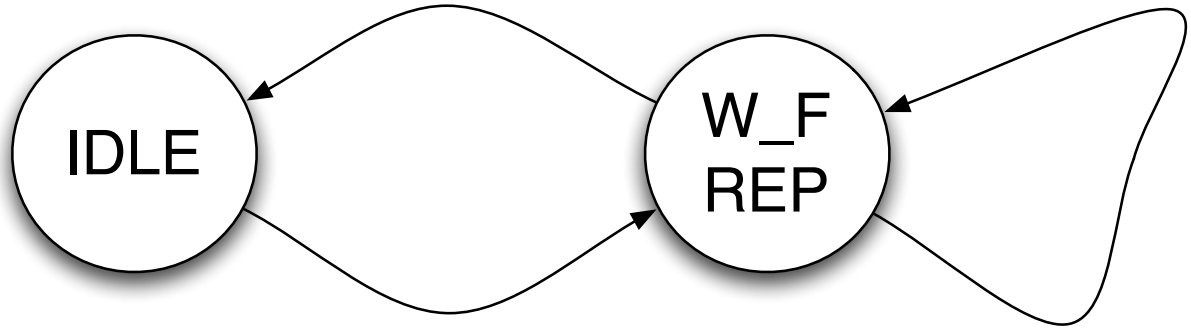


*Figure 3.7 : Finite State Machine for MDAOP Setup Procedure, TX*

States :

- IDLE : it is the initial state in which nodes are before starting a MDAOP Setup Procedure
- WFREP : node has sent a MDAOP Request message and is waiting for a MDAOP Reply message.

State transitions :

- From IDLE to WFREP : Node has received an ack for the MDAOP Request message sent. That indicates that neighbor has certainly received the request.
- From WFREP to IDLE : Node has received the MDAOP Reply message from neighbor.

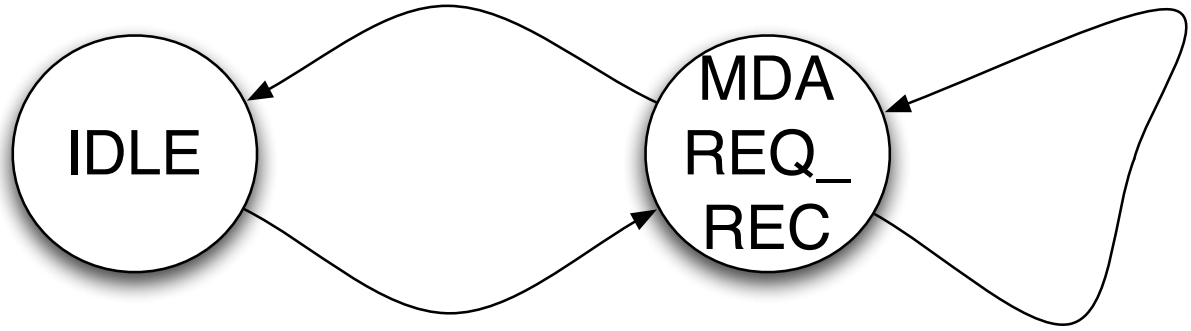


*Figure 3.8 : Finite State Machine for MDAOP Setup Procedure, RX*

States :

- IDLE : it is the initial state in which nodes are before starting a MDAOP Setup Procedure

- MDAREQ_REC : node has received a MDAOP Request message and are waiting for a MDAOP Reply message

State transitions :

- From IDLE to MDAREQ_REC : Node has received a MDAOP Request message and it is ready evaluate if it can be accepted.
- From MDAREQ_REC to IDLE : Node has received an ack for the MDAOP Reply message sent. That indicates that neighbor has certainly received the reply.

As concurrent MDAOP Setup Procedures are allowed, states are distinctly traced for every setid.

Consider case 2 :

if an ack for a MDAOP Request message is lost, transmitter resends the request and receiver can misunderstand the message and start a new MDAOP Setup Procedure. To avoid this situation, nodes must be configured to ignore MDAOP Request messages already received. When a node receives a MDAOP Request Message it firstly checks its current state for specified setid. If state is IDLE, message is accepted. If state is MDAREQ_REC, that probably means that the ack previously sent for the MDAOP Request message has been lost. In this case, node deletes MDA State for specified setid and sends a MDAOP Reply message again.

Consider case 4 :

if an ack for a MDAOP Reply message is lost, receiver resends the reply and transmitter can misunderstand the message. To avoid this situation, nodes must be configured to ignore MDAOP Reply messages already received. When a node receives a MDAOP Reply Message it firstly checks its current state for specified setid. If state is IDLE, message is accepted. If state is WFREP, that probably means that the ack previously sent for the MDAOP Reply message has been lost. In this case, node ignores the message.

### 3.2.3. Slot Selection Procedure

MDAOP Setup procedure is called when a node wants to select a set of contiguous slots to use for a new MDAOP.
Node firstly builds up a bit vector with length equal to number of slots inside a DTIM and sets all bits corresponding to locations already in use, according to its Times and Neighborhood Times data structures. Node also sets bits that cannot be used because they are blacklisted (see Dynamic Relocation) or have been chosen by a MDAOP Setup Procedure still in progress. This last check permits to avoid that two concurrent MDAOP Setup procedures at the same node may select overlapped locations.

At this point node can enumerate locations that can satisfy the request. Procedure ends if no locations can be found, otherwise node builds up a data structure in which are listed all locations found and their length.
Node selects one of these locations according to slot selection algorithm :

- Random allocation : node randomly selects a location among those listed in data structure
- Best Fit : node selects the location that leaves less slots unused
- Worst Fit : node selects the location that leaves more slots unused

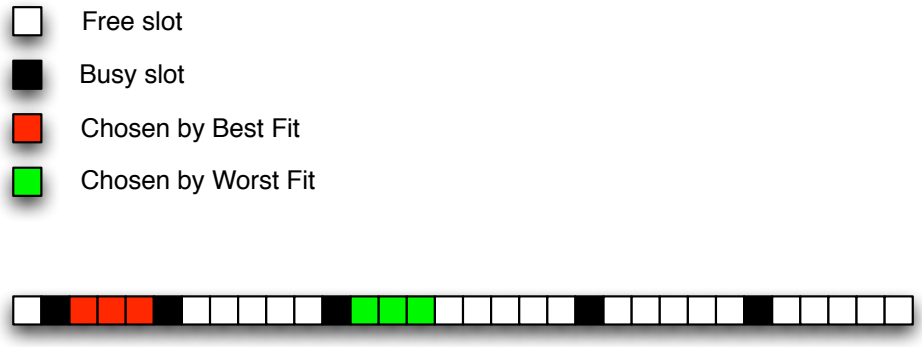


*Figure 3.9 : Example of a selection of a three-slot location*

### 3.2.4. MDAOP Teardown Procedure

MDAOP Teardown procedure is called when a node wants to deallocate resources obtained through MDAOP Setup procedure. Node specifies setid, so the procedure involves all MDAOPs that have the same setid.
Node updates its internal state and sends a unicast MDAOP Teardown message to the peer specifying the setid.

A node that receives a MDAOP Teardown message checks if it is actually the receiver for specified setid and then updates its internal state.
After a MDAOP Teardown message is either sent or received, a broadcast MDAOP Advertisement is transmitted.

The procedure of updating internal state involves the same steps for transmitter and receiver :

- Delete state for each MDAOP of the setid
- Update internal timer to not preempt for just deleted MDAOPs

Transmitter also does the following :

- Drops all remaining packets associated with setid
- Checks if the MDAOP Teardown procedure happens inside one of the just deleted MDAOPs and in that case releases current MDAOP before sending MDAOP Teardown message

### 3.2.5. MDAOP Advertisement

MDAOP Advertisement messages are used to inform neighbors of internal MDA state. These messages are broadcasted every time MDA state changes at a node (Setup, Teardown ...) or sent periodically (if node is configured to do that).
The node that sends a MDAOP Advertisement includes two elements :

- Times in which node has a MDAOP as transmitter or receiver
- Times in which a neighbor of the node has a MDAOP as transmitter or receiver

These elements are specified in terms of setid and location inside DTIM.

A node that receives a MDAOP Advertisement firstly compares the list of setid included in the message with the list in the last MDAOP Advertisement message (if any) received from that neighbor. If there is no change, it means that nothing has happened in the meantime and this messages can be ignored. Secondly, node checks if there are any setid not included in previous MDAOP Advertisement message and for which node is neither transmitter nor receiver. These elements will be inserted in Neighborhood Times vector. Then, node checks if there are any setid included in previous MDAOP Advertisement message that are not present in the list of the current MDAOP Advertisement. These elements will be erased from Neighborhood Times vector. Finally, node puts the new MDAOP Advertisement message in its buffer.

### 3.2.6. Medium access during MDAOPs

At the beginning of every DTIM each node sets a timer for every MDAOP that it knows. When a timer expires, node behaves differently depending on it is the owner of the MDAOP or not.

Node is the owner :
Node firstly checks if it is transmitter or receiver for that MDAOP. If it is the transmitter, puts the current non-MDA packet that has to be transmitted, if any, in a temporary buffer. Secondly compares timestamps of MDA packets, if any, with current time and drops those that expired. Now node can send MDA packets until the MDAOP expires. Figure shows an example of transmission during MDAOP.

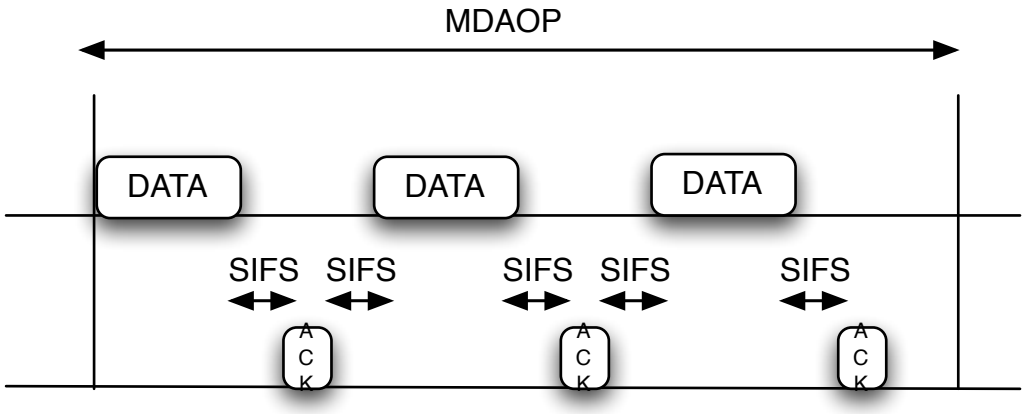


*Figure 3.10 : Access during MDAOP*

If medium is found idle node transmits and then waits for an ACK after a SIFS. If an ACK is received node starts a new transmission after a SIFS. If ACK is not received, node can retransmit it, if configured to do so. If medium is found busy, node acts using normal DCF with contention window between MDACWinMin and MDACWinMax.

If node is the receiver for current MDAOP, it can only sends Control packets during MDAOP (ACK and CTS). So it cannot initiate any transmission, but can receive packet from any other node.

Node is not the owner:
node can only sends Control packets during MDAOP (ACK and CTS). So it cannot initiate any transmission, but can receive packet from any other node. This is to avoid situations in which they won't ack transmissions from nodes that are not aware of current MDAOP (legacy stations or nodes that missed some MDAOP Advertisements).

Sensing medium with MDA active is slightly different from sensing during normal DCF. In fact this should prevent situations in which a node starts a transmission that could eventually end after a MDAOP starts. The following steps are processed by a node that wants to start a transmission in a 802.11s wireless Mesh network :

1. Senses medium
2. Checks NAV
3. Checks if the entire transmission of a packet (RTS-CTS-DATA-ACK) can be accomplished before next MDAOP

The following steps are in addition processed by a node that wants to start a transmission in a 802.11s wireless Mesh network during a MDAOP :

1. If node is not the owner or is the receiver of MDAOP, it cannot start any transmission but can only send ACK and CTS (Control frames)
2. If node is the transmitter of MDAOP, checks if the entire transmission of a packet (RTS-CTS-DATA-ACK) can be accomplished before the end of MDAOP

### 3.2.7. Network Model

An IEEE 802.11s network is modeled as a weighted oriented graph G(V , E), where vertex $v_i$ represents the i-th IEEE 802.11s node, and an edge (or link) $e_{ij}$ exists between nodes i and j if the node i can transmit directly its neighbor j . The weight of any edge is equal to the physical transmission rate of the link. Graph *G* is also called connectivity graph. Let *N* be the number of nodes in the network and *L* the number of links in the network.

The connectivity graph of an IEEE 802.11s with nodes arranged in a 3 × 2 grid is illustrated in figure.

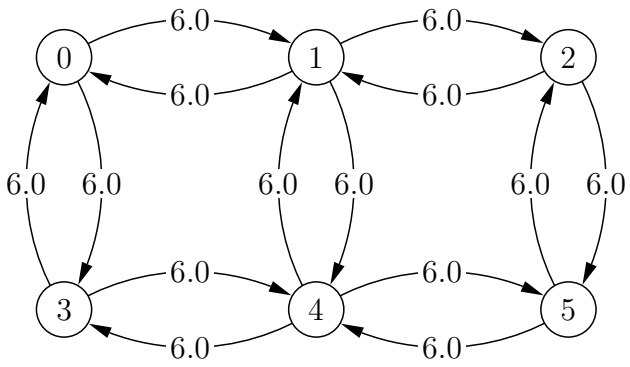


*Figure 3.11 : Connectivity graph of an IEEE 802.11s*

The connectivity matrix *C* is derived from the connectivity graph, so that c(i, j ) contains the physical transmission rate of node i towards node j if there is a link between these nodes, otherwise it is 0. The connectivity matrix induced by the connectivity graph in figure is the following :

$$\mathbf{C} = \begin{pmatrix} 0 & 6.0 & 0 & 6.0 & 0 & 0 \\ 6.0 & 0 & 6.0 & 0 & 6.0 & 0 \\ 0 & 6.0 & 0 & 0 & 0 & 6.0 \\ 6.0 & 0 & 0 & 0 & 6.0 & 0 \\ 0 & 6.0 & 0 & 6.0 & 0 & 6.0 \\ 0 & 0 & 6.0 & 0 & 6.0 & 0 \end{pmatrix}$$

In reality each node periodically advertises in the network the list of its neighbors and the physical transmission rates that are currently used to reach them. These advertisement messages can be conveyed, for instance, piggybacked into the periodical DTIM beacons used for time synchronization among IEEE 802.11s nodes or into the routing messages used to maintain the tree topology to/from the Root node.

In the analysis, the connectivity graph is derived as follows. Each node is placed in flat X-Y grid. The physical transmission rate is determined depending on the Euclidean distance between any two nodes. To this aim, the Signal-to-Noise-Ratio ($SNR_{sr}$) perceived by the receiver node r when the sender node s is the only transmitting node in the network is computed first:

$$SNR_{sr} = \frac{1}{P_N} \cdot \frac{P_t}{d_{sr}^{\gamma}} \cdot \left( \frac{\lambda}{4\ \pi} \right)^2$$

where λ is the wavelength (in Hz), $P_t$ is the transmitting power (in W), which is assumed to be the same for all nodes, $d_{sr}$ is the Euclidean distance between nodes s and r, γ is the path loss exponent, which is in the range between 2 and 6 depending on the physical environment, and $P_N$ is the thermal noise power (in W).
The rate of the link from node s to node r is then set to the highest rate supported such that:

$$SNR_{sr} \geq SNR_{\min,R} + P_G$$

where $SNR_{min,R}$ is the receiver sensitivity for rate $R$ and $P_G$ is a guard factor, both in dB.

In the analysis, a MAC frame transmitted at rate $R$ from node s to node r is assumed to be received correctly by the r if:

$$\frac{\frac{P_t}{d_{sr}^{\gamma}} \cdot \left(\frac{\lambda}{4\ \pi}\right)^2}{P_N + \sum_{k \in \mathcal{T}} \frac{P_t}{d_{kr}^{\gamma}} \cdot \left(\frac{\lambda}{4\ \pi}\right)^2} \geq SNR_{\min,R}$$

where T is the set of nodes transmitting MAC frames (partially) overlapping in time with the MAC frame from node s.

Optionally, shadowing can be taken into account by decreasing the power of the signal received by a random factor, drawn from a Gaussian distribution with zero mean and standard deviation equal to $sigma_{dB}$ .

Furthermore, any node detects a busy channel while sensing the medium before transmitting whenever the sum of the received power if greater than an Energy Detection (ED) threshold, even though none of the MAC frames contributing to it can be actually decoded. In the analysis, setting the ED threshold greatly impact on the network dynamics. On the one hand, setting this value too high (e.g. higher than the highest receiver sensitivity) increases the spatial reuse, since nodes farther from the sending node do not detect busy channel and can transmit at the same time. However, this creates “hidden nodes” in the network, which potentially leads to a high number of retransmissions of MAC frames. On the other hand, setting the ED threshold to as small value can remove the hidden node problem, but can create “unaware nodes”, i.e. nodes that are unnecessarily prevented from transmitting their MAC frames. The IEEE 802.11 standard specifies the ED threshold to be set to 20 dB higher than the higher receiver sensitivity (i.e. hidden nodes can exist), since its use is mostly intended to avoid interference from other sources than the IEEE 802.11 network itself, such as co-existing wireless networks co-existing in the same

frequency bands. However, it is found that commercial implementations set the ED threshold to a much lower value so as to avoid hidden nodes as well.

This channel model does not take into account multi-path fading, i.e. fast variations of the channel quality perceived by the receiver node that happen on a time-scale smaller than the MAC frame transmission duration. In other words, provided that the SNR requirement is met, all MAC frames are assumed to be correctly received in a deterministic manner.

### 3.2.8. MDA Manager

We introduced in our implementation a new entity that has a complete view of the entire mesh. MDA Manager is unique in the Mesh network and allows to start end-to-end flows using MDA features. MDA Manager is responsible for the following tasks :

- Path computation
- Invoking MDAOP Setup Procedure on node along the path

A node that wishes to start an end-to-end flow firstly contacts MDA Manager specifying TSPEC. When the MDA Manager receives the flow admission message, it decides whether the flow can be admitted or not. In the former case, i.e. the flow is admitted, a path using MDA is opened from source to destination. On the other hand, i.e. the flow is rejected, the source node is not allowed to transmit the incoming packets. The source node can later re-request the admission of the traffic flow with the same TSPEC in the hope that network resources are freed meanwhile, or it can immediately request the admission of the traffic flow with a less demanding (= lower nominal transmission rate) TSPEC.

MDAManager can compute path in two different ways :

1. Using Dijkstra algorithm with a metric that is proportional to the number of slot needed on a link
2. Using Ford algorithm with a sort of admission control, verifying DTIM Utilization in terms of busy slots on each link of candidate paths.

Subsection 3.2.9. describes how MDA Manager obtains the number of slots required on each link from TSPEC.

After MDAManager has obtained a suitable path, invokes MDAOP Setup procedure on the first node, by specifying :

- Number of slots on the link
- Periodicity (i.e. how many MDAOPs reserve in the DTIM)
- Destination node
- Flow id

Then waits until receives a notification from node. MDAManager can receive three kind of notifications :

1. MDAOP Setup procedure has been successful
2. MDAOP Setup procedure has failed
3. A Relocation has been accomplished

If node tells that MDAOP Setup procedure has been successful, MDAManager invokes MDAOP Setup procedure on next node, if any. If the node that sent the notification is the last node of the flow, MDAManager schedules starting of application at the beginning of next DTIM.
If node tells that MDAOP Setup procedure has failed, MDAManager is responsible of invoking MDAOP Teardown procedure on intermediate node of the flow.

