ADR LoRa : Les bandits sont rattrapés par la loi !

Jules COURJAULT[1] Baptiste VRIGNEAU[1] Olivier BERDER[1] Claude GUICHAOUA[2] Yvon LEGOFF[3]

[1]Univ. Rennes, CNRS, IRISA, 6 rue de Kerampont, 22300 Lannion, France

[2]CG-Wireless, Le Moustoir, 29710 Plogastel St Germain, France.

[3]CG-Wireless, Espace Volta, 22300 Lannion, France

Résumé – Les réseaux d'objets connectés font face à deux problèmes majeurs : la consommation d'énergie et la congestion du réseau susceptible d'accueillir un grand nombre de nœuds. Certaines normes de communication permettent une optimisation des paramètres de transmission, grâce notamment à des mécanismes d'intelligence artificielle, mais les performances de ces derniers sont souvent surestimées. Grâce au nouveau simulateur J-LoRaNeS, nous montrons que les bandits multi-bras utilisés pour optimiser les communications LoRa n'apportent pas le gain annoncé dans la littérature lorsqu'ils sont confrontés à des réseaux réalistes et en profitons pour ouvrir quelques possibilités d'amélioration.

Abstract – The Low Power Area Networks (LPWAN) are very promising for IoT but have to face two major challenges : energy consumption and possible congestion due to the huge number of nodes they have to handle. Fortunately LPWAN transmission parameters can be optimised, e.g. thanks to artificial intelligence algorithms, but the performance of the latter are often overestimated. Thanks to the new J-LoRaNeS simulator, we show that multi-arm bandits do not bring the gain announced in the litterature when a realistic LoRa network and legacy constraints are considered, and we also propose some possible improvements.

1 Introduction

Plusieurs technologies sans fil à longue portée, regroupées sous le terme LPWAN *Low-Power Wide Area Network* sont apparues ces dernières années, avec pour objectif de répondre aux besoins croissants de remonter les informations récoltées par divers objets connectés. La technologie LoRa [9], portée par Semtech et l'alliance LoRa, offre notamment un très bon compromis entre portée, débit et consommation. Elle permet d'avoir un débit allant de $0,3$ kbps à 50 kbps dans les bandes Industrielle, Scientifique et Médicale (ISM) 433 MHz, 868 MHz ou 915 MHz en utilisant le principe de l'étalement linéaire de fréquence. Les principaux paramètres de la communication sont le facteur d'étalement (SF pour *Spreading Factor*), la puissance d'émission (TP pour *Transmit Power*), la largeur de bande (BW pour *Bandwidth*) et le taux de codage (CR pour *Coding Rate*). Le débit, la consommation d'énergie et les performances de la transmission varient ainsi selon les choix de ces paramètres [3]. Afin de s'adapter au canal de communication et d'assurer la réception des données transmises, un mécanisme d'adaptation des paramètres de communication nommé *Adaptive Data Rate* (ADR) est utilisé [2]. Cet algorithme souffre de plusieurs problèmes [6], notamment un temps de convergence conséquent, entraînant une surconsommation au démarrage du réseau. Pour corriger ce problème et augmenter les performances finales, des propositions d'améliorations de l'algorithme ont été effectuées dans la littérature, dont certaines [14, 13] utilisent des algorithmes du Bandit Multi-Bras (BMB), qui sont des techniques d'apprentissage par renforcement, un pan de l'intelligence artificielle. Ces algorithmes ont besoin de calculer une récompense pour évaluer la qualité du choix des paramètres lors d'une transmission, et l'on constate dans les propositions de la littérature que cette récompense se base en général sur un accusé de réception (ACK) transmis par le serveur après réception d'un paquet montant. Cependant, en émettant sur les bandes ISM, les équipements LoRa doivent respecter un *Duty Cycle* (DC) allant de 1% à 10% selon les régions et les bandes utilisées. Or, la majorité des comparaisons effectuées entre l'ADR et les algorithmes BMB se font sans le respect de celui-ci. Ceci peut poser problème car l'absence d'accusé de réception est interprétée par ces algorithmes comme un paquet montant non reçu, ainsi le calcul de la récompense se fera sur une hypothèse fausse. Que deviennent les performances des mécanismes d'adaptation basés sur les algorithmes BMB lorsque le DC est respecté ? La suite de l'article s'organise de la façon suivante : la section 2 présente le simulateur utilisé, J-LoRaNeS, développé dans le langage Julia. La section 3 donne quelques détails sur le fonctionnement de l'ADR, ainsi que le fonctionnement des algorithmes BMB, et la section 4 présente les paramètres et résultats de simulations avant de tirer les conclusions dans la section 5.

2 J-LoRaNeