

# CHAPITRE 3

## MESURES DE COMPARAISON ET ENVIRONNEMENT DE SIMULATION

### 1 Introduction :

L'étude et la mise en œuvre d'algorithmes de routage pour assurer la connexion des réseaux ad hoc et réseaux maillés au sens classique du terme (tout sommet peut atteindre tout autre), est un problème complexe. L'environnement est dynamique et évolue donc au cours du temps, la topologie du réseau peut changer fréquemment. Les protocoles de routage doivent être évalués afin de mesurer les performances de la stratégie utilisée et de tester sa fiabilité. L'utilisation d'un réseau ad hoc ou maillé réel dans une évaluation est difficile et coûteuse, en outre de telles évaluations ne donnent pas généralement des résultats significatifs. Le réseau réel n'offre pas la souplesse de varier les différents paramètres de l'environnement et pose en plus le problème d'extraction de résultats; c'est pour cela la majorité des travaux d'évaluation de performances utilisent le principe de simulation vu les avantages qu'il offre.

En effet, la simulation permet de tester les protocoles sous une variété de conditions. Le simulateur, qui constitue une plate-forme construite avec un certain langage, permet de varier les différents facteurs de l'environnement tel que le nombre d'unités mobiles, l'ensemble des unités en mouvement, les vitesses des mouvements, le territoire du réseau et la distribution des unités dans ce territoire. Initialement, chaque unité est placée aléatoirement dans l'espace de simulation. Une unité reste dans sa position courante pendant une certaine durée (pause time), par la suite elle choisit une nouvelle vitesse et une nouvelle localisation vers laquelle elle se déplace. Chaque unité répète ce même comportement jusqu'à la fin de la simulation.

Les paramètres mesurés dans une évaluation dépendent de la stratégie de routage appliquée (par exemple dans le cas où on veut comparer deux versions d'un même protocole), mais généralement tout simulateur doit être en mesure d'évaluer : (a) le contrôle utilisé dans le mécanisme de mise à jour de routage. (b) les délais moyens du transfert des paquets et (c) le nombre moyen de nœuds traversés par les paquets de données.

### 2 Les indicateurs de performance :

Les mécanismes de routage sont pour le but de trouver les chemins de routage tout en conservant la meilleure performance de réseau. Pour comparer les performances des différents

mécanismes, nous avons besoin de critères de comparaison. Ces critères sont des métriques de performance. Les métriques de performance varient selon les critères recherchés par le protocole de routage. Dans notre cas, nous cherchons à maximiser les débits, réduire les délais du trafic dans le réseau et limiter la surcharge due aux paquets de contrôle. Nous utiliserons le débit du réseau, le délai, le taux de perte, le taux d'efficacité normalisée (normalized effectiveness ratio) et le surdébit de routage (routing overhead) comme métriques de performance.

Les critères de performance suivants sont essentiels pour les systèmes de mise en réseau:

- Les paramètres par flux: Ce groupe de paramètres s'occupe des paramètres de qualité de service intra-débit tels que le délai, le ratio de perte de paquets, nervosité, nombre de sauts, le débit et les interférences.

- Les paramètres Par nœud: La complexité de calcul et l'efficacité énergétique sont parmi les calculs qui révèlent les performances par nœud des mécanismes de routage.

- Les paramètres par-lien: les paramètres par liaison incluent la qualité de la liaison, l'utilisation des canaux, la vitesse de transmission et la congestion.

- Les paramètres d'inter flux: Ces paramètres considèrent l'interaction entre les différents flux de trafic sur les différents liens tels que interférence inter-flux et l'équité.

- Les paramètres de l'ensemble du réseau: Pour l'amélioration de la performance générale du réseau et pour le support de la QoS pour chaque flux de trafic, les paramètres de l'ensemble du réseau tels que le débit total du réseau sont pris en compte. Il y a deux groupes d'indicateurs de performance de point de vue des utilisateurs, qui sont:

- Les paramètres de performances directs: Les utilisateurs sont directement affectés par ces paramètres. Les exemples sont QoS, le débit et l'efficacité énergétique.