If a relocation procedure has been accomplished, MDAManager checks status code of notification. If code is SUCCESS, MDAManager does nothing, otherwise invokes MDAOP Teardown procedure on all node of the flow

MDA Manager is also invoked to terminate existing flows. When a node starts an end-to-end flow using MDA features, specifies to MDA Manager finishing time as well.
When flow has to be terminated, MDA Manager is responsible to invoke MDAOP Teardown Procedure on every hop.

### 3.2.9. TSPEC Conversion function

TSPEC is specified as :

- Packet Size : *S* (bytes)
- Rate : *R* (bit/s)
- Maximum Delay : *D* (s)

So application is going to send packets of *S* bytes at a rate *R* bit/s and will expect a maximum application-to-MAC delay of *D* seconds.

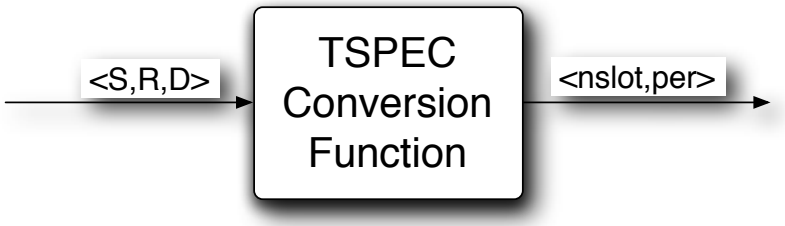


*Figure 3.12 : TSPEC Conversion Function*

We obtain that packet inter-arrival time is *S*8/R* seconds.

Assuming DTIM duration in seconds (called *DTIM*) greater than D, we obtain that the frequency of MDAOPs in a DTIM in order to satisfy delay constraints is *ceil(DTIM/D)*, and that the number of packets to be sent in each DTIM is *ceil(DTIM/(S*8/R))*.

We denote *ceil(DTIM/D)* as *NPER* and *ceil(DTIM/(S*8/R))* as *NPKT*.

So in each period, except for the last one, the application will send *ceil(NPKT/NPER) packets*. In the last period *NPKT- ceil(NPKT/NPER)*(NPER-1)* packets will be sent.

The number of slots during DTIM to reserve for the application depends on transmission rate, packet size, number of packets and ack duration. Given :

- *pkt_txtime* : time to transmit a packet
- *ack_txtime* : time to transmit an ack

- *interpkt_time* : time between two consecutive packet transmissions (which is a SIFS during a MDAOP)
- *slot_dur* : single slot duration (32 us)

We have that to successfully transmit *NPKT* packets, the number of slots needed is : *ceil((NPKT * (pkt_txtime + ack_txtime + (2 * interpkt_time))) / slot_dur)*

TSPEC Conversion function takes TSPEC as input and gives the total number of slot to reserve on the logical link and the frequency of the MDAOP inside DTIM as output.

We propose an example. Given the following TSPEC :

- Packet Size = 1000 bytes
- Rate = 2 Mbit/s
- Max Delay = 0.2 s

Given a DTIM duration of 0.32 seconds, we obtain :

- Inter-arrival Time = 0.004 s
- NPER = 2
- NPKT = 80

So 80 packets, divided in 2 groups of 40, will be sent during DTIM.
Given that total packet length is 1054 bytes (including MAC Header), we obtain the following transmission times :

- Packet at 54 Mbps = 0.000176 s
- ACK = 0.000044 s
- SIFS = 0.000016 s

Supposing transmission rate 54 Mbps, we need to reserve 630 slots in total (315 + 315) in every DTIM.

## 3.3. Dynamic Relocation

After extensive simulation we understood that MDA alone does not perform well. We found the following three reasons :

1. Partial overlapping of MDAOPs
2. Interference with other MDAOPs which are not considered in Neighborhood Times
3. DTIM fragmentation

### 3.3.1. Partial overlapping of MDAOPs

Consider the following scenario :

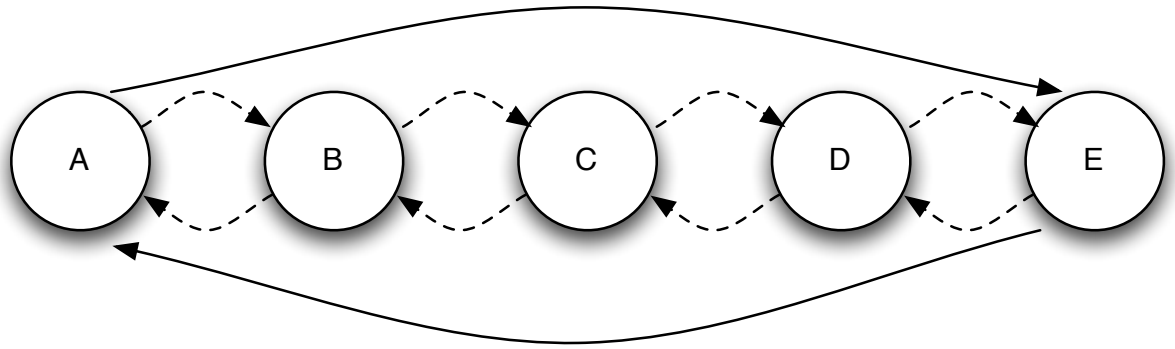


*Figure 3.13 : Chain scenario with two flows*

Each node can establish a logical link (dashed line) only with its next hop.
At time *t* two application are started from node A and E to node E and A respectively.
A and E start their setup by asking B and D a location for a new MDAOP.
Suppose that A obtains its location before E, B starts asking C for a new location. C accepts the request because slots required by B are available. At the same time D accepts the request from E.

In this scenario there is the following problem : C does not know anything about the concurrent setup between E and D, so it can potentially accept a set of slots that are also asked from E to D.

### 3.3.2. Interference with other MDAOPs

Consider the following scenario :

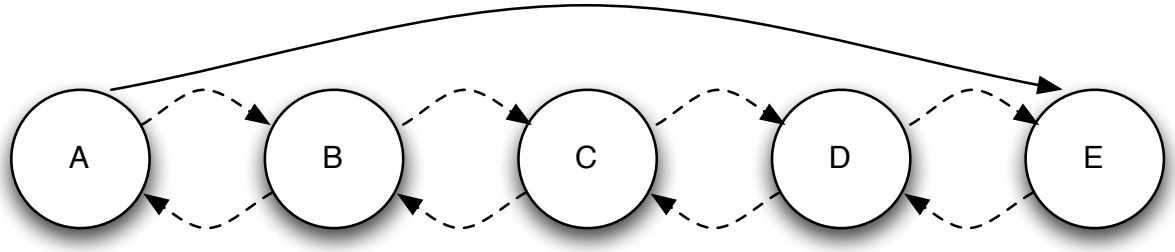


*Figure 3.14 : Chain scenario with one flow*

Each node can establish a logical link (dashed line) only with its next hop.
At time *t* one application is started from node A to E, so four MDAOP Setup Procedures are carried on.
Suppose that flow is accepted by the network, we observe four groups of MDAOPs.
MDA standard allows overlapping locations between two nodes at least three hops away (eg. A and D).

Simulation results, using a SINR-based channel model, demonstrated that MDA packets transmitted from A during overlapping MDAOPs may collide as signal received at B is not always sufficient to decode the packet, while packet from D are successfully received from E.

This problem is not generated by MDA, but it is an intrinsic characteristic of the wireless channel model used during simulations. MDA is not capable of avoiding this situation itself.

A first possible solution could be enlarging Neighborhood Times diameter, but this also reduce spatial reuse as it avoids using location that could possibly work well. In this case prevention is not the best thing to do.

### 3.3.3. DTIM fragmentation

The algorithm used to select available locations is based on probabilistic choose of a set of contiguous slots.
Simulation results demonstrated that Maximum DTIM Utilization measured at the end of run reaches a peak of 78% in the previous scenarios.
We also evaluate performance using different types of slot selection algorithms, like Best Fit and Random Fit, in section 4.1.4.

### 3.3.4. Dynamic Relocation procedure

The first two problems discussed before lead to the same effect, which is increase of packet loss and consequent reduction of throughput.
*Partial overlapping of MDAOPs* cannot be avoided because each node has a different view of DTIM utilization, *Interference with other MDAOPs* can be avoided with an inefficient solution that dramatically reduces spatial reuse.

The solution for the two problems proposed here is called **Dynamic Relocation**.
Dynamic Relocation is mechanism that permits to overcome MDA limits by reallocating MDAOPs basing on statistics collected during transmission times.

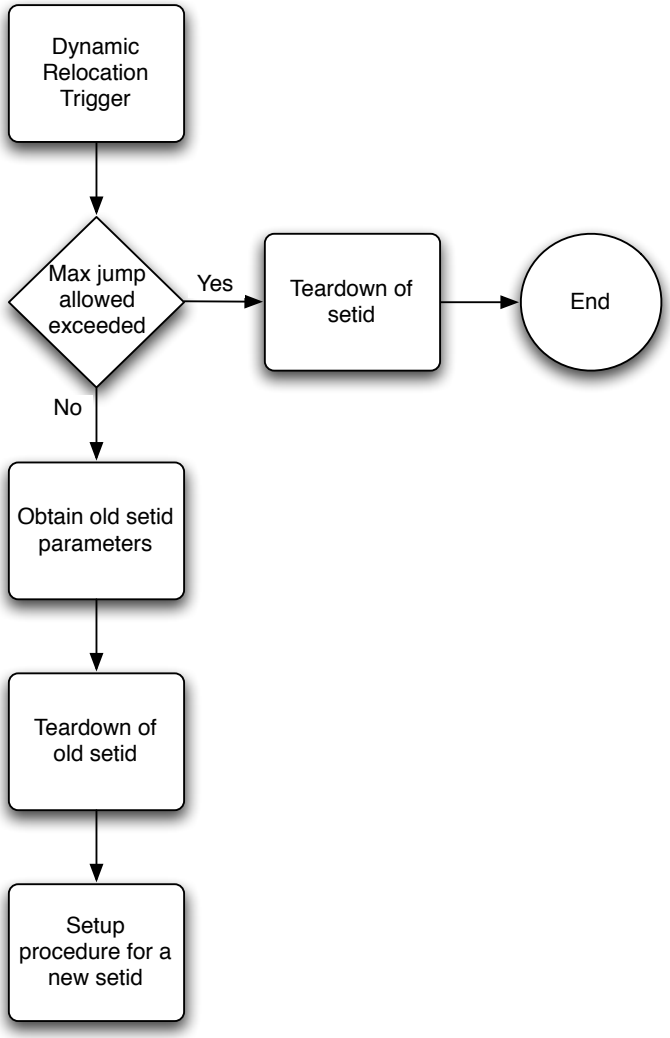


*Figure 3.15 : Dynamic Relocation Procedure*

Each setid has a finite number of possible relocations (called **jumps**), after that the setid will be torn down. If the maximum number of jumps has not been reached, the procedure obtains setid parameters : total number of slots and periodicity.
Now old setid can be safely torn down and a setup procedure for a new setid is started.

There are two different ways in which dynamic relocation procedure can be started, as this mechanism is used to solve two different problems.

The first one activates relocation after a MDAOP Advertisement has been received.

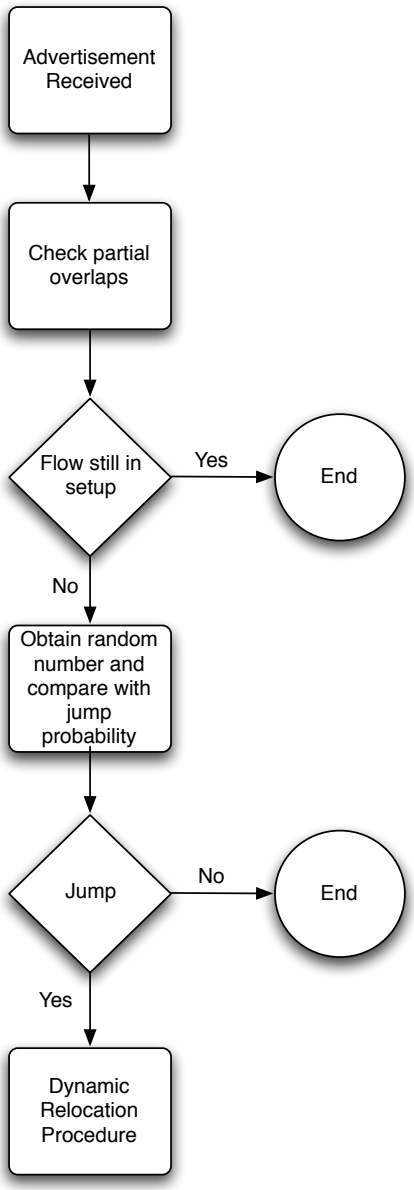


*Figure 3.16 : Trigger based on MDAOP Advertisement receipt*

It looks for partial overlaps between MDAOPs, for which the node is the transmitter, and the just updated Neighborhood Times.

If a overlap is detected, node asks MDAManager if flow associated with candidate setid is still in setup. If flow is still in setup, node does not try to relocate it, but waits for next MDAOP Advertisement to see if flow setup has finished. This is done to avoid relocation of a flow that has not been started yet.

If application has already started to send its packets, node obtains a random number and compare that with the probability to jump of the flow. This probability varies with time in such a way that older setid have less probability to relocate respect to younger ones.

The second trigger relies on estimating packet loss during MDAOP :

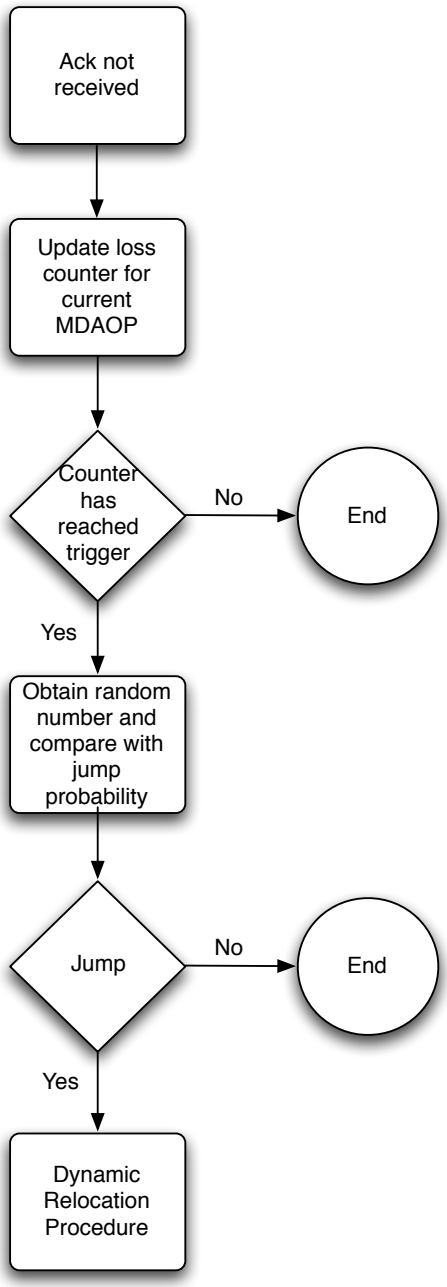


*Figure 3.17 : Trigger based on estimating packet loss during MDAOP*

Every time an ack is not received, loss counter is updated. Because a setid can contain multiple MDAOPs that use different locations, loss counter is implemented as a multi-counter. Every time the multi-counter has to be updated, node checks in which MDAOP the packet was transmitted and updates the counter associated with that MDAOP. We did not choose to use a single counter for the whole setid because in some cases that would be misleading.

For example, if there is a setid with five different MDAOPs and node experiments packet loss only during the first, with a single counter the trigger could not be reached, while with five different counters node can surely determine that there is something wrong with the first MDAOP and relocate the entire setid.

If an ack is received during a MDAOP, multi-counter is update with the same rule as in case of loss, until the specified farthest value away from the trigger.

If trigger is reached node obtains a random number and compare that with the probability to jump of the flow. This probability varies with time in such a way that older setid have less probability to relocate respect to younger ones.
After the comparison, node resets multi-counter with the same rule as in case of loss.

# 4. Performance analysis

## 4.1. Chain

We first compare MDA and DCF in 5-node chain scenario as proposed in [4].
Several independent replications for each simulation scenario have been run, according to the method of independent replications [5]. Mean values are then estimated, along with 95% confidence intervals [6], which are not reported in figures whenever negligible.

We list the values of most important parameters :

| Parameter | Value |
|---|---|
| DTIM Duration | 0.32 s |
| DTIM unused by MDA | 1% |
| Advertisement Periodicity | 0.96 s |
| MDA Packet Life | 0.32 s |
| Minimum prob to jump | 20% |
| Maximum prob to jump | 80% |
| Step to increase prob to jump | 1% |
| Loss lower bound | -10 |
| Loss upper bound | 10 |
| Step to increase loss counter | 1 |
| Step to decrease loss counter | -2 |
| Maximum number of jump | 10 |

*Parameter configuration*

Nodes are positioned in such a way that logical links can be established only among nodes at one-hop distance and with a rate set to 12 Mbps.

### 4.1.1. First Scenario : High ED Range

In this scenario we start a CBR micro-flow with constant rate from node 0 to node 4 every 0.5 seconds.
We set Energy Detection Range to two hops. In this way nodes always consider medium busy if a node two hops away is transmitting. With this setting we reduce interference problems caused by hidden terminal, but we limit spatial reuse.

Metric of interest are :

- End to End Throughput
- DTIM Utilization
- Rejected micro-flows.

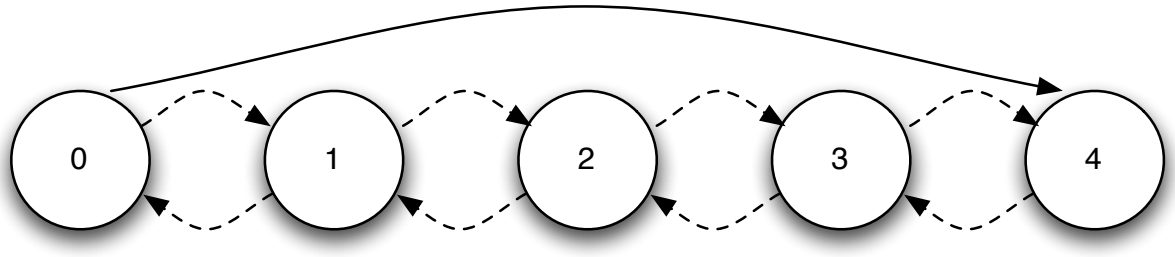


*Figure 4.1 : Chain Topology with single macro-flow*

The number of micro-flows is expressed as offered load, obtained by multiplying the number of micro-flows activated by their constant rate.

