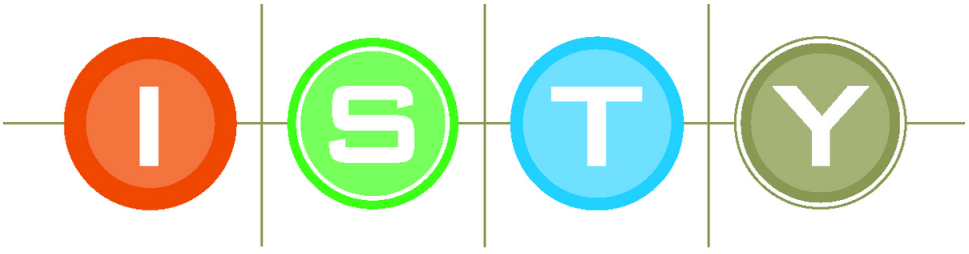
Projet Réseau

Pierre AYOUB – Océane FLAMANT

12 février 2019



INSTITUT DES SCIENCES ET TECHNIQUES DES YVELINES

Table des matières

1 Introduction

Afin d'éviter les coupures de l'Internet des Objets il est nécessaire de trouver des solutions automatisées. Cela permettrait aux équipements d'être autonome dans un environnement dynamique. Dans ce projet, nous allons appliquer ce concept au contexte des réseaux émergents LoRaWAN. Pour cela on va utiliser un algorithme d'apprentissage par renforcement de type bandit manchot. De plus afin d'accélérer la convergence de l'algorithme nous n'utiliserons que deux canaux et trois facteurs d'étalement.

2 Explication du code

Pour réaliser ce projet, implémenter l'algorithme, nous avons d'abord dû réfléchir aux structures de données et aux fonctions dont nous aurions besoin ainsi qu'à l'initialisation de certaines données.

2.1 Structures données

La formule utilisée et certaines contraintes nécessite de conserver plusieurs données :

La moyenne de la machine Il faut conserver tous les choix de chaque machine à chaque tour. Pour ce faire nous avons décidé de créer une structure "tour" qui est composée du canal et du facteur d'étalement choisit ainsi que de la valeur (s'il y a eu collision 0 et un valeur inversement proportinel au SF sinon). La moyenne est calculée pour toutes les combinaisons mais une seule est choisie à chaque tour donc il ne faudra recalculer qu'une seule valeur c'est pour cela que nous avons décidé que chaque machine posséderais un tableau qui contiendra la moyenne de chaque combinaison.

Le nombre d'utilisation de la machine Il faut savoir pour chaque combinaison combien de fois elle a été utilisé. Pour cela nous avons pris la décision encore une fois de stocker ces valeurs dans un tableau.

La position des équipements En effet, en fonction de sa position un équipement n'a pas forcement accès au même facteur d'étalement c'est pour cela que nous conservons cette donnée dans un tableau.

Pour finir nous avons décidé de réunir toutes ces données dans une seule structure"gain" afin de faciliter la compréhension du code et pour facilité la manipulation des données.

2.2 Fonctions

Ensuite nous avons réfléchi aux fonctions que nous devions implémenter et nous en avons trouvé trois :

- Une pour initialiser à une valeur précise certains tableaux.
- Une autre pour calculer la moyenne.
- Une pour calculer quelle combinaison serait la plus efficace pour la machine donnée. Cette fonction choisit la combinaison dont l'envoie du paquet à le plus de chance de réussir. Si deux ou plus on la même chance alors on en prend une au hasard.
- Et enfin une dernière qui choisit aléatoirement une combinaison. Cette fonction nous permettra de faire une comparaison avec la fonction précedente et ainsi voir si nous avons pu améliorer le nombre de paquet envoyé.

2.3 Initialisation

Passons maintenant à l'initialisation des données. Nous avons décidé d'initialiser la moyenne de chaque combinaison utilisable à 1 afin d'obliger le programme à choisir au moins une fois chaque combinaison disponible. Les combinaisons qui ne doivent pas être utilisées sont initialisées à moins un. Nous ne pouvions pas l'initialiser à 0 sinon le programme ne prendrait toujours la même combinaison. On initialise aussi le nombre de fois que la combinaison a été utilisé à un car sinon avec la formule cela peut créer une division par 0. De plus il faut, pour le compte du nombre de tour, faire attention a bien commencé a un sinon, à cause de la formule encore une fois, on va avoir log(0). Pour positionner les équipements dans les trois cercle nous avons utilisé le hasard.

3 Résultat

Une fois le programme implémenté il a fallu le tester et voir si les résultats nous semblaient corrects. Pour cela nous avons décidé de ne faire varier que le nombre d'équipements de 5 à 10 et de garder constant le nombre de tours sachant que le nombre de combinaisons possible est 6. Pour chaque nombre d'équipements nous avons exécuté le programme 20 fois afin d'avoir une moyenne et non pas une valeur unique pour que notre résultat soit le plus fiable possible.
Afin de pouvoir mieux observer la différence entre les deux méthodes nous avons décidé d'utiliser un graphique.
Dans les deux cas, plus le nombre d'équipements augmente plus le nombre de paquets envoyés diminue. Cela nous semble cohérent puisque une fois que les six combinaisons ont été choisit plus aucun paquet ne peut passer cela pose moins de problèmes tant que le nombre d'équipements souhaitant envoyer un paquet est inférieur au nombre de combinaison possible car les équipements ont toujours la possibilité de choisir une combinaison non utilisée alors que quand le nombre d'équipement est plus élevé que le nombre de combinaisons, les derniers équipements n'ont presque aucune chance d'avoir encore une combinaison de disponible.
On remarque cependant qu'en utilisant la méthode bandit manchot quand le nombre de combinaisons est inférieur au nombre d'équipements alors cette méthode est bien plus efficace et entraine bien moins de collision. Le programme sait quelle combinaison choisir pour réduire le nombre de colisions.