- Les paramètres de performance indirects: Ceux-ci sont invisibles pour les utilisateurs et ils influent indirectement sur la qualité de service, le débit et l'efficacité énergétique éprouvée par l'utilisateur.

L'optimisation de performance de la métrique de routage ou tous les paramètres cités sont pris en compte est une tâche très compliqué. Les mécanismes de routage considèrent seulement une partie de ces paramètres. Dans ce travail, les métriques suivantes sont envisageables pour l'estimation de performance de routage :

### **2.1 Le délai de bout en bout :**

Le délai de bout en bout estime le temps total écoulé pendant que le paquet est transmis à partir de sa source à son destinataire à travers un réseau. Dans un réseau non encombrée, qui

contient des N-1 routeurs entre la source et nœuds de destination, le délai de bout en bout est calculé avec la formule suivante :

$$d_{end-end} = N(d_{proc} + d_{trans} + d_{drop}) \quad (1)$$

Où;

$d_{proc}$ : Le délai de traitement à chaque routeur et hôte source

$d_{trans}$ :  $L$  (taille des paquets) /  $R$  (vitesse de transmission en bits par seconde)

$d_{drop}$ : Délai de propagation sur chaque lien

Dans un canal sans fil, la durée dont le paquet est tamponné à la couche réseau avant d'être desservie par la couche MAC et les défaillances possibles de transmission MAC sont d'importantes sources de délai.

## 2.2 Proportion de perte de paquets :

Elle correspond au non délivrance d'un paquet de données par rapport à ceux envoyés. En raison de la capacité limitée des fils d'attente des routeurs et les nœuds du réseau, les paquets sont susceptibles d'être abandonné après avoir atteint une file d'attente pleine. Tant que l'intensité de l'augmentation du trafic réseau, le réseau est susceptible de d'éprouver d'autres paquets qui ne dégagent à leur destination. Même si un paquet peut être retransmis, le taux de paquets perdus révèle de manière significative les caractéristiques de performance d'un WMN, en plus du délai de bout-en-bout.

## 2.3 Durée de voyage par saut(RTT) :

Les paquets Unicast de sondes peuvent être utilisés pour mesurer RTT. Cette procédure consiste à mesurer le temps écoulé pour la procédure sonde-ACK (la réponse du paquet de sonde). De nombreux échantillons sont traités pendant la moyenne de mobilité est chargé, d'où un échantillon ne peut pas représenter l'état de la liaison complète. La combinaison des RTTs par-saut révèle la RTT total pour un chemin de routage. Les mesures sont effectuées à la couche réseau, bien que per-hop RTT caractérise les performances des liens.

Il y a deux problèmes qui empêchent per-hop RTT d'être un indicateur de performance efficace. La première préoccupation est que la fonction de la charge du trafic et du délai d'attente diminue sa précision. La deuxième préoccupation a résulté de la dépendance au système d'où moyenne de mobile chargé. Des variations trop grandes sur les mesures causent des calculs peu fiables de per-hop RTT, quel que soit la moyenne de mobilité utilisé.

## 2.4 Adaptation à la qualité du lien

La qualité de la liaison d'un WMN est considérablement inconsistante en raison de facteurs suivants :

- Les liens entre les nœuds ont se soumis à des effets d'atténuation telles que la décoloration et les interférences, d'ailleurs les liens peuvent être complètement disjoint.

- Les nœuds sont mobiles, de nouveaux nœuds pourraient joindre le réseau tandis que les nœuds actuels pourraient quitter le réseau.

- L'équilibrage de charge et la répartition de trafic sont strictement liées les unes aux autres dans WMNs. Cette relation engendre des chemins de routage dynamiques et une variation de la qualité de liens.

Pour les raisons exposées ci-dessus, le mécanisme de routage d'un WMN doit être robuste aux changements de la qualité de la liaison.