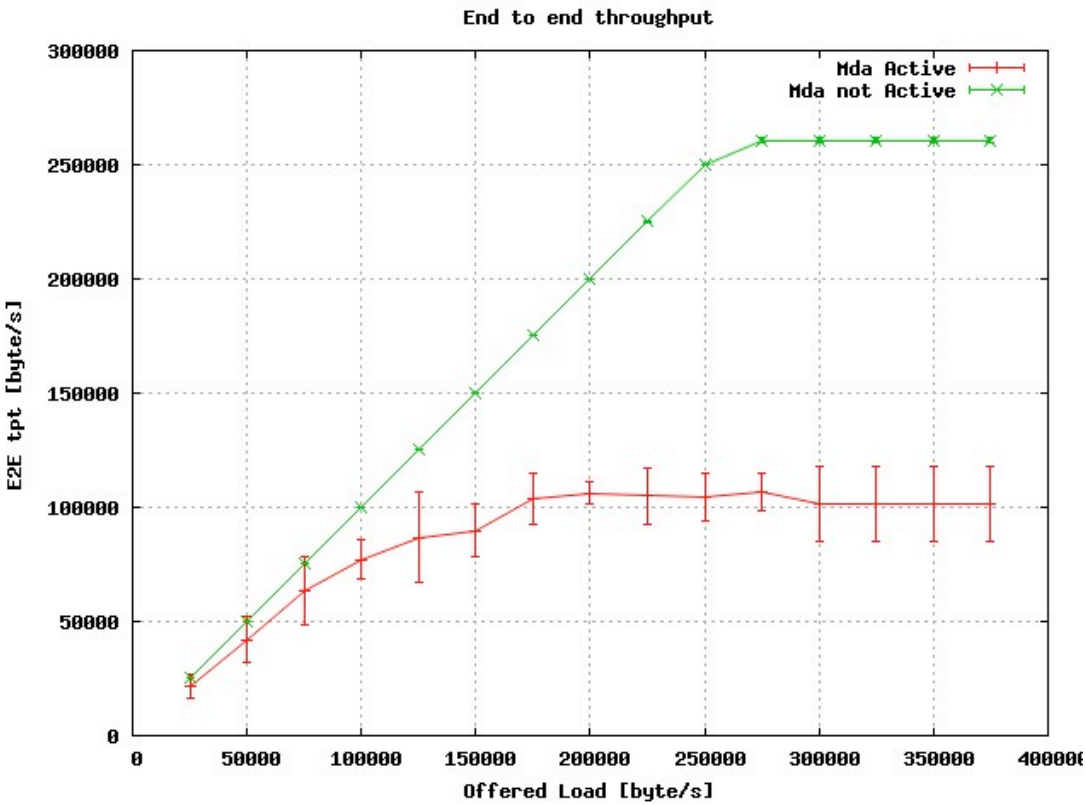


*Figure 4.2 : End to end throughput*

This scenario permits DCF to obtain maximum throughput until saturation (about 250 Kbyte/s). MDA behaves badly because of partial overlapping, interference and DTIM fragmentation, discussed in section 3.4.

DTIM Utilization is collected at the end of simulations and represents the view that each node has of DTIM.

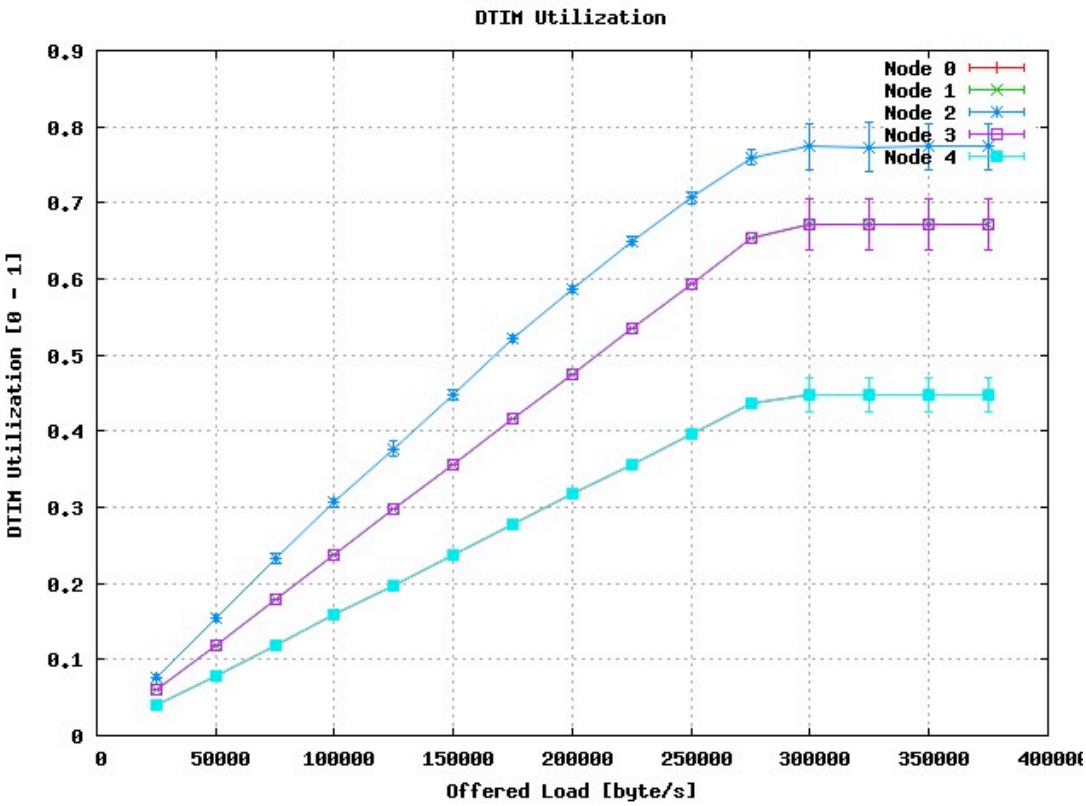


*Figure 4.3 : DTIM Utilization*

Node 2 has a more consistent view because it stays in the middle of the chain. As offered load increases, DTIM utilization increase until it reaches about 78% (saturation) at node 2. Not all DTIM is used because of fragmentation.

Flow Rejected metric is updated every time a flow is rejected at MDAManager and represents the fraction of flows that are rejected.

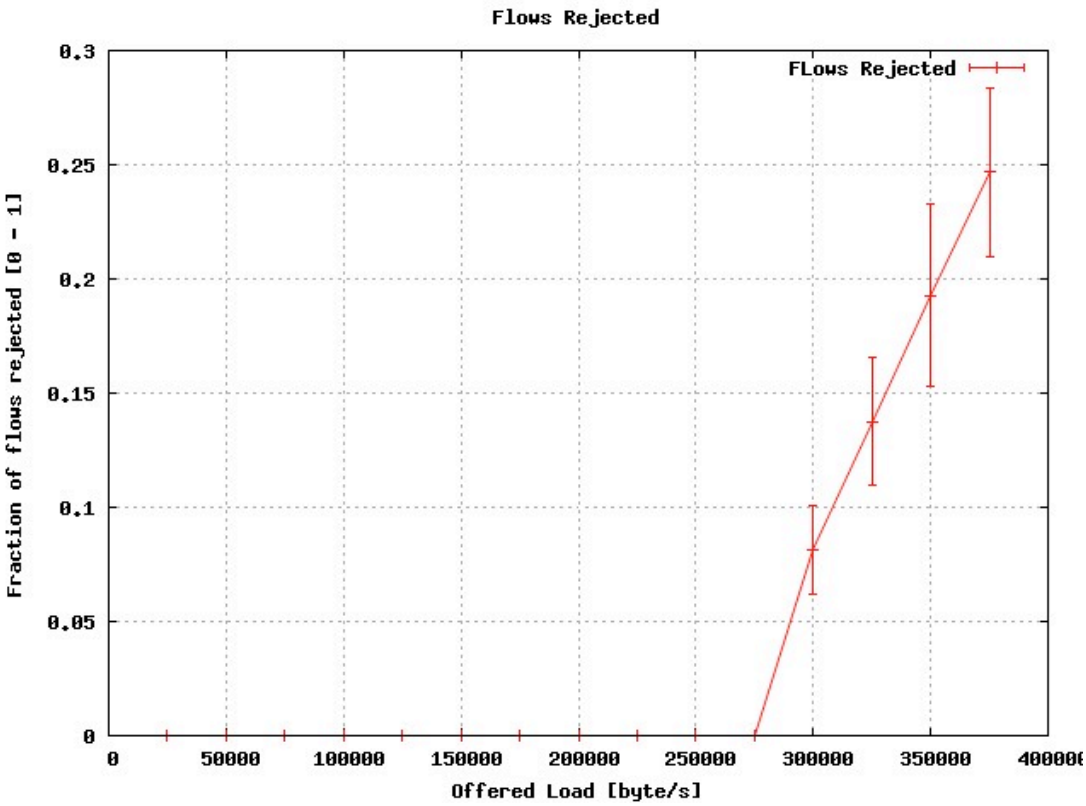


*Figure 4.4 : Flow Rejected*

DTIM Utilization and Flow Rejected metrics are strictly correlated as when DTIM Utilization reaches saturation, MDA Manager starts to reject flows. Throughput, instead, reaches saturation before 275 Kbyte/s because of overlapping MDAOPs.

### 4.1.2. Second Scenario : Reduced ED Range

In this scenario we compare MDA and DCF when Energy Detection Range is reduced to the distance of one hop. We want to evaluate the impact of the increasing interference. We use the same traffic configuration used in the first scenario.

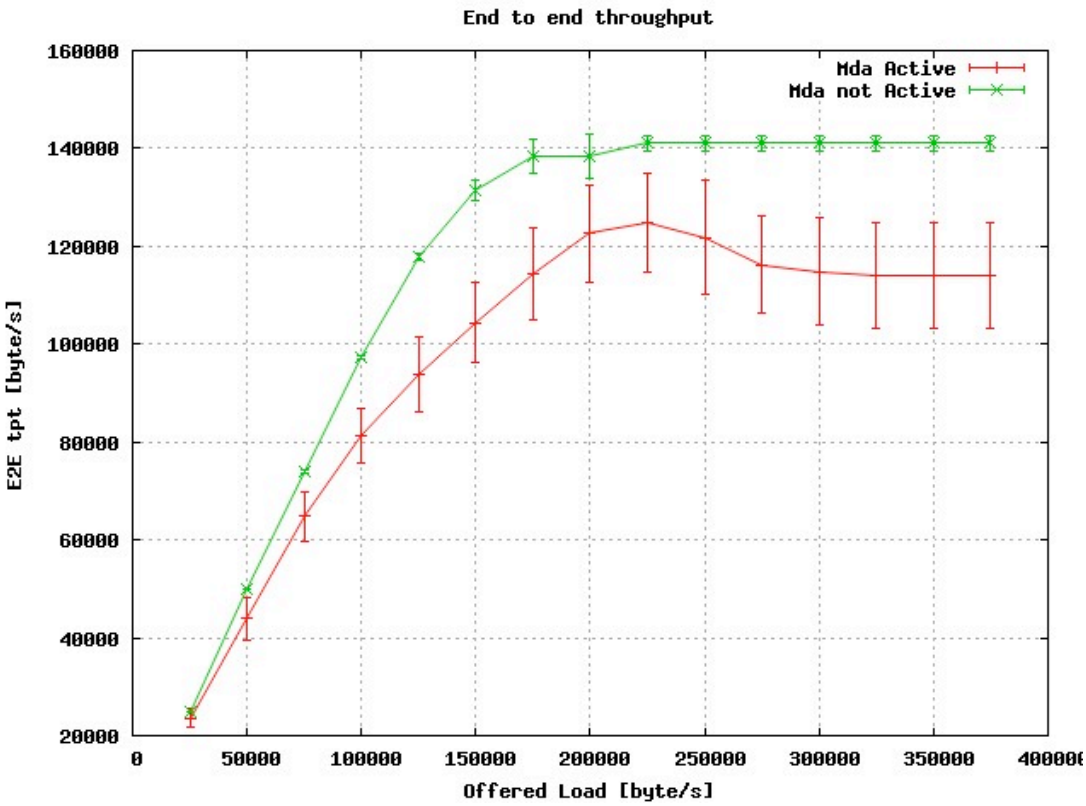


*Figure 4.5 : End to end throughput*

The difference between DCF and MDA is now only about 20 Kbyte/s when saturation has been reached, instead of 150 Kbyte/s obtained in the first scenario.
This is due to problems of DCF, as it suffers from hidden terminal.

In this particular network configuration, interference problems can be resolved by increasing Energy Detection Range. That is the reason because DCF could behave well in the first scenario.
That is to say that increasing Energy Detection Range reduces spatial reuse, especially in more complex scenarios.

In this particular network configuration, MDA does not suffer of interference as DCF does. Lower throughput is given by other kind of phenomena that do not depend on Energy Detection Range, like overlapping MDAOPs. So MDA can exploit spatial reuse better than DCF.

### 4.1.3. Third scenario : Dynamic Relocation

In this scenario we compare DCF and MDA with dynamic relocation enabled.
We use the same traffic and network configuration used in the first scenario. Energy Detection Range is set to the distance of one hop. Throughput is collected when system has reached steady state, so no flows attempt to relocate anymore.

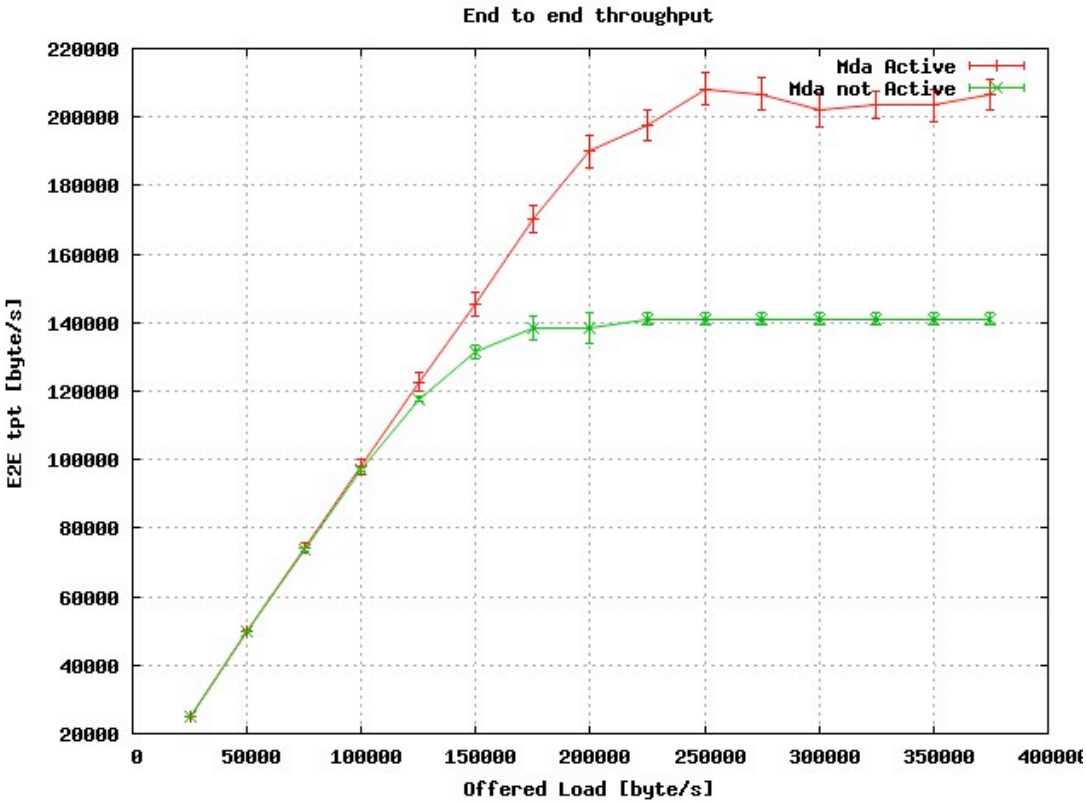


*Figure 4.6 : End to end throughput*

MDA with dynamic relocation boosts up throughput. That demonstrates that this feature permits active flows to reach a state in which they do not interfere with others.

Even though packet loss is not present in steady state, throughput obtained by MDA with dynamic relocation is not at the top, which is obtained, in this particular network configuration, by DCF when Energy Detection Range is set to the distance of two hops. That happens because DTIM is not totally used.

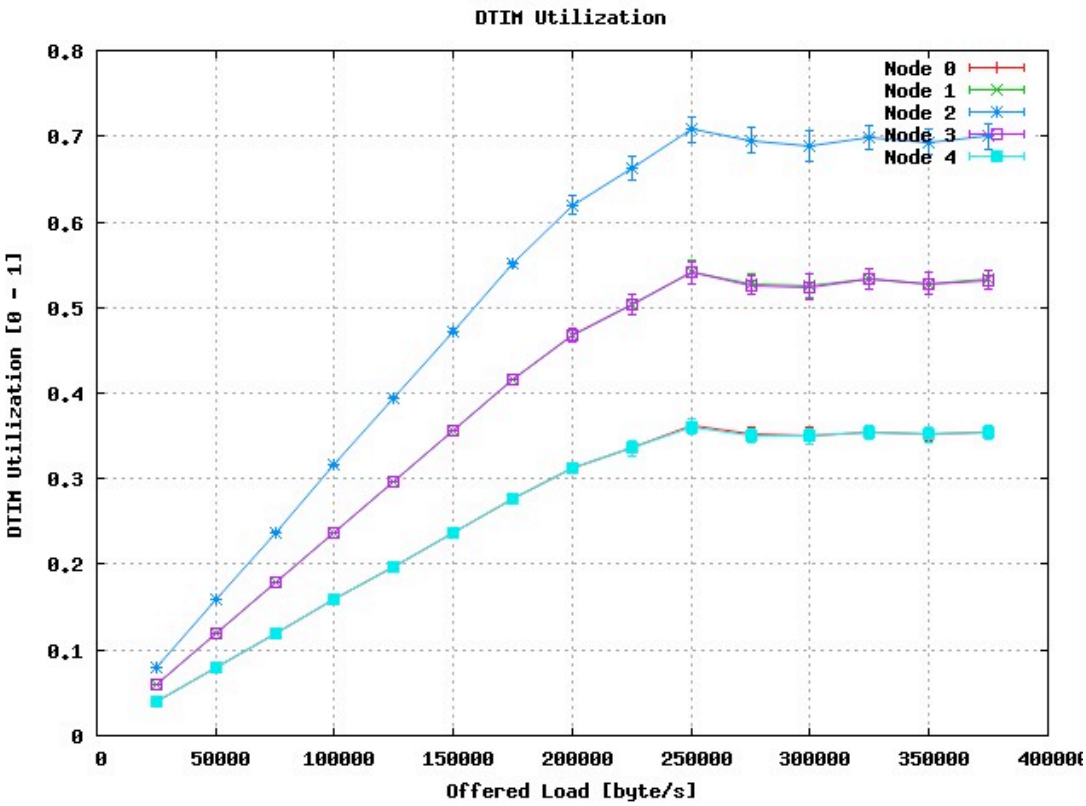


*Figure 4.7 : DTIM Utilization*

DTIM Utilization metric shows that fragmentation is greater when dynamic relocation is active. In fact, MDA alone obtained a maximum utilization of about 80%, while now is 70%. This is probably due to the fact that Slot Selection Procedure is invoked more in case of dynamic relocation active.

Now we consider what happens during stale state by examining the evolution of relocations until steady state has been reached.

