Projet MIN17212 Simulation

Estimation du taux de collision dans un réseau LoRa

Youssef Ait El Mahjoub et Franck Quessette

8 mars 2022

— Le projet est à faire en **trinôme** (quelques projets peuvent être acceptés en monôme ou binôme, mais ne bénéficieront d'aucun avantage).
— Il faut rendre un tar (ou zip) du répertoire contenant les fichiers nécessaires à l'exécution de votre programme et un compte-rendu au format pdf.
— Ce fichier tar devra s'appeler NOM1-NOM2-NOM3.tar ou NOM1–NOM2–NOM3.zip. NOM1, NOM2 et NOM3 étant les noms de famille des membres du trinôme.
— Le projet est à déposer sur moodle avant le JJ MM 2022 23h59. La date de remise sera communiquée sous peu.

Objectif

L'objectif principal de ce projet est de programmer un simulateur LoRa en temps continu et de mesurer certains indices de performances.

1 Introduction

Les technologies LPWAN (Low Power Wide Area Network) se caractérisent par des liaisons à longue portée (plusieurs kilomètres) et présentent des typologies de réseaux en étoile. Ces systèmes ne visent pas à permettre des débits de données élevées (ni à minimiser la latence). Les principales mesures concernées sont plutôt l'efficacité énergétique, la modularité et la couverture de zones étendues, qui s'accompagnent d'une consommation d'énergie et de coûts de maintenance minimaux. Dans la famille LPWAN, la technologie la plus utilisée est le Long Rang (LoRa). Ceci est principalement dû à l'utilisation de bandes sans licence.

Par conséquent, les réseaux LoRa sont faciles à déployer sur une portée de plusieurs kilomètres. L'architecture du système LoRa se compose de trois éléments principaux : les noeuds ou capteurs, les passerelles, un serveur central.

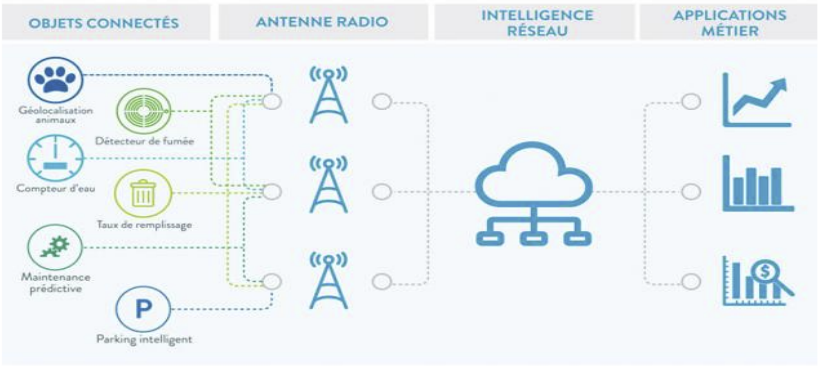

FIGURE 1 – Architecture d'un réseau LoRa.

2 Description du modèle

Dans ce projet, on s'intéresse à la partie "Objets connectés - Antenne radio" (voir figure ci-dessus). Le but étant de proposer une version simplifiée d'un simulateur LoRa en se concentrant sur le taux de collisions. Le réseau est constitué de K capteurs. Les capteurs émettent les informations (i.e. paquets) vers les différentes passerelles du réseau. La puissance et distance parcourus par un paquet (dans l'air) est défini par un facteur d'étalement (SF - Spreading Factor), il s'agit d'une sorte de canal de communication. A un instant t, un seul paquet peut être dans un SF, donc en émission.

Traitement d'une collision :

Une collision de paquets se produit, lorsqu'un paquet est en émission par un capteur et qu'en même temps, un autre paquet rentre en émission. Le premier paquet a donc été victime d'une collision, alors que le deuxième paquet a provoqué la collision. Suite à une collision, les deux paquets tentent à nouveau d'émettre jusqu'à 7 fois. Au bout d'une non réussite la 7 ème fois, le paquet est perdu.

Les durées d'attente :

La durée d'émission d'un paquet est une v.a qui suit une loi $Exp(e)$, un paquet qui n'a pas réussi son émission doit attendre un certain temps $Exp(w)$ avant de tenter de nouveau sa chance. Après une émission réussie, un capteur doit attendre $Exp(i)$ (les arcs en vert, Fig. 2) avant d'émettre un nouveau paquet (duty-cycle LoRa).

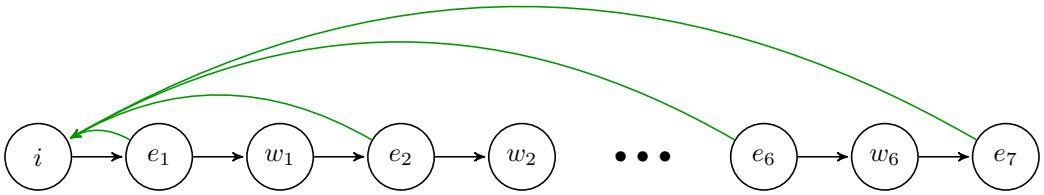

FIGURE 2 – Évolution de l'état d'un capteur.