## 2.5 Débit

La performance de débit de WMN est un facteur clé pour déterminer la caractéristique de l'évolutivité (scalabilité) d'un mécanisme de routage. En respectant la nécessité des messages de signalisation des protocoles de routage, le débit d'un WMN sans considération d'évolutivité est susceptible de diminuer de façon significative avec l'augmentation du nombre des nœuds. Le débit sur une liaison unique peut être calculé avec la formule ci-dessous:

$$T_l = \frac{K(P+H)}{\left[ K \frac{P+H}{R} + d \right]} \quad (2)$$

Où;

D: délai de lien

R: taux de service de liaison

P: Nombre de bits utiles par paquet

H: Nombre de bits d'en-tête par paquet

K: nombre de paquets par la fenêtre

## 2.6 Latence de Convergence

Un mécanisme de routage doit être capable de découvrir correctement la topologie du réseau et former ses chemins de routage en fonction de cette topologie. Les mécanismes de routage WMN suivent généralement une découverte de topologie distribuée plutôt que les approches centralisées pour les raisons découlant de leurs caractéristiques distribuées multi-hop. Les concepts suivants sont décisifs pour la valeur de latence de convergence:

- La fréquence des informations échangées: Bien que l'échange d'informations suffisant entre les nœuds soit vital pour adapter les changements de topologie, des messages d'informations trop fréquentes gaspillent les ressources du réseau.

- Le contenu des messages de signalisation: les messages de signalisation doivent être optimisés en ce qui concerne les mécanismes de protocole de routage pour réduire la surcharge de protocole.
- Approche d'échange d'information: la décision d'utilisation d'une diffusion unicast ou la diffusion multicast devrait être accordée pour maintenir l'équilibre entre la surcharge des messages et la précision des informations sur la topologie.

### **3 Environnement de simulation :**

#### **3.1 La simulation en général**

Une simulation consiste à représenter par un programme informatique un réseau, et un scénario d'utilisation de ce réseau afin de recueillir des statistiques permettant d'évaluer le fonctionnement d'un mécanisme donné. L'exécution du programme de simulation est donc enregistrée dans une trace, qui peut être analysée (soit en cours de simulation, soit a posteriori).

La simulation est l'un des outils classiquement mis en œuvre lorsque l'on veut analyser les performances quantitatives d'un système. Cet outil complète celui de la modélisation mathématique (qui ne permet pas en général de bien représenter les détails de fonctionnement d'un système complexe) et celui du test (qui demande une mise en œuvre lourde, et nécessite la réalisation effective des éléments du système).

La simulation quantitative permet de représenter un niveau de détails variable, et d'évaluer les performances d'un système ou d'un protocole avant que ce dernier ne soit réellement construit ou effectivement déployé dans un réseau. Par contre, comme le problème réel est « modélisé » par un programme, il est souvent nécessaire d'effectuer des simplifications du comportement réel qui peuvent rendre les résultats concernant le système simulé obtenus non représentatifs du comportement du système réel. Tout l'art du simulateur consiste à modéliser de façon simple mais représentative un système complexe.

Les différentes parties d'un programme de simulation sont les suivantes :

- La spécification du système à simuler (éléments d'un réseau, description des protocoles mis en jeu) ;
- La spécification des types de trafic devant circuler sur le réseau (les « sources », les « puits » (en anglais « sinks »), les caractéristiques des flux en terme de trafic, éventuellement les chemins devant être suivis par ces flux ;
- Le scénario simulé (quand débute l'activité de chaque source, quand elle se termine ; modifications éventuelles des caractéristiques de ces sources) ;

- Les outils de recueil d'information et de visualisation des résultats.

### 3.2 Le simulateur OMNET ++

Dans ce projet, nous allons réaliser nos expérimentations à l'aide d'OMNET++ (Objective Modular Network Testbed in C++) qui est un simulateur à événements discrets orienté objet, basé sur C++. Il a été conçu pour simuler les systèmes réseaux de communication, les systèmes multi processeurs, et d'autres systèmes distribués. OMNET++ est un projet open source dont le développement a commencé en 1992 par Andras Vargas à l'université de Budapest. Actuellement, Ce simulateur est utilisé par des dizaines d'université pour la validation de nouveaux matériels et logiciels, ainsi que pour l'analyse de performance et l'évaluation de protocoles de communication.

L'avantage de OMNET ++ est sa facilité d'apprentissage, d'intégration de nouveaux modules et la modification de ceux déjà implémentés.

#### 3.2.1 Architecture de OMNET++

L'architecture de OMNET++ est hiérarchique composé de modules. Un module peut être soit module simple ou bien un module composé. Les feuilles de cette architecture sont les modules simples qui représentent les classes C++. Pour chaque module simple correspond un fichier .cc et un fichier .h. Un module composé est composé de simples modules ou d'autres modules composés connectés entre eux. Les paramètres, les sous modules et les ports de chaque module sont spécifiés dans un fichier .ned (Network Description).

La communication entre les différents modules se fait à travers les échanges de messages. Les messages peuvent représenter des paquets, des trames d'un réseau informatique, des clients dans une file d'attente ou bien d'autres types d'entités en attente d'un service. Les messages sont envoyés et reçus à travers des ports qui représentent les interfaces d'entrer et de sortie pour chaque module.

La conception d'un réseau se fait dans un fichier .ned et les différents paramètres de chaque modules sont spécifiés dans un fichier .ini. OMNET++ génère à la fin de chaque simulation deux nouveaux fichiers omnet.vec et omnet.sca qui permettent de tracer les courbes et calculer des statistiques.

#### 3.2.2 Les composants d'OMNeT++

- Librairie de simulation (noyau ou kernel)
- Compilateur pour le langage de description de topologie (NED pour Network Descriptor)
- Editeur graphique de réseau (GNED pour Graphical NeTwork Descriptor)

- Interface Graphique (GUI pour Graphical User Interface) pour l'exécution de simulation (appelée Tkenv)
- Interface ligne de commande pour l'exécution de simulation (appelée Cmdenv)
- Outil de traçage de vecteur graphique (Plove)
- Divers outils (outil de génération de nombres aléatoires, outil de création de fichier makefile, etc...)

### 3.2.3 Langage de description de réseau (Network DEscription (NED)) et ses composants:

Un réseau est formé de "noeuds" reliés par des "liens". Les noeuds représentent les blocs (modules simples ou composés), tandis que les liens représentent les canaux, raccordements, etc.

La façon dont des éléments fixes (c.-à-d. noeuds) dans un réseau sont reliés ensemble s'appelle la topologie.

OMNeT++ utilise le langage NED qui facilite la création, l'édition, et la représentation graphique d'un certain réseau à simuler. Il suit une structure hiérarchique tenant compte de différents niveaux d'organisation. Un noeud par exemple peut être structuré selon les différentes couches OSI dont il se compose : couche physique, liaison de données, réseau, transport, application... De même, chaque couche peut être structurée selon les applications ou protocoles qui tournent au niveau de cette couche. Les fichiers de description de réseau ont un suffixe «.ned». Le compilateur NEDC traduit les descriptions de réseau en code C++. Puis, ces descriptions seront compilées par le compilateur C++ et incorporées dans la simulation exécutable. Donc une description NED peut contenir les composants suivants, d'une façon et en nombre arbitraires:

- Les déclarations d'Importation
- Définitions de Canaux
- Modules Simples et Composés

### 3.2.4 Interface utilisateur

L'interface utilisateur d'OMNeT++ concerne l'exécution de simulation. Elle permet à l'utilisateur de lancer et terminer des simulations et de changer des variables à l'intérieur des modèles de simulation. L'utilisateur peut examiner et corriger la simulation avec une interface graphique puissante.