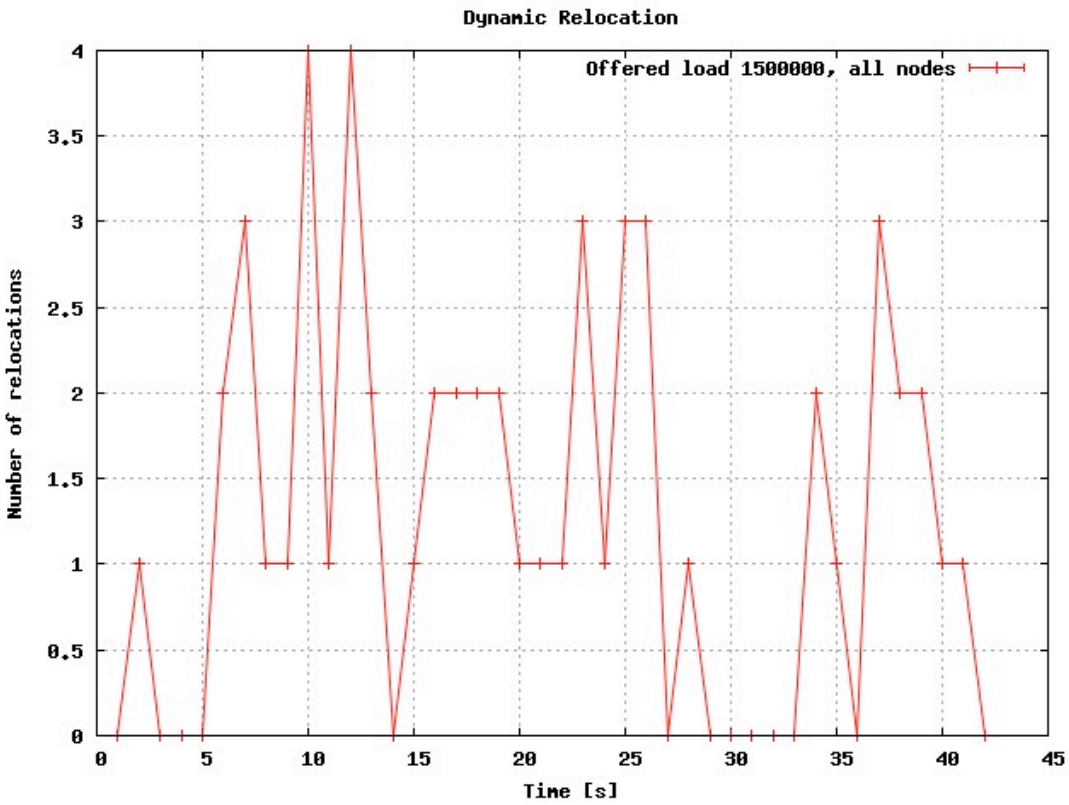


*Figure 4.8 : Total number of relocations accomplished*

Micro-flows are started from time 1 to time 7 every 0.5 seconds. Most of relocations are accomplished immediately after 7 seconds. After 42 seconds network has reached steady state.

### 4.1.4. Fourth scenario : Slot Selection Algorithms

We implemented Best Fit and Worst Fit algorithms to evaluate maximum DTIM Utilization.

Using the same network configuration of previous scenario, we start an increasing number of MDA micro-flows in three different cases :

1. Using Random Slot Selection
2. Using Best Fit Slot Selection
3. Using Worst Fit Slot Selection

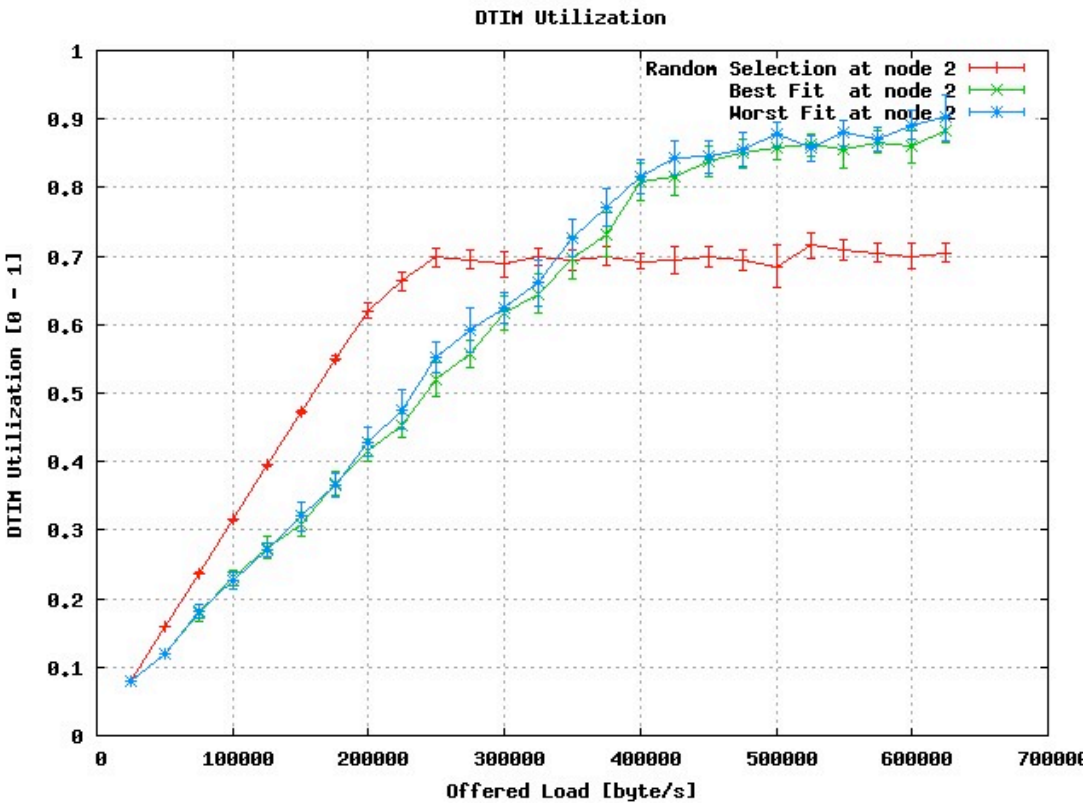


*Figure 4.9 : DTIM Utilization*

DTIM Utilization metric, taken at node 2, shows that Best Fit and Worst Fit can reach a DTIM Utilization peak of 90% in a 5-node chain. It also shows that at lower rates Random Allocation permits to accept more flows.

In fact, with BF or WF there is an increase in the number of jumps made by all flows, as there are more chances to have overlapping MDAOPs. Consequently, more flows are dropped during stale state because they have reached the maximum number of jumps allowed, which is set to only 10 in this scenario.

Because of that, BF and WF need more offered load to reach saturation.

We will evaluate BF with maximum number of jumps allowed set to a higher value in a different topology.

### 4.1.5. Fifth scenario : Two macro-flows.

In this scenario we start a CBR micro-flow with constant rate from node 0 to 4 and from node 4 to 0 every 0.5 seconds.

We want to evaluate how MDA behaves with two different sources.

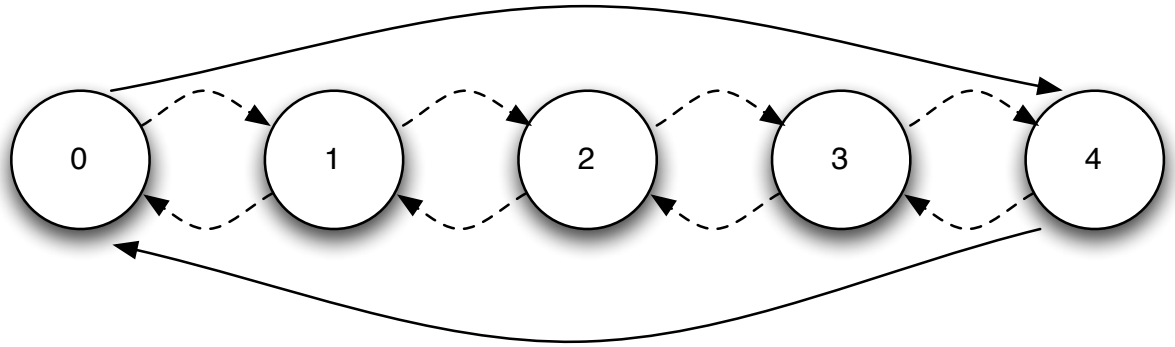


*Figure 4.10 : Chain Topology with two flows*

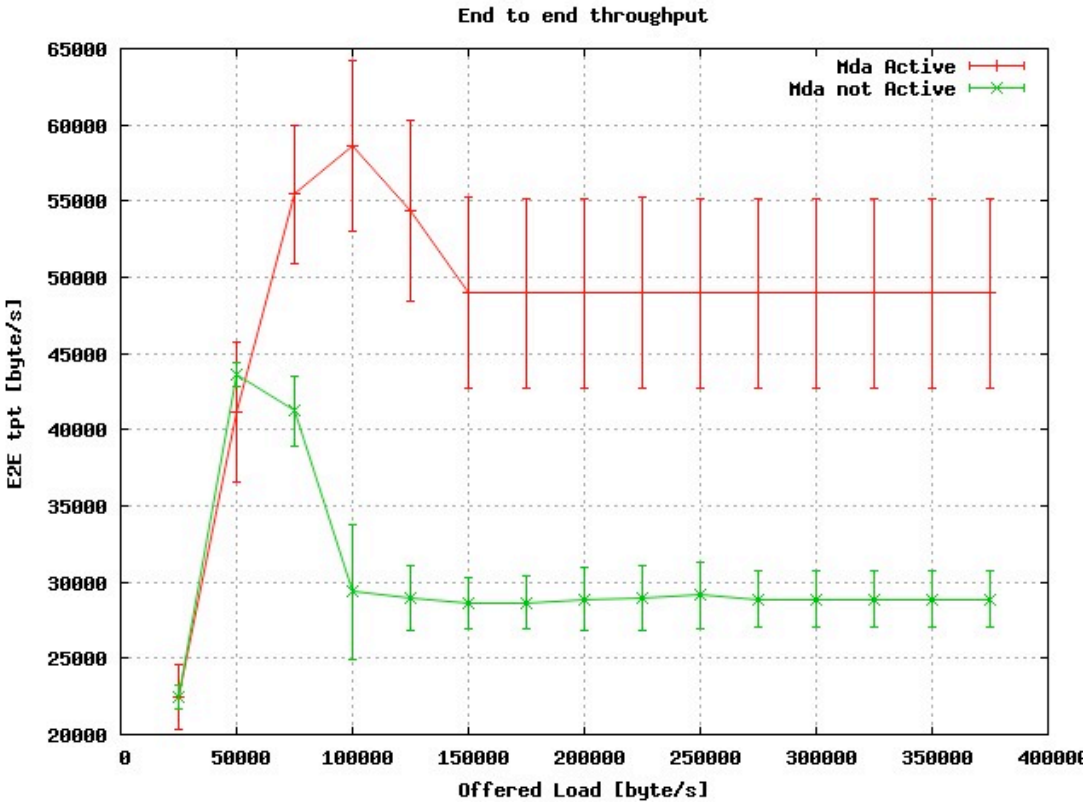


*Figure 4.11 : End to end throughput of flow from node 4 to 0*

MDA performs better than DCF, which suffer from interference. If we compare these results with those of second scenario, we can see that DCF looses a lot in terms of throughput : in second scenario, throughput reached 140 Kbyte/s; now there are two

flows with less than 30 Kbyte/s. On the other hand, MDA passes from a flow at 115 KByte/s to two flows at 49 KByte/s.

We repeated the same scenario enabling Dynamic Relocation.

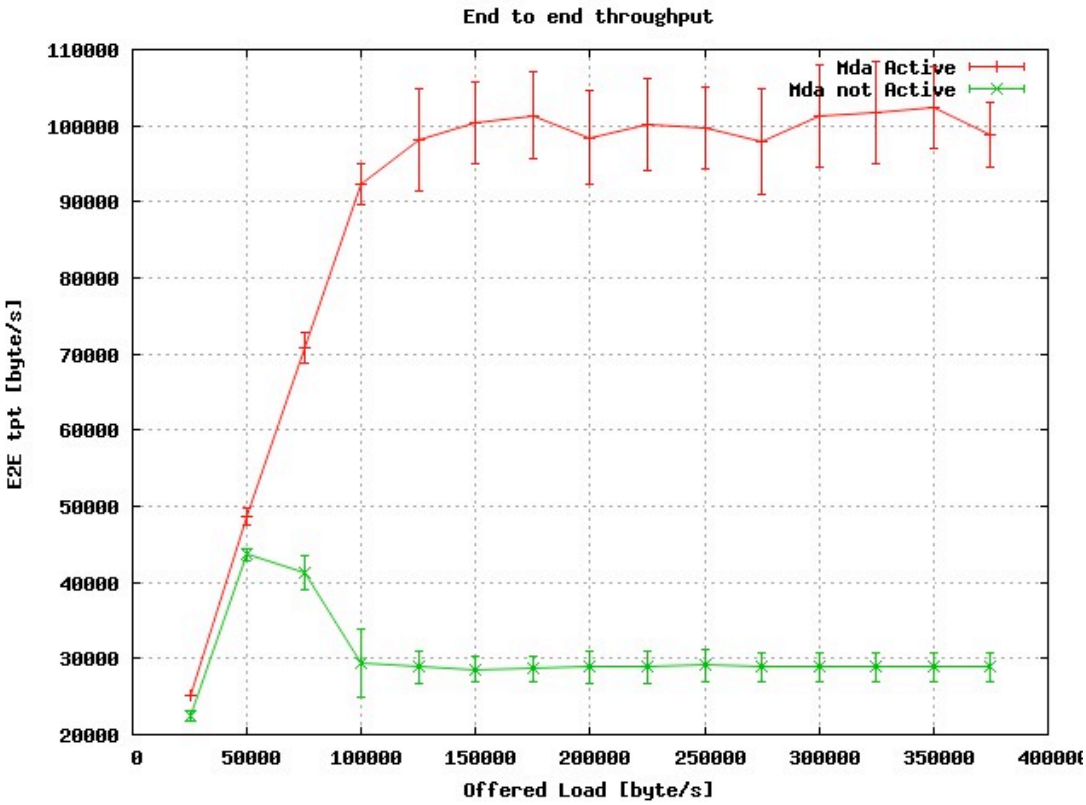


*Figure 4.12 : End to end throughput of flow from node 4 to 0*

We can observe that in third scenario throughput reached about 200 KByte/s and now there are two flows with 100 KByte/s.

We can assume that performance of MDA with Dynamic Relocation enabled does not depend on number of sources and throughput is only limited by DTIM fragmentation.

## 4.2. Grid

We evaluate MDA in a 5x5 grid to understand the limits of this mechanism in a more complex topology.

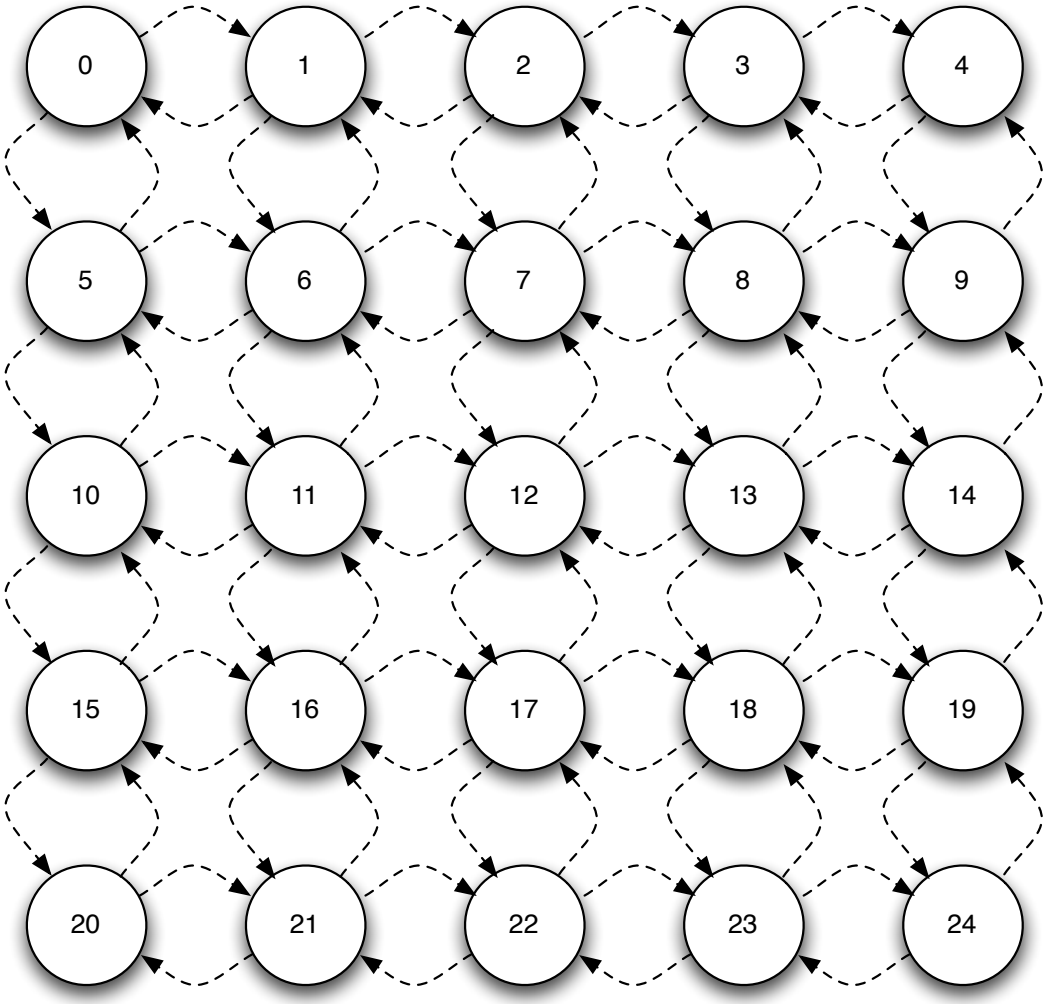


*Figure 4.13 : Grid Topology*

Nodes are positioned in such a way that logical links can be established only among nodes at one-hop distance and with a rate set to 12 Mbps.

We first compare throughput obtained with MDA and DCF as following :

1. MDA with Random Allocation
2. DCF
3. DCF with Admission Control

Table lists most important parameters used for simulations :

| Parameter | Value |
|---|---|
| DTIM Duration | 0.032 s |
| DTIM unused by MDA | 1% |
| Advertisement Periodicity | 0.16 s |
| MDA Packet Life | 0.032 s |
| Minimum prob to jump | 20% |
| Maximum prob to jump | 80% |
| Step to increase prob to jump | 1% |
| Loss lower bound | -50 |
| Loss upper bound | 50 |
| Step to increase loss counter | 1 |
| Step to decrease loss counter | -10 |

*Parameter configuration*

### 4.2.1. First scenario : Three macro-flows.

Traffic macro-flows are started from 0 to 4, 10 to 14 and 20 to 24 at different times. Macro-flow from 10 to 14 is started before the others.

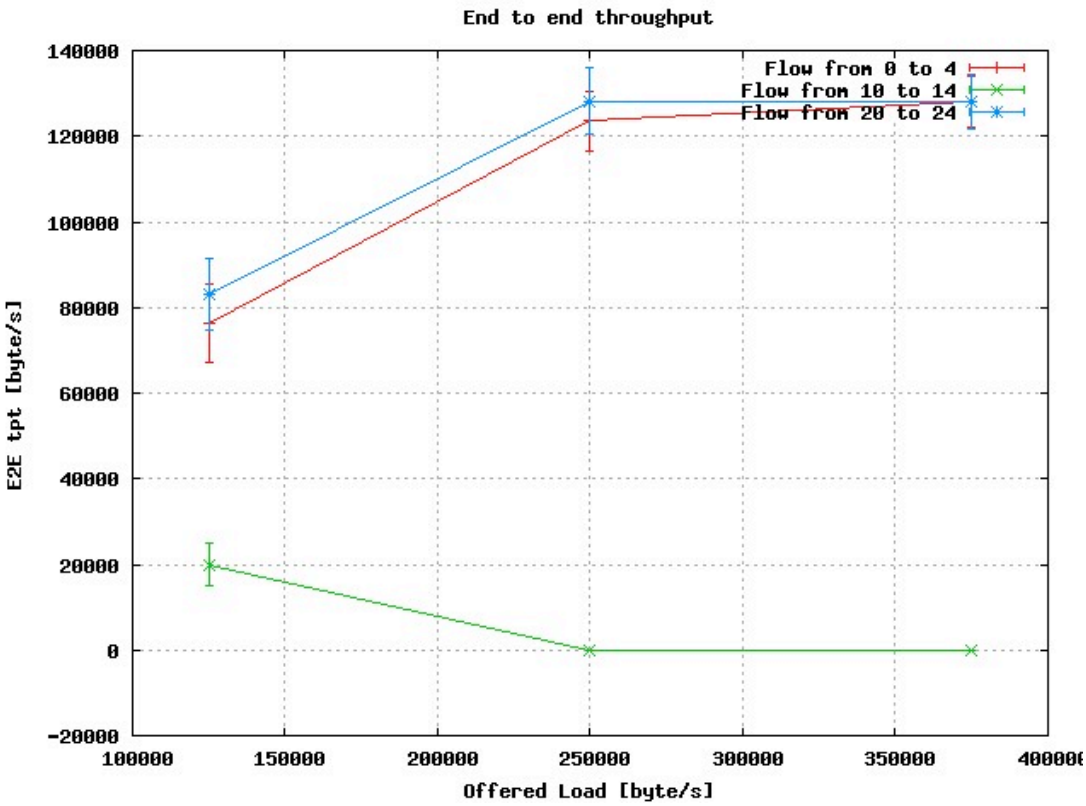


*Figure 4.14 : End to end throughput with MDA and Dynamic Relocation*

Macro-flow from 10 to 14 is able to send only at low rate, even if it is started before the others, so it should have less chances to jump. This happens because it suffers more from interference and maximum probability to jump is set to 80, so it is not impossible for that flow to jump. Once that the threshold has been reached, flow jumps and comes back to minimum probability to jump and from that moment starts to jump until it reaches the maximum number of jumps allowed and it is dropped.

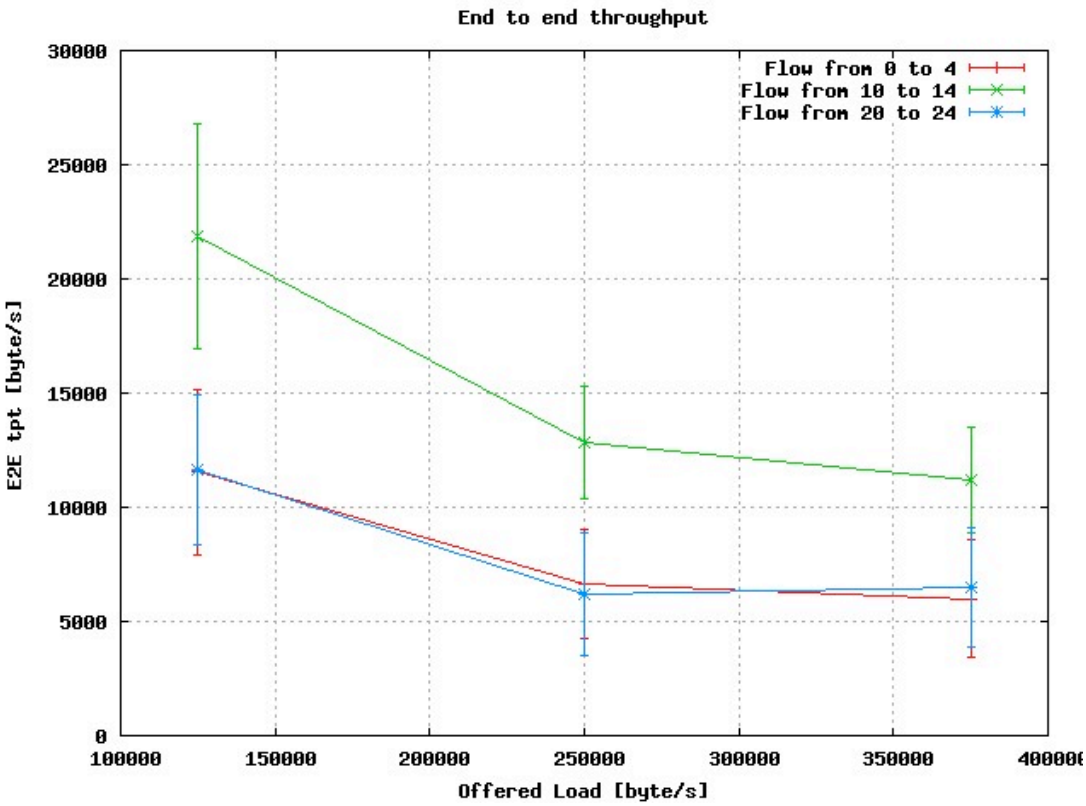


*Figure 4.15 : End to end throughput with DCF*

DCF does not perform well in the grid scenario, as it cannot overcome from interference. Macro-flow from 10 to 14 performs better than the others as when they start it is injecting lots of packets.

Finally, we simulated DCF with Admission Control, in the sense that we do not start all micro-flows, but only a subset of those. The exact number of micro-flows to start is the same that MDA Manager accepts during simulations with MDA. We want to find out if MDA is really necessary in the grid, or DCF can be used reducing the offered load.

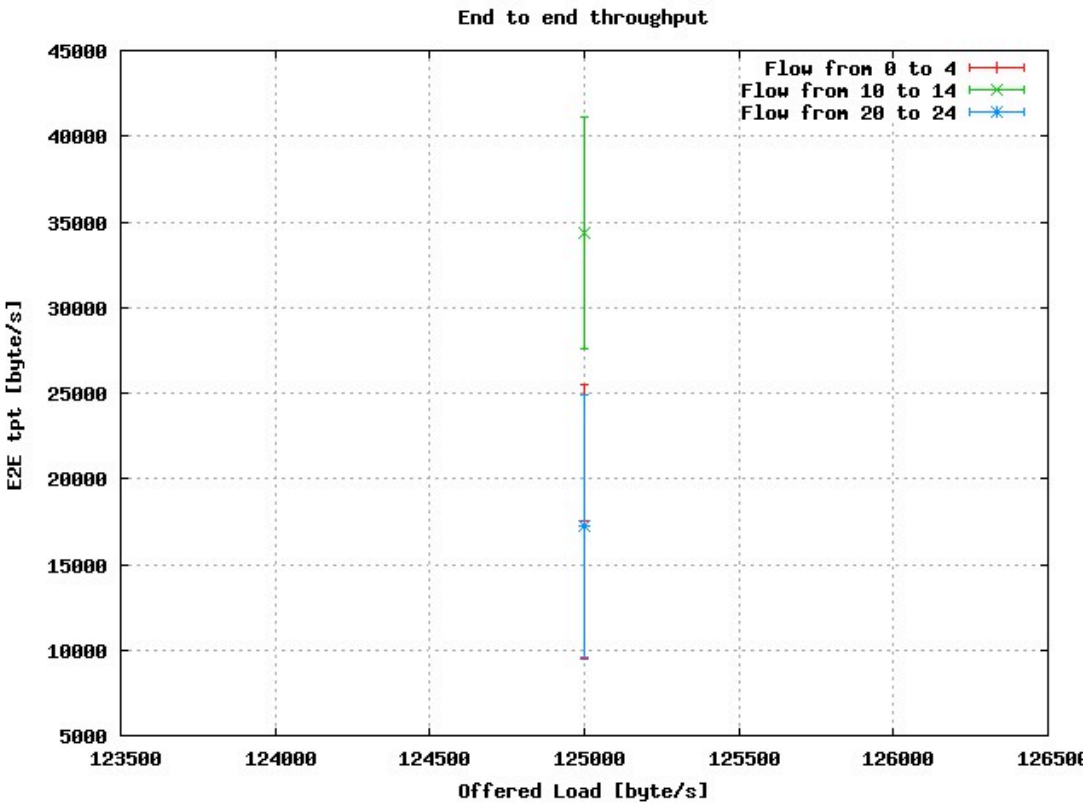


*Figure 4.16 : End to end throughput*

Throughput of every macro-flow is more than that obtained with simple DCF, especially for macro-flow from 10 to 14, but total throughput (obtained as the sum of the throughput of all macro-flows) is still lower than that obtained with MDA.

Figure shows total throughput obtained in different cases.

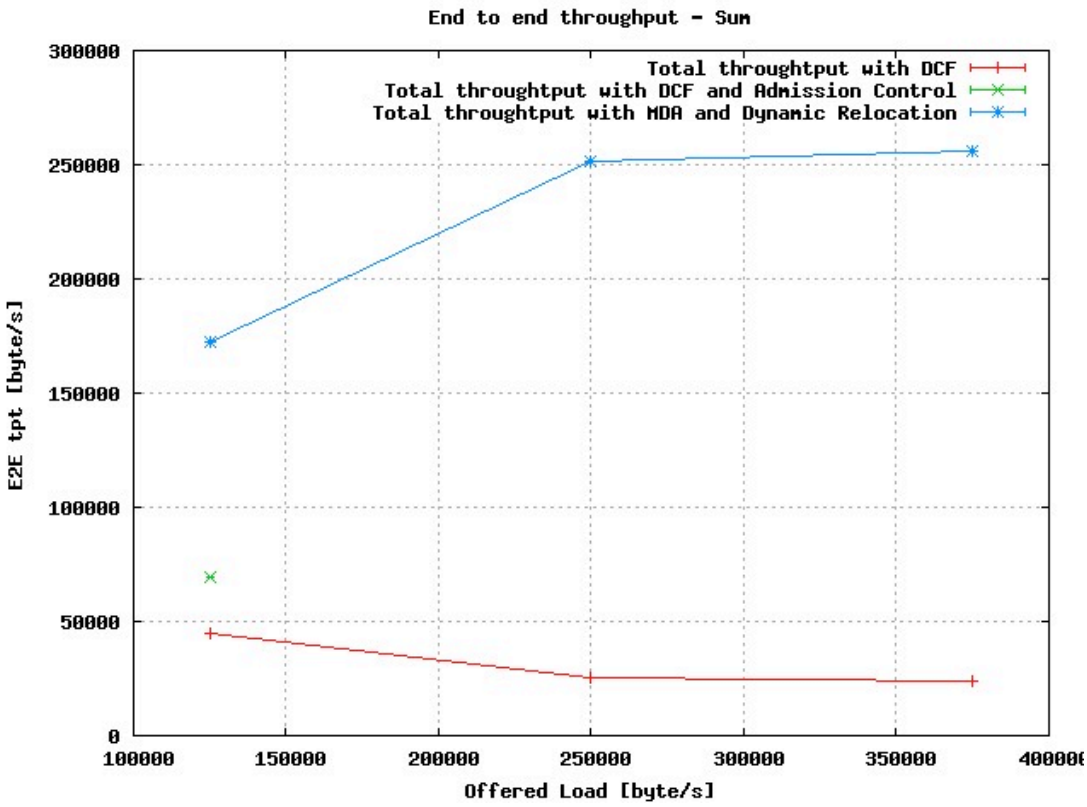


*Figure 4.17 : Total E2E throughput for different cases*

### 4.2.1. Second scenario : Five macro-flows.

Now we focus simulation on evaluating spatial reuse that can be obtained with MDA. We have five macro-flows from nodes 0, 5, 10, 15 and 20 to nodes 4, 9, 14 and 24 respectively. Central macro-flow start before external macro-flows.

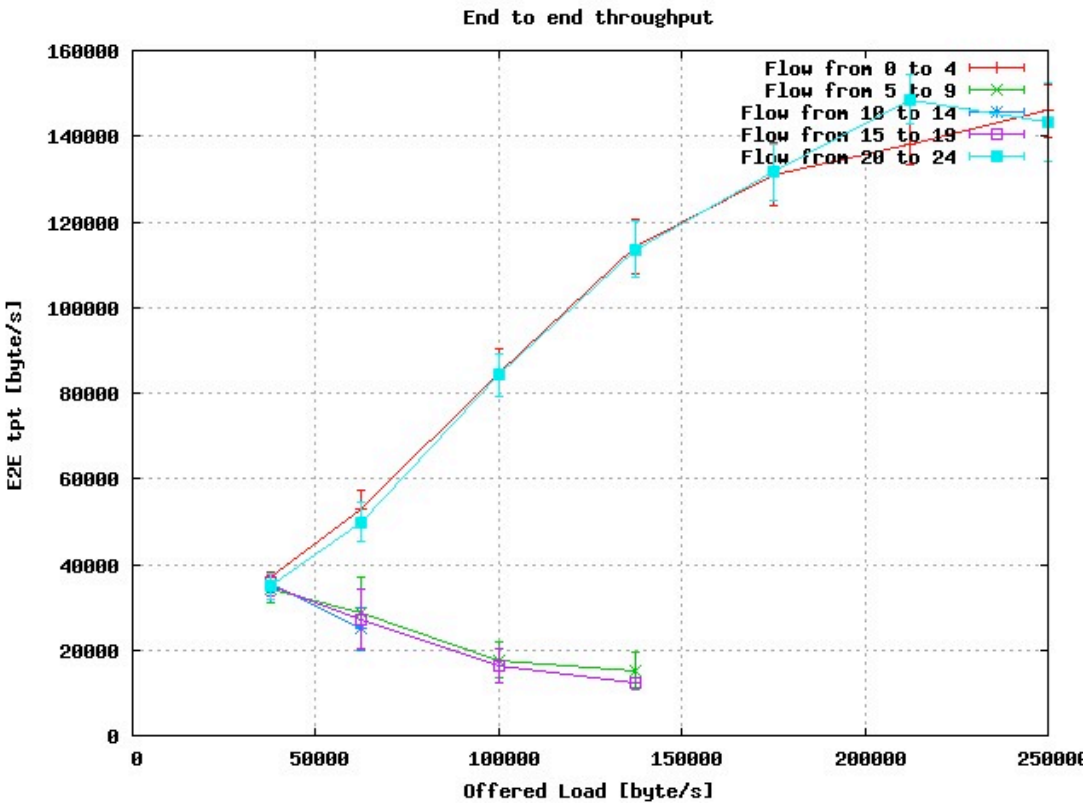


*Figure 4.18 : End to end throughput*

We can see that until offered load is lower than 75000 all flows are accepted, between 75000 and 140000 central flow is rejected and then only external flows are admitted.

Now we compare total throughput obtained in this scenario with total throughput obtained by a single flow in chain scenario, as we want to evaluate spatial reuse.

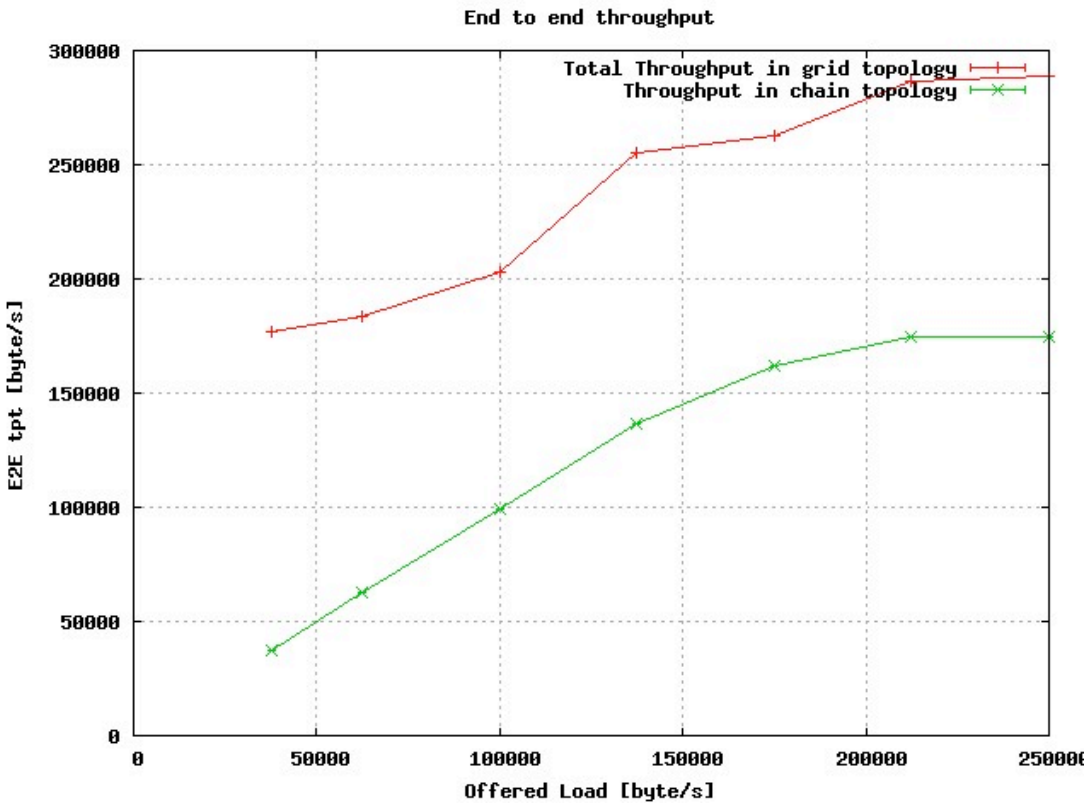


*Figure 4.19 : Comparison between chain and grid*

Figure shows that grid can support a maximum end to end throughput of 290 KByte/s, which is much more than 170 KByte/s obtained in the chain with a single flow.

### 4.2.1. Third scenario : Two macro-flows.

We start two flows at the same time : one from node 0 to 4 and the other from different nodes in different cases :

- from 5 to 9
- from 10 to 14
- from 15 to 19
- from 20 to 24
- from 25 to 29 (after adding a new row to the grid)

We are varying the distance from chain-shape flows. We want to evaluate the number of accepted flows.