3 Ce qu'il faut simuler

Soit un capteur $c \in \{1, \ldots, K\}$. Alors l'évolution de l'état de c est décrite par l'automate dans Fig. 2.

Un capteur est initié à l'état i, une fois le temps à l'état i passé, le capteur rentre à l'état $e1$ qui est la première tentative d'émission. Si l'émission est réussie, alors le capteur retourne à l'état i afin d'émettre un nouveau message. Sinon, ce dernier passe à l'état $w1$ où il attend. A la fin de $w1$, le capteur passe à l'état $e2$ et ainsi de suite. A la dernière réémission donc une émission dans $e7$, le capteur passe à l'état i après une collision ou après une émission réussie. Il va falloir maintenir l'état de chaque capteur au cours de votre simulation.
A noter : quand une collision se produit, chaque capteur, s'il le peut, passe en état $w_j,\ j \in \{1, \ldots, 6\}$. Et ce au même instant que la collision. Le paquet qui provoque la collision est donc entré en émission pour une durée nulle, mais le paquet victime de la collision avait déjà démarré son émission, donc le temps avant collision du deuxième paquet doit aussi être comptabilisé !
Exemple : A la fin de l'attente en $w1$ d'un capteur1, alors ce dernier s'apprête à émettre un message dans $e2$. Or s'il se trouve qu'un capteur2 est en émission dans, par exemple $e6$, alors une collision est comptabilisée dans $e1$ (qui est l'état du capteur1) puis aussi comptabilisée dans $e6$ qui a reçu la collision du capteur1. Après collision, capteur1 passe à $w2$ et capteur2 passe à $w6$.

$\Longrightarrow$ Le but est de déterminer **la probabilité de collision ainsi que le temps d'émission observé dans chaque état** e_j avec $j \in \{1, \ldots, 7\}$.
L'échéancier doit stocker l'état de chaque capteur, ainsi que la date d'arrivée (+durée) du prochain évènement.

4 Travail à faire

4.1 Implémentation

a) Quelles sont les variables principales de la simulation ?
b) Proposez une structure de données décrivant correctement l'échéancier.
c) Vous devez écrire les fonctions suivantes :

- "**Expo_Duree**" qui renvoie une durée suivant une loie Exponentielle de paramètre λ.
- "**Traitement_Event**" qui traite un événement, donc met à jours les variables de la simulation.
- "**Traitement_Collision**" qui traite la collision, en particulier met à jours l'état des deux capteurs en collision.
- "**Simulateur**" qui code le fonctionnement du simulateur.
- "**Main**" qui initialise la graine du générateur aléatoire et appelle la fonction "Simulateur".
- Vous êtes libre d'ajouter d'autres fonctions.

La simulation s'arrête, dans l'idéal, lorsque 10^3 messages ont été correctement émis par chaque capteur.
d) Ensuite vous devez :

- Fixer les paramètres : $K = 5$, $i = 0.1$ et $\forall j \in \{1, \ldots, 7\}$ $e_j = 10$ et $\forall j \in \{1, \ldots, 6\}$ $w_j = 0.25$.
- Dans une seule figure, tracer les 7 courbes des probabilités de collision dans les états e_j en fonction du temps de simulation.
- Dans la même figure, en couleur noir, rajouter l'intervalle de confiance à 90% de la probabilité de collision dans l'état $e2$. Ceci en exécutant 50 simulations.
- Tracer deux histogrammes du temps d'émission observé. Le premier concerne l'état $e1$, le deuxième concerne l'état $e2$. Indiquer la moyenne du temps observé dans chaque histogramme.
- Tracer la courbe de la probabilité de collision moyenne (moyenne sur les 7 états d'émission) en fonction du nombre de capteurs dans le réseau (K allant de 1 à 100).
- Analyser et expliquer les résultats obtenus.

Quelques détails :

- Vous devez utiliser R-project ou gnuplot pour la création des courbes.
- Vous devez avoir un fichier Makefile ou script Shell permettant d'enchaîner automatiquement la simulation et la création des courbes.

4.2 Questions

— A partir de quelle valeur de K la probabilité de collision dépasse les 70% ? Justifiez
— Que proposeriez-vous pour remédier à ce taux très élevé de collisions ? Justifiez
— Combien de capteurs faut il utiliser, afin d'assurer que dans 90% des cas, un message réussisse son émission en deux tentatives maximum ?

4.3 Compte rendu

Vous devez fournir un compte-rendu en pdf avec :

— Vos noms, prénoms et numéros d'étudiant.
— Un rappel en 3 lignes du sujet.
— Une explication en une page maximum de la programmation que vous avez faite.
— Des courbes issues de votre simulation. Ces courbes doivent être expliquées.
— Les réponses aux questions.
— L'analyse des résultats.