Actuellement, deux interfaces utilisateur sont soutenues :

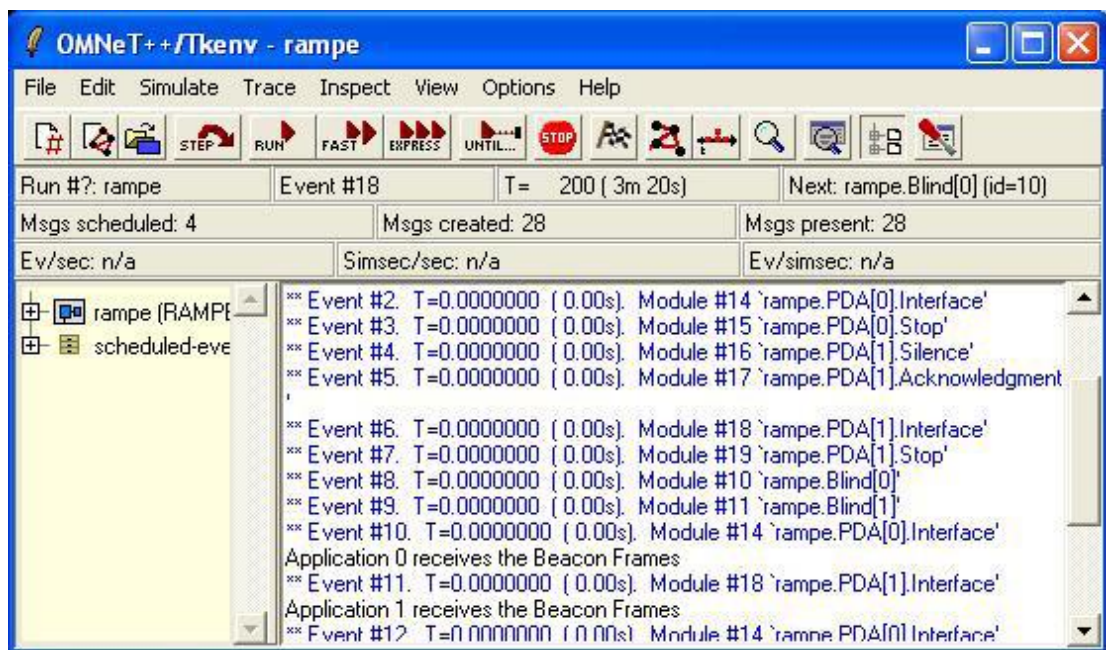
- **Tkenv** ("Tk-based graphical, Windowing User Interface"): c'est une interface graphique qui permet le « traçage », la correction et l'exécution de simulation. Elle a la capacité de fournir une image détaillée de l'état de la simulation à un point quelconque pendant l'exécution.

Caractéristiques de Tkenv:

- Fenêtre séparée pour chaque module

- Les messages programmés peuvent être observés dans une fenêtre pendant que la simulation progresse
- Exécution d'événement-par-événement
- Animation de l'exécution
- Fenêtres pour examiner et changer des objets et des variables dans le modèle
- Affichage graphique des résultats de simulation pendant l'exécution
- La simulation peut être remise en marche

Figure 3.1 présente l'interface Tkenv.



**Figure 3.1** Tkenv avec OMNeT++

- **Cmdenv** ("Command-line User Interface for Batch execution"): conçue principalement pour l'exécution en batch. C'est une interface par ligne de commande portative, petite et rapide. Cmdenv exécute simplement toutes les simulations « runs » qui sont décrites dans le fichier de configuration.

### 3.2.5 Différents types de fichiers

Les fichiers présents dans une simulation OMNeT++ :

- Fichier NED pour décrire le modèle de simulation
- Fichier de configuration «omnetpp.ini» pour l'initiation des variables et des paramètres
- Fichier C++ pour initialiser les différents modules et décrire leur fonctionnement
- Fichier de messages .msg décrivant les différents messages échangés entre les modules



- Fichier .h pour la déclaration des classes et variables qui seront utilisées dans le fichier de simulation C++
- Fichiers XML pour la représentation des données et scénarios de simulation

## **4 Conclusion**

Dans ce chapitre nous citons les critères d'évaluation et de comparaison des mécanismes de routages exposés dans le deuxième chapitre ainsi que l'environnement de simulation utilisé pour leurs réalisation. Dans ce qui suit, nous présentons les résultats et l'analyse de cette simulation.