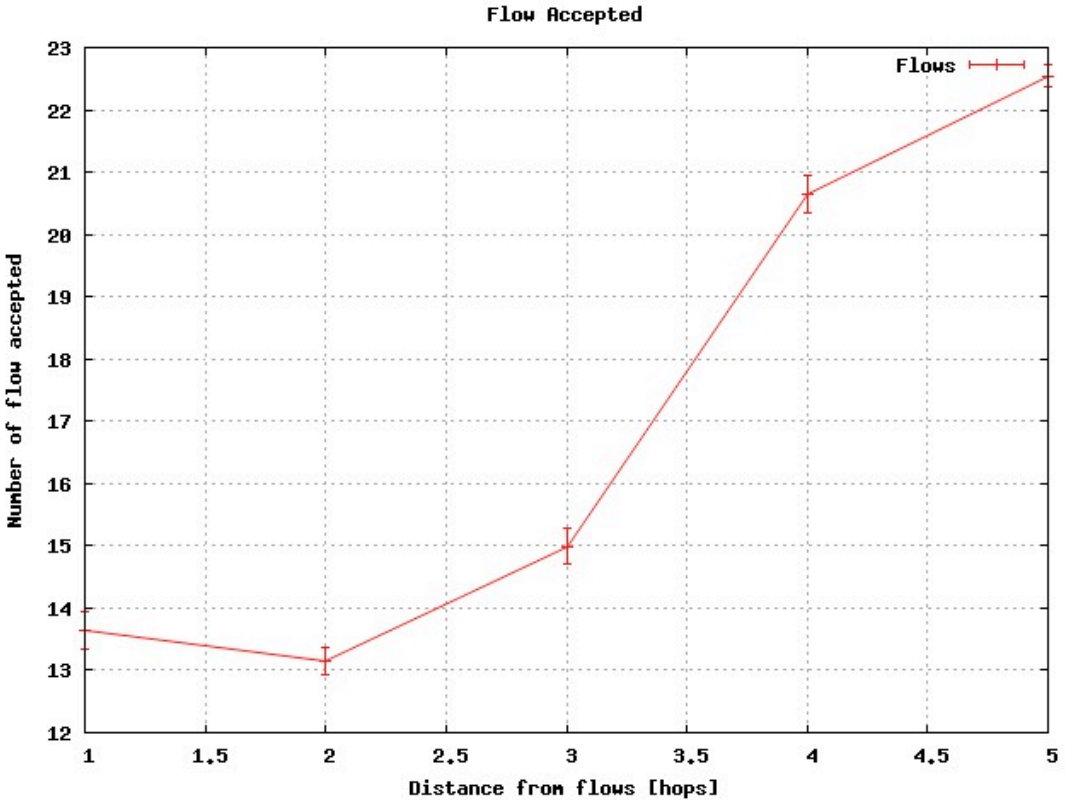


*Figure 4.20 : Flows Accepted*

Figure shows that number of accepted flows increases so that MDA exploits spatial reuse in this topology.

## 4.3. Grid with APs

We evaluate MDA in a more realistic scenario. We use a 5x5 grid topology with the following parameters :

| Parameter | Value |
|---|---|
| DTIM Duration | 0.032 s |
| DTIM unused by MDA | 1% |
| Advertisement Periodicity | 0.16 s |
| MDA Packet Life | 0.032 s |
| Minimum prob to jump | 20% |
| Maximum prob to jump | 80% |
| Step to increase prob to jump | 1% |
| Loss lower bound | -50 |
| Loss upper bound | 50 |
| Step to increase loss counter | 1 |
| Step to decrease loss counter | -10 |

*Parameter configuration*

Average distance from nodes is set to 100 meters. This distance is perturbed of quantity extracted from a uniform distribution with average 25 meters and of angle extracted randomly in a uniform manner between 0 and 2π [7].

In this topology, logical links are not regular as in previous grid, but depends on distance and rate selection threshold.

Rate of logical links is tuned with a parameter named “tolerance”, which stands between 0 and 1 (0 excluded). If tolerance is set to 1, the rate used between two nodes is the highest in order to have received power greater than minimum SNR for

that rate when interference from other nodes is null. If tolerance is less than 1, then minimum SNR is increased, so nodes use lower rates but have stronger logical links. Tolerance can be seen as a mean to increase robustness of logical links to the detriment of supported rate.

Each traffic flow is carried with a constant bit-rate (CBR) traffic source, generating packets of constant size equal to 400 bytes at a fixed rate. The interval between two consecutive traffic flows is drawn from a Weibull distribution with scale λ (called trf-int-scale in figures) and shape k (set to 2). If admitted, the traffic flow is torn down after a duration, which is drawn from a log-normal distribution [8] with mean μ (set to 3) and standard deviation σ (set to 0.5).

There are 6 APs in the network, corresponding to nodes 4, 12, 14, 16, 18 and 21 from which flows are started.

Metrics of interest are :

1. Blocking probability
2. Dropping probability
3. Outage probability
4. Average number of active flows
5. DTIM Utilization

#### 4.3.1. First scenario : No Call Admission Control.

We start by deactivating Call Admission Control and we evaluate the system varying maximum number of jump permitted and Slot Selection Procedure. Tolerance is set to 0.5.
We now consider the case with maximum number of jump allowed set to a low value (5).

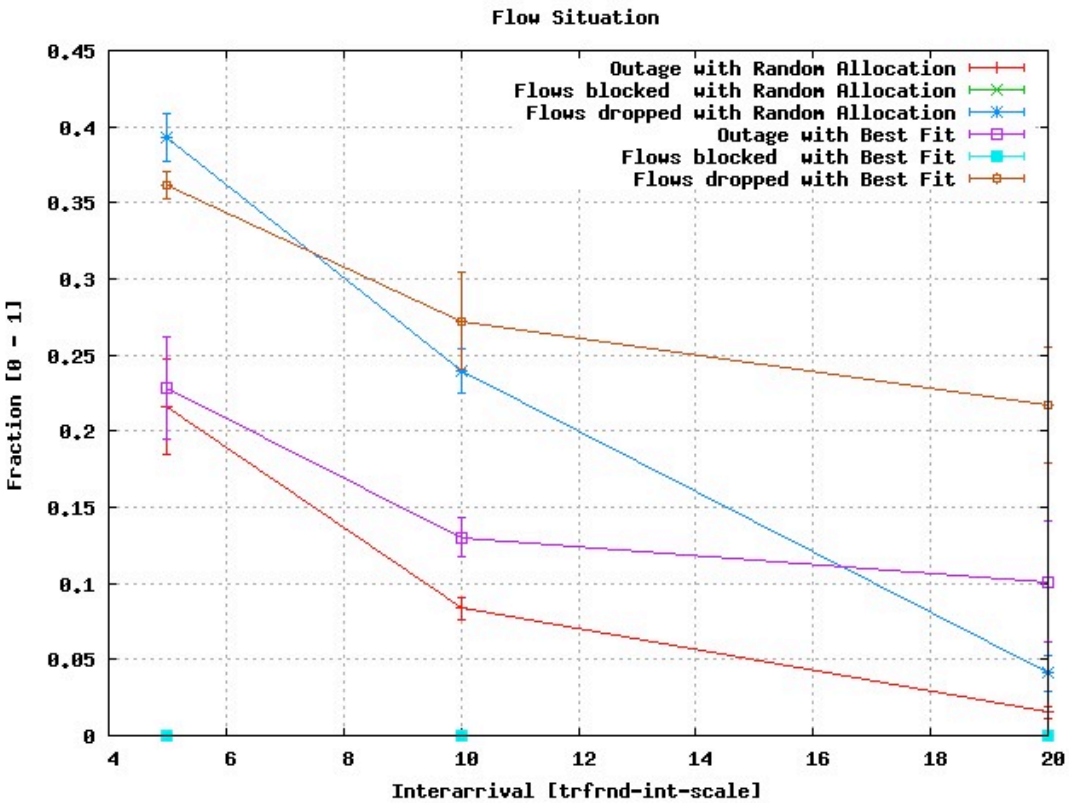


*Figure 4.21 : Outage, blocked and dropped flows metrics*

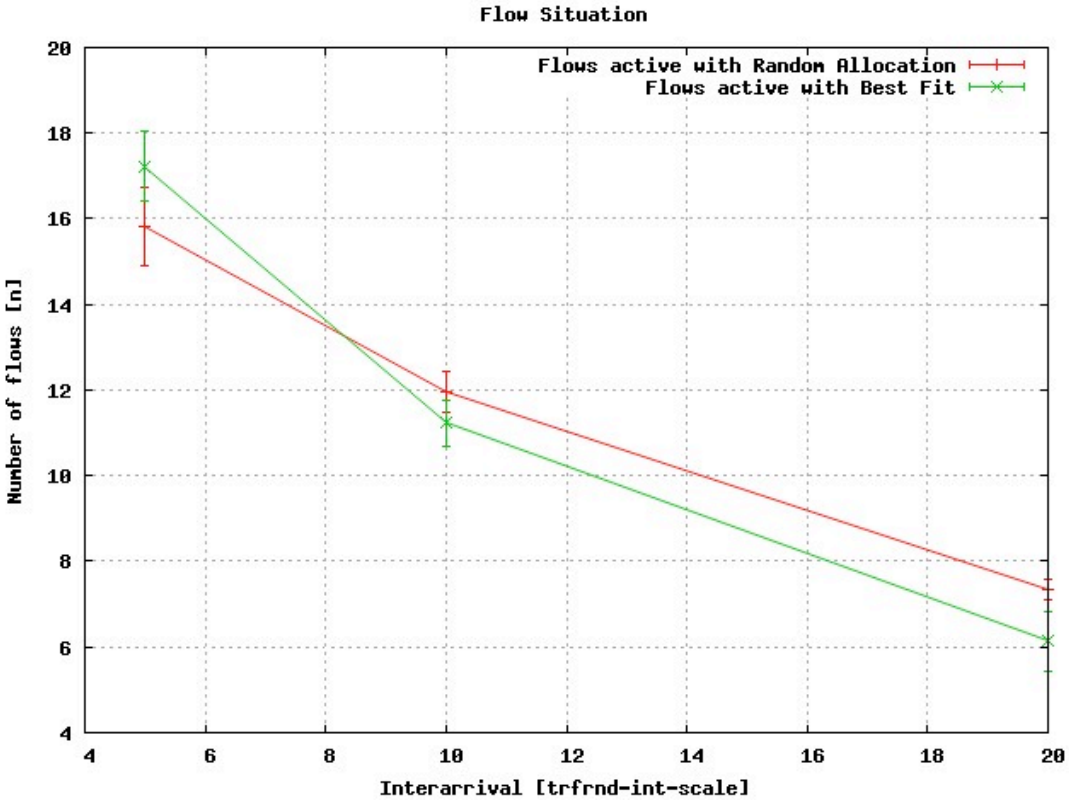


*Figure 4.22 : Flow active metric*

When inter-arrival is high (network is not loaded), Random Allocation behaves better than Best Fit because there are less chances to have MDAOPs that overlap or interfere.
With Best Fit, instead, nodes tend to use the same locations, so Dynamic Relocation procedure is used more often. As max jump is set to a low value, more flows are dropped

Reducing inter-arrival time, Best Fit starts to behave better than Random Allocation. This probably happens because in overload condition Random Allocation selects locations that possibly interfere, as Best Fit already did before. In addition to this, Random Allocation suffers from DTIM Fragmentation. So this condition exchange is not due to Best Fit improvement, but to Random Allocation that get worse.

We now consider the case with maximum number of jump allowed set to a high value (100).

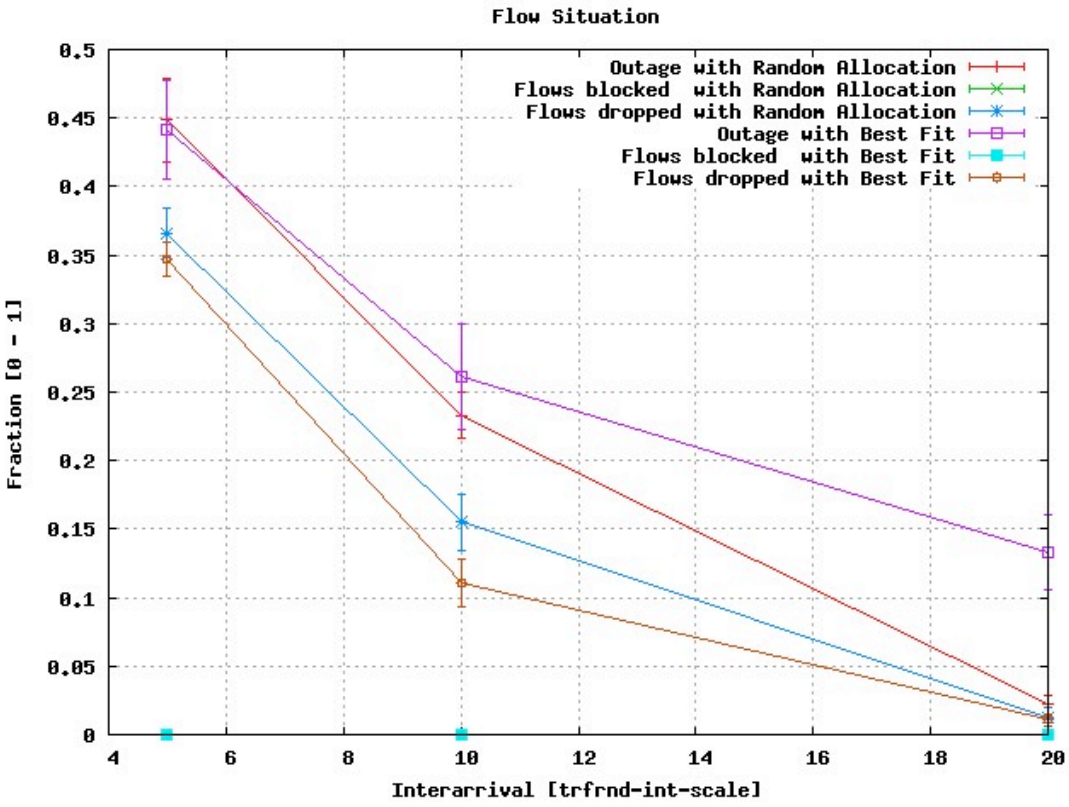


*Figure 4.23 : Outage, blocked and dropped flows metrics*

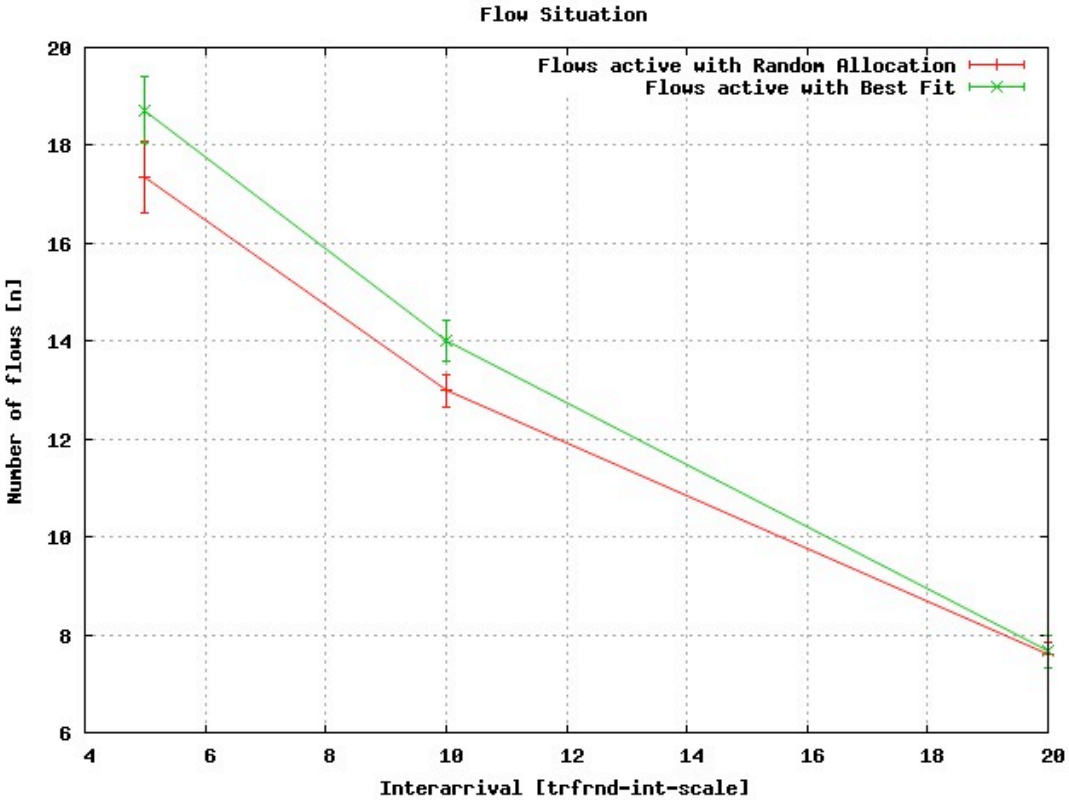


*Figure 4.24 : Flow active metric*

Best Fit maintains a number of active flows always higher than Random Allocation even if invokes Dynamic Relocation procedure more often. This is due to high value of maximum jump allowed. Unfortunately, multiple reallocations lead to increase interference. In fact, outage is always bigger under Best Fit.

When network is loaded, outage values under Random Allocation get worse as it start to reallocate MDAOPs more frequently (like in case of low max jump).

We can observe that outage values, no matter of Slot Selection procedure, are bigger when maximum number of jump allowed is high, as more reallocations lead to more interference.

We now compare DTIM Utilization measured at each node in case of max jump 5 and 100.

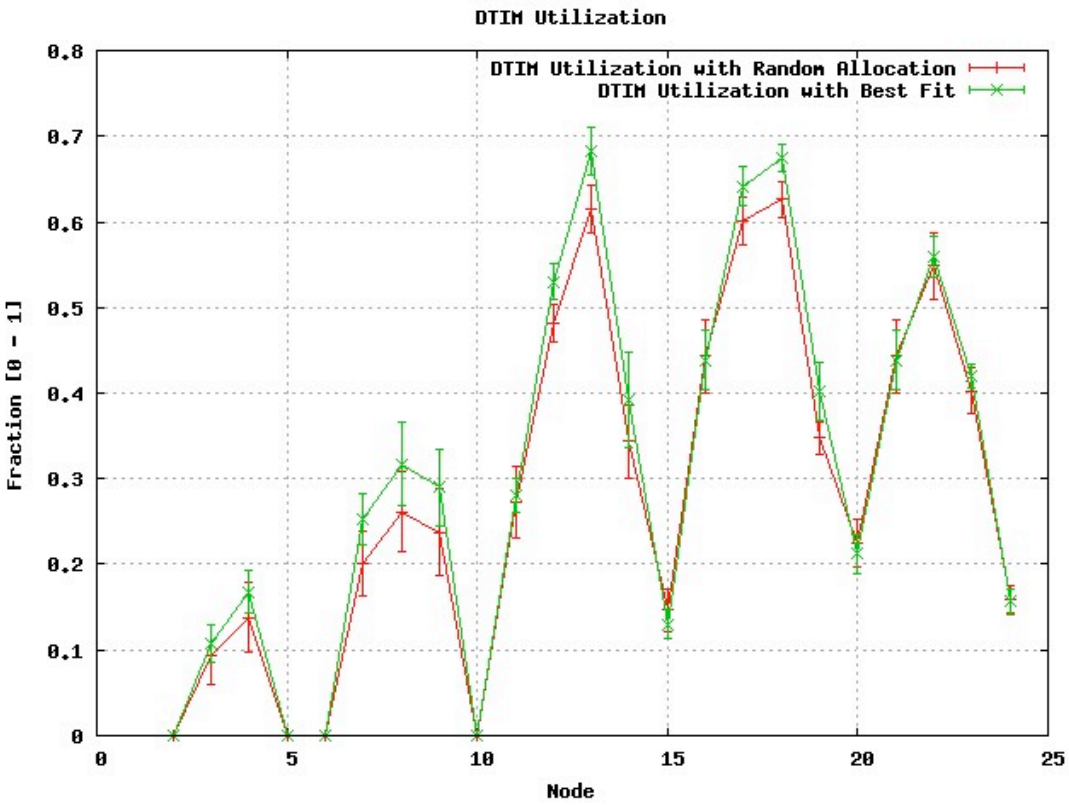


*Figure 4.25 : DTIM Utilization with max jump 5*

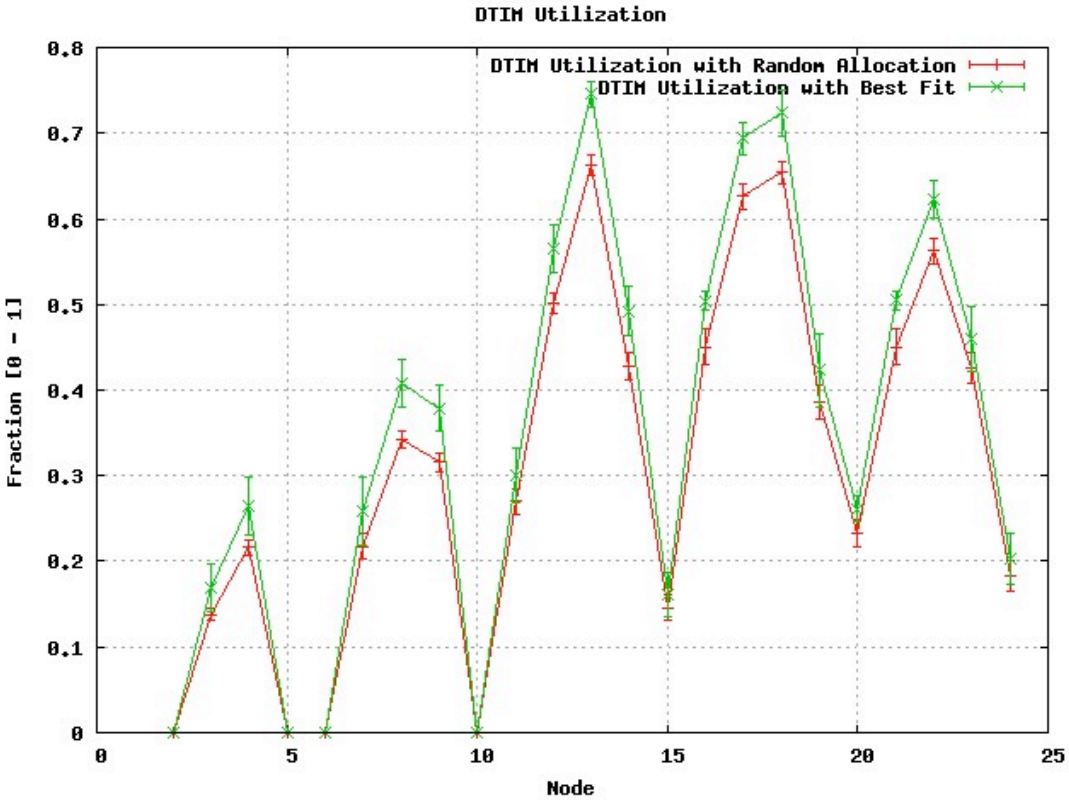


*Figure 4.26 : DTIM Utilization with max jump 100*

We observe the following things :

- Best Fit uses more DTIM respect to Random Allocation
- DTIM Utilization increases with maximum number of jumps allowed.

Moreover, we can observe points in the grid where DTIM is more used.
DTIM Utilization peak is at node 13, which is surrounded by three APs (nodes 12, 14 and 18). Then node 18, which is an AP itself and it is near to other APs.
Node 22 is near to three APs too, but its logical links to the APs are established at an higher rate respect to those of 13 to its APs, so it sees a bigger DTIM capacity.
Node 4 is an AP but it is isolated respect to other sources of traffic.
Node 1 and 2 are far from any flows, so they see a completely free DTIM.

We now consider the case with Dynamic Relocation disabled and Random Allocation.

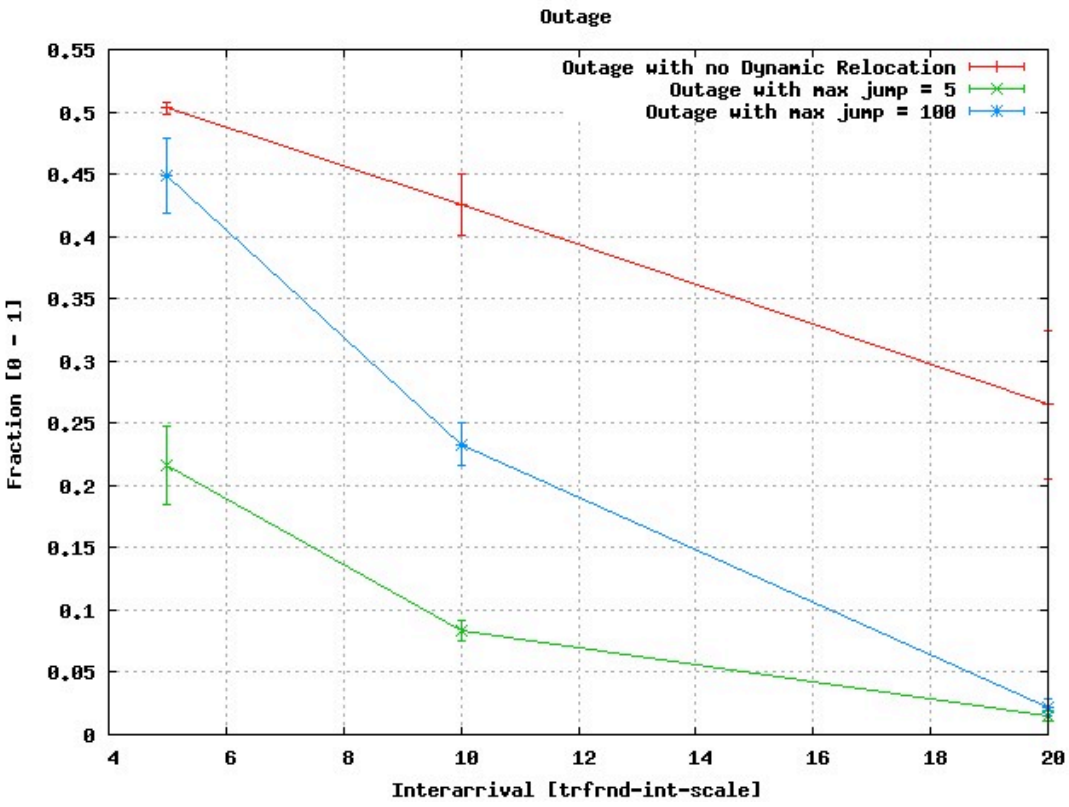


*Figure 4.27 : Outage comparison*

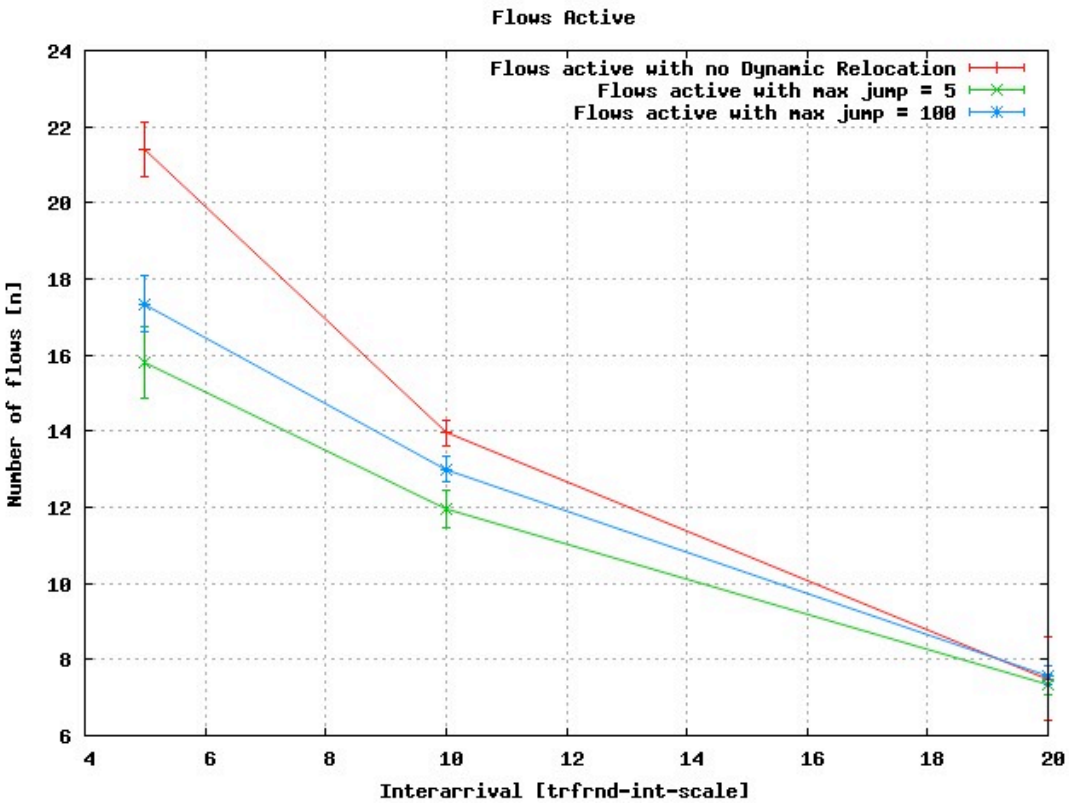


*Figure 4.28 : Flow active metric*

We compare results obtained with Dynamic Relocation disabled and those with max jump 5 and 100.

With no dynamic relocation there more active flows, but outage is bigger as well because there is no control of interference. Flows are dropped only if they cannot find available resources at setup time as they are not reallocated.

We can observe that Dynamic Relocation does not always resolve interference problems, especially when maximum number of flow allowed is high. We consider a new metric, which corresponds to the number of active flows that experience packet loss below 5%. This metric is expressed as :

*flows_active * (1 - outage)*

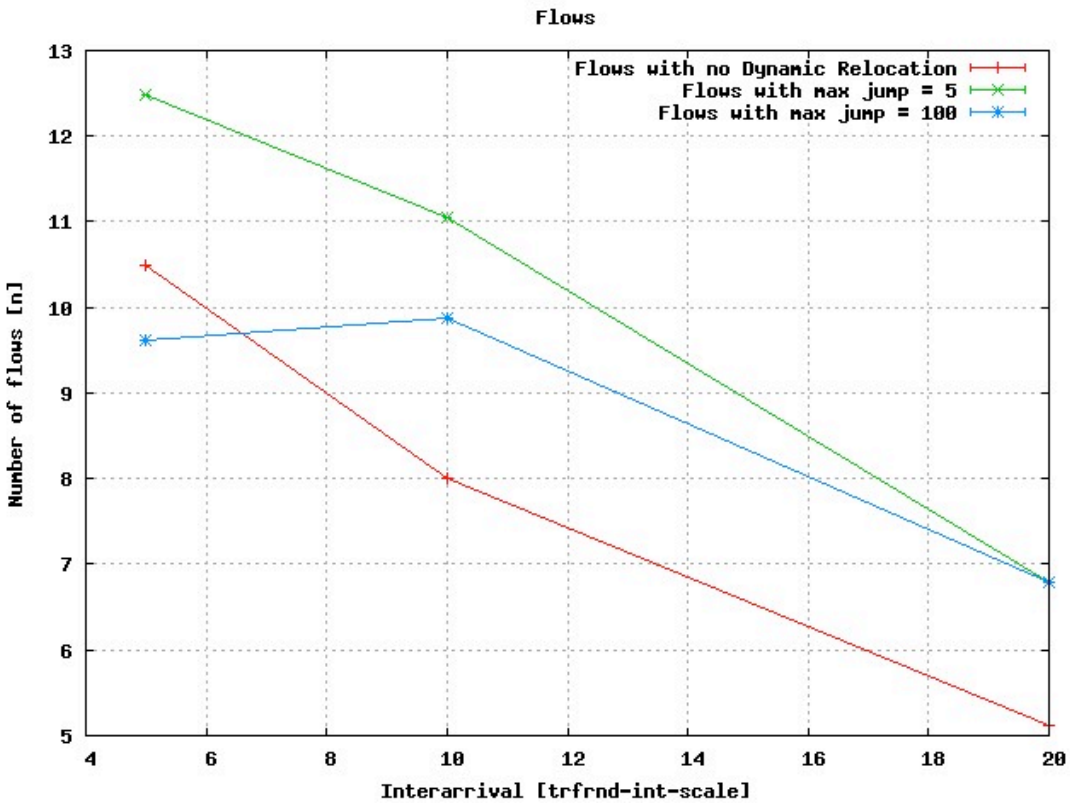


*Figure 4.29 : Flows with QoS requirements satisfied*

When network is lightly loaded, Dynamic Relocation behaves well. As load varies it works fine only with a limited amount of jumps.

### 4.3.2. Second scenario : Call Admission Control.

We evaluate the system varying maximum number of jump permitted and CAC acceptance parameter (called in figures “mpls-min-meas”).
Flows are accepted basing on DTIM Utilization measured by MDA Manager, which compares this value with the acceptance parameter. We use two values :

- 0.3 : flow is accepted if DTIM Utilization at intermediate hops is less 0.7
- 0.5 : flow is accepted if DTIM Utilization at intermediate hops is less 0.5

We now consider the case with maximum number of jump allowed set to a low value (5).

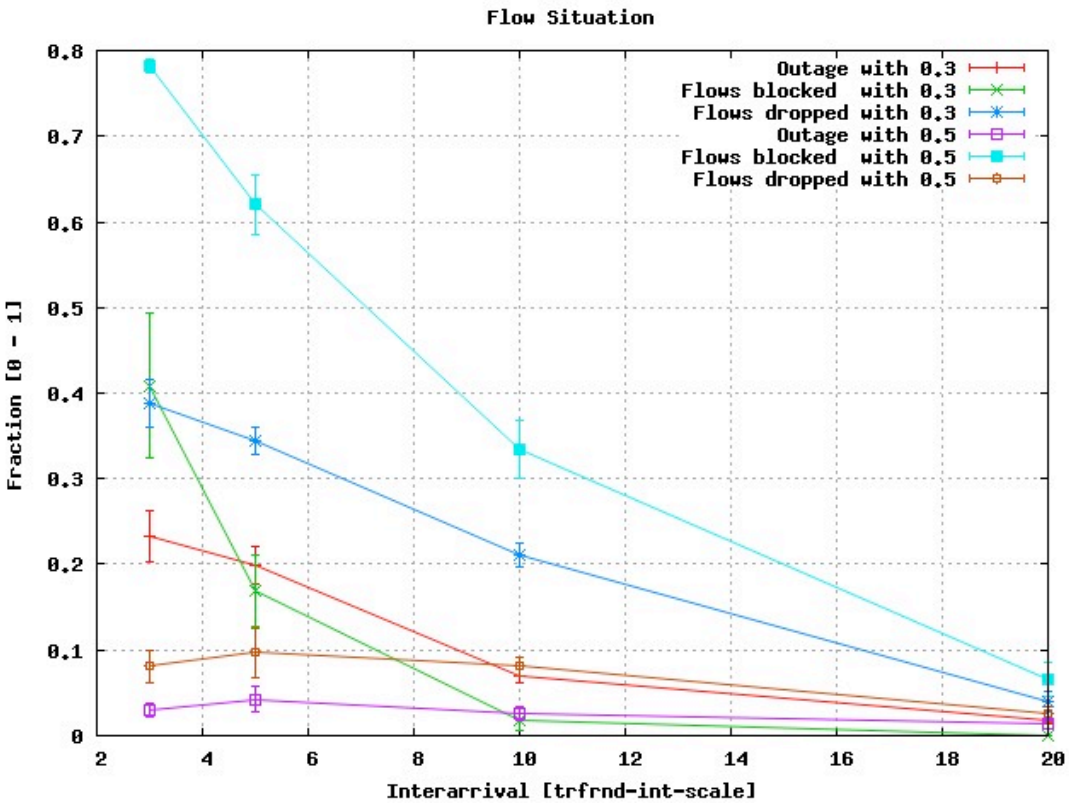


*Figure 4.30 : Outage, blocked and dropped flows metrics*

With CAC Acceptance parameter set to 0.5 many flows are blocked, so there are less interfering flows in the system. This can be seen with the other metrics : there are less flows dropped and outage remains at a lower value.

We can compare these results with those obtained when maximum number of jump allowed set to a high value (100).

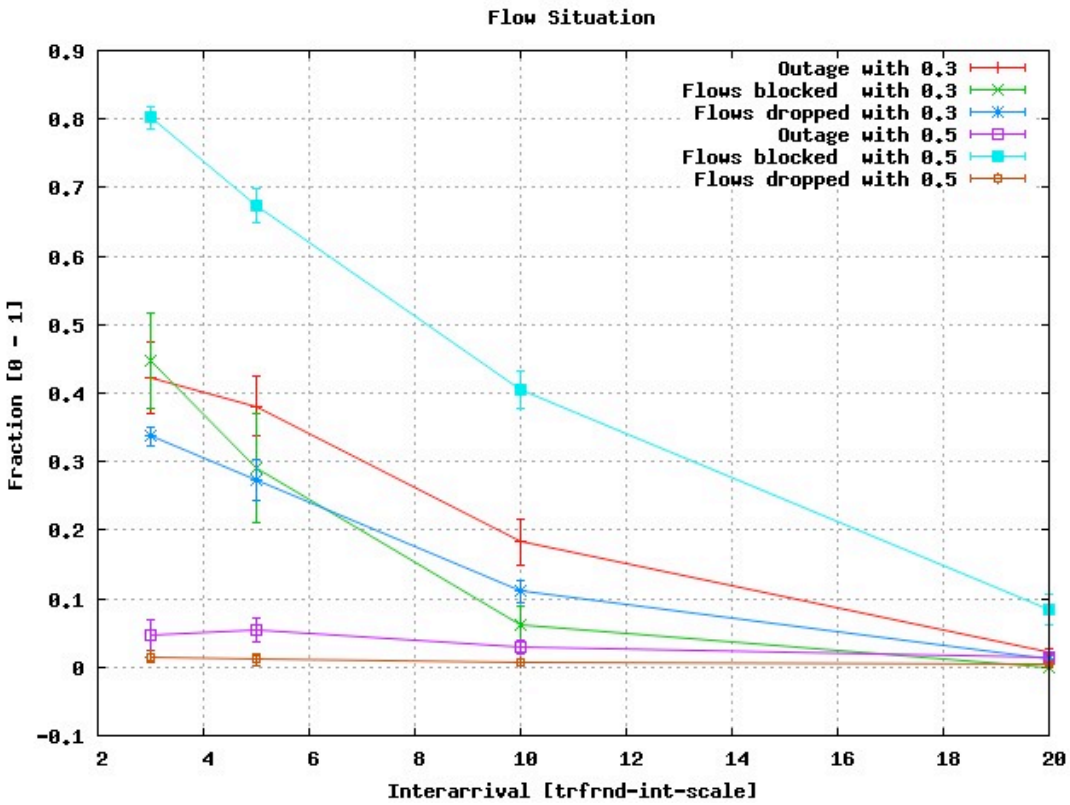


*Figure 4.31 : Outage, blocked and dropped flows metrics*

With CAC Acceptance parameter set to 0.5 there are few differences in outage metric obtained with maximum jump set to 5. This happen because CAC blocks most part of flows. There is a difference in the fraction between blocked flows and dropped flows, as with maximum jump set to 5 there are more dropped flows than blocked ones.

With CAC Acceptance parameter set to 0.3, outage metric changes more depending on number of maximum jump allowed. This happens because with max jump set to 100 flows interfere more. We can evaluate the number of flow that experience a packet loss less than 5%.

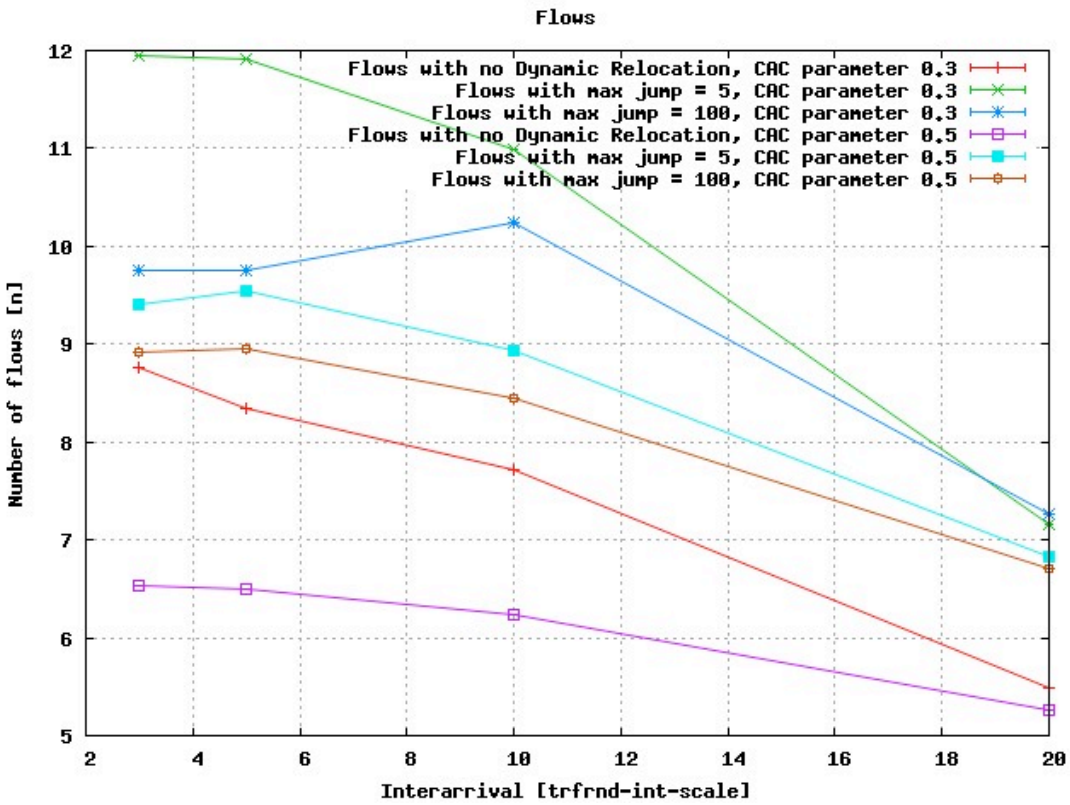


*Figure 4.32 : Flows with QoS requirements satisfied*

More flows are accepted when Dynamic Relocation is active and maximum number of jump is set to 5, while when set to 100 interference is higher. When Dynamic Relocation is disabled, more active flows experience packet loss.
Figure shows that CAC Acceptance parameter set to 0.5 is too restrictive.

Referring to previous figures and comparing the number of flows that experience a packet loss less than 5% in case of CAC active and not active we can assume that Call Admission Control is not necessary in this scenario, especially for MDA.

### 4.3.3. Third scenario : Tolerance set to 1.

We evaluate the system with the same condition of previous scenario, but setting tolerance parameter to 1.
We compare the number of flows that experience a packet loss less than 5% in case of CAC active, with acceptance parameter set to 0.3, by varying max jump and disabling Dynamic Relocation.

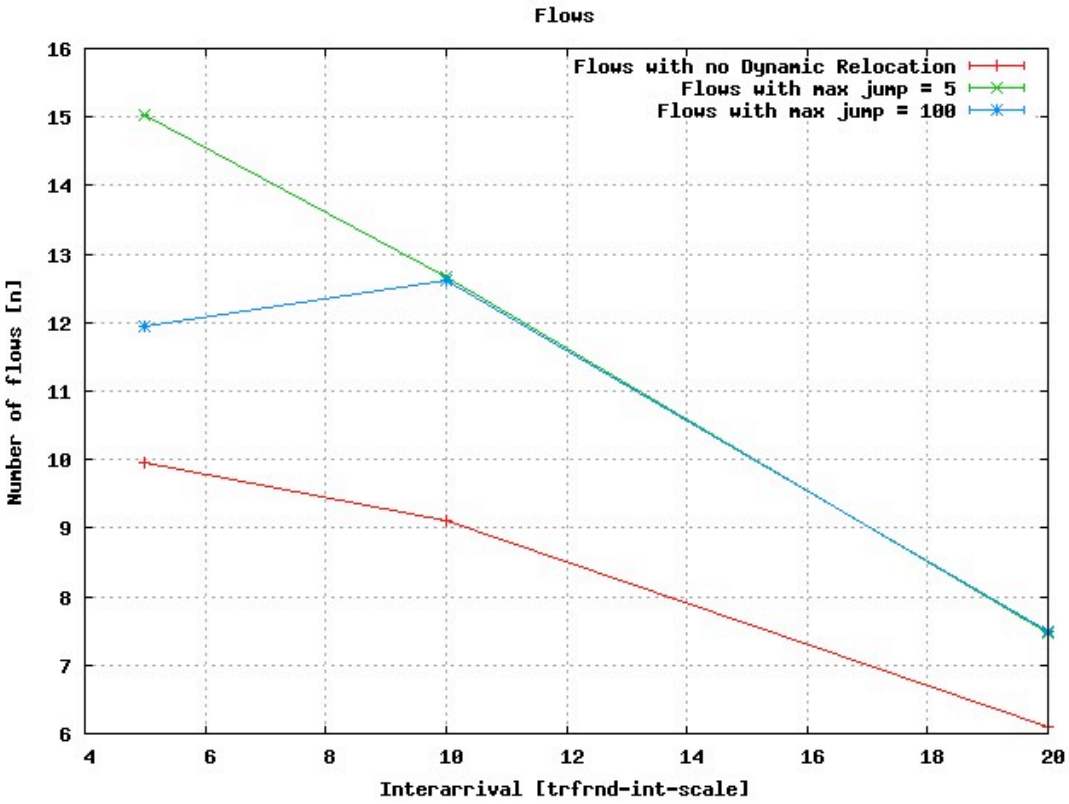


*Figure 4.33 : Flows with QoS requirements satisfied*

In general, more flows are accepted as logical link capacity has been increased. Moreover, the number of happy flows with max jump set to 100 starts to diverge from the value obtained with max jump set to 5 after trfnd-int-scale 10, while in previous scenario was before trfnd-int-scale 10.

### 4.3.4. Fourth scenario : Protocol-model.

We evaluate the system with the same condition of previous scenario, but using a different interference model called, for historical reasons, protocol model. By activating protocol model, concurrent transmissions of nodes that are two hops distant do not interfere.
Obviously, this is not realistic respect to the model used so far.

We observe how MDA behaves with and without Dynamic Relocation.

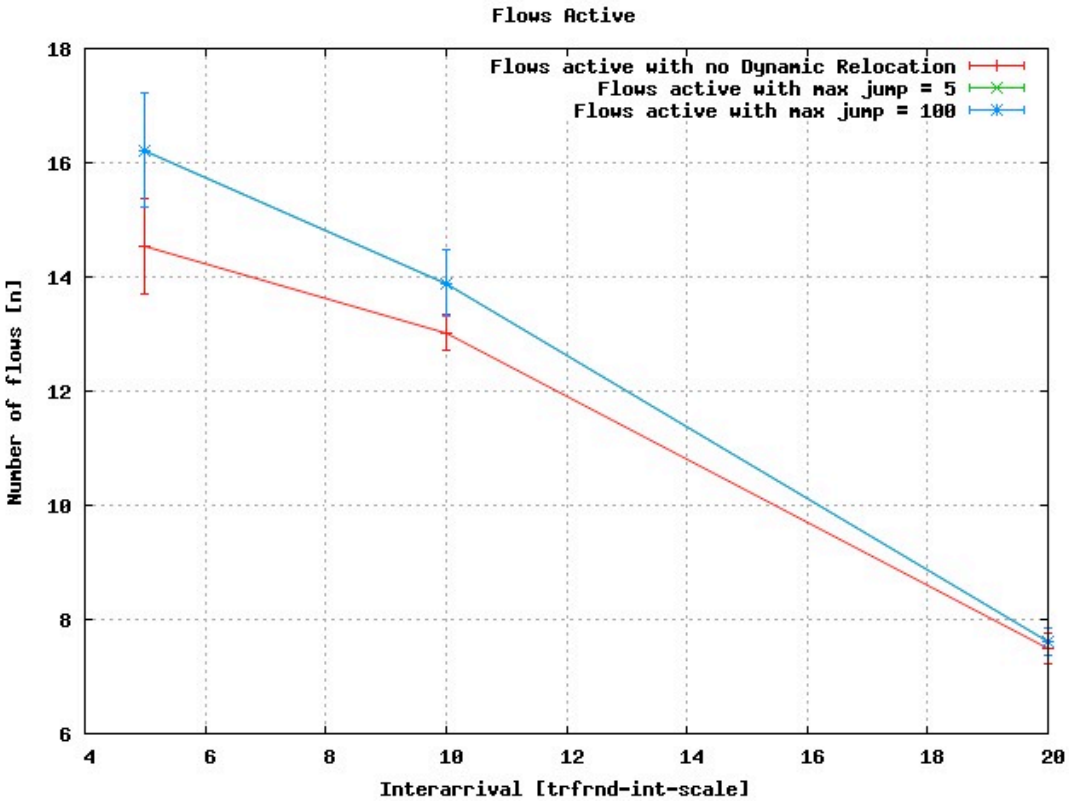


*Figure 4.34 : Flow active metric*

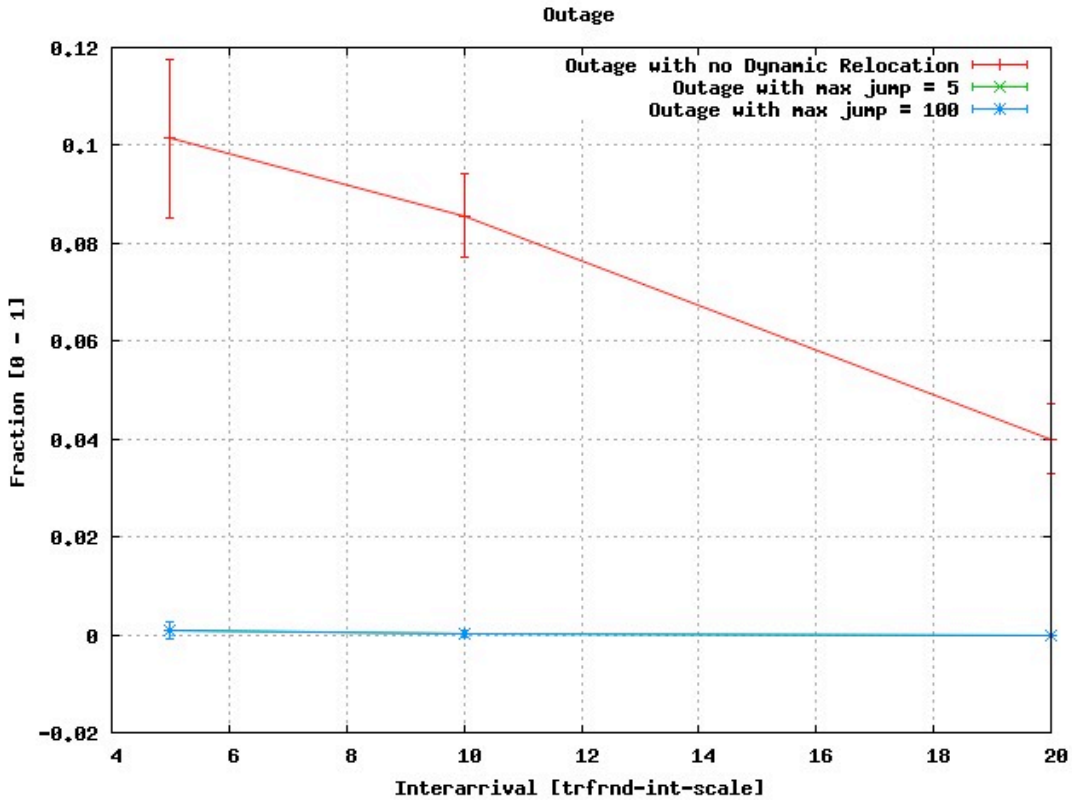


*Figure 4.35 : Outage*

Figures show that MDA behaves better than in previous scenario. In fact, now Times and Interfering Times data structures are more consistent with interference model. However, Dynamic Relocation is still useful and permits to obtain null interference among flows.

We observe that performance are the same if setting maximum number of allowed jumps to 5 or 100. That probably means that a flow never jumps more than five times.

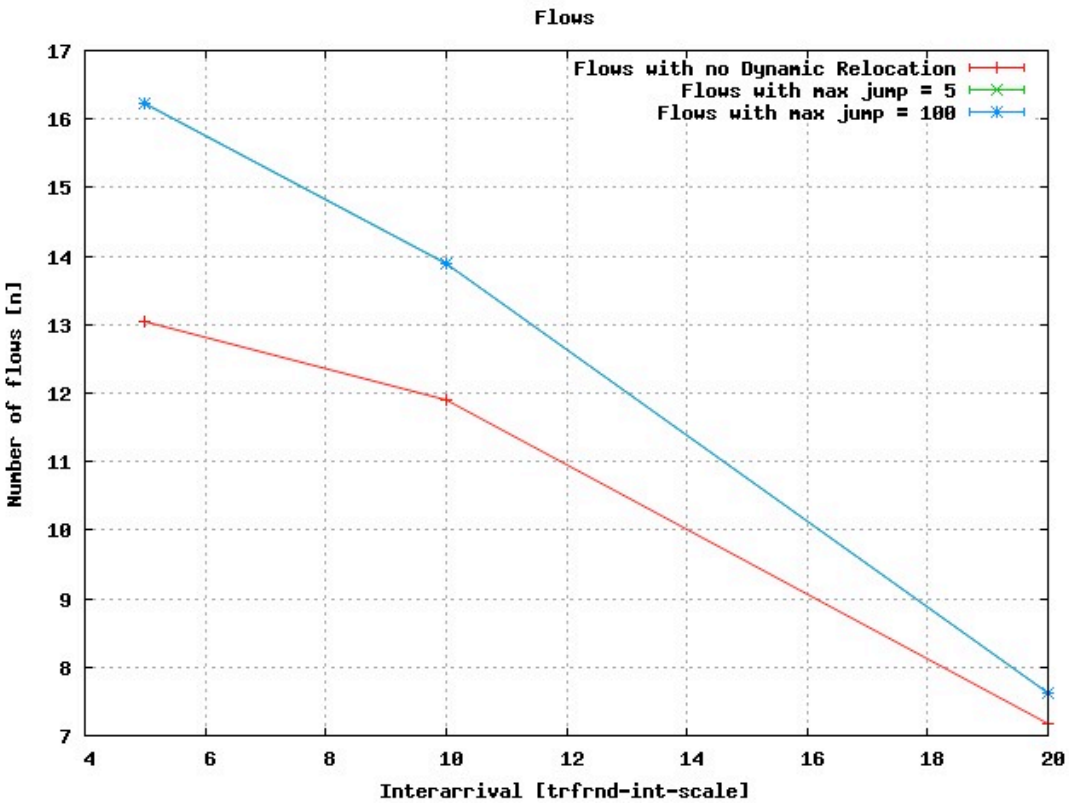


*Figure 4.36 : Flows with QoS requirements satisfied*

Figure shows the number of flows that experience less than 5% of packet loss. Referring to previous figures, system accepts more flows and difference between MDA with or without Dynamic Relocation is less evident.

# Conclusions

In this work we analyzed Mesh Deterministic Access (MDA), which is an optional mechanism of 802.11s standard, and we implemented it in ns-2 simulator in order to assess its capability of reducing interference. Furthermore, we have designed an algorithm, called Dynamic Relocation, that intends to reduce interference under realistic conditions.

We have assessed the effectiveness of MDA and Dynamic Relocation through simulation in different scenarios (chain and 5x5 grid topology) with CBR traffic. Results have shown that standard MDA is able to reach its goals only when interference outside two-hop range is considered negligible, while Dynamic Relocation is necessary in more realistic conditions.

MDA with Dynamic Relocation is more suitable than DCF to be deployed in a Wireless Mesh Network as it can exploit spatial reuse and satisfy QoS requirements, even in case of high load and high number of sources. Results obtained in grid topology show that Call Admission Control is not necessary if Dynamic Relocation is in use as it can adapt to traffic load changes in a efficient manner.

# References


[1] Ian F. Akyildiz, Xudong Wang, “A Survey on Wireless Mesh Networks”, IEEE Radio Communications, September 2005

[2] 802.11 Working Group of the IEEE 802 Committee IEEE “P802.11sTM/D1.06”, July 2007

[3] Myung J. Lee, Jianliang Zheng, Young-Bae Ko, Deepesh Man Shrestha, “Emerging standards for wireless mesh technology”, IEEE Wireless Communications, April 2006

[4] G. Hiertz , T. Junge, S. Max, Y. Zang, L. Stibor, D. Denteneer, “Mesh Deterministic Access (MDA) - Optional IEEE 802.11s MAC scheme - Simulation Results”,IEEE 802.11 WLAN Working Group Session, September 2006

[5] Law, A. M. and Kelton, W. D. Simulation modeling and analysis. Third edition, McGraw Hill, 2000, ISSN 0070592926.

[6] Claudio Cicconetti, Enzo Mingozzi, Giovanni Stea, “An Integrated Framework for Enabling Effective Data Collection and Statistical Analysis with ns-2”, Dipartimento di Ingegneria dell’Informazione, University of Pisa, Italy, October 2006

[7] J. Camp, J. Robinson, C. Steger, and E. Knightly, "Measurement Driven Deployment of a Two-Tier Urban Mesh Access Network," Proceedings of MobiSys 2006, Uppsala, Sweden. June 2006.

[8] Xiaoqiao Meng, Starsky Wong, Yuan Yuan, Songwu Lu, "Characterizing Flows in Large Wireless Data Networks," in ACM MOBICOM 2004, Philadelphia, Pennsylvania, September 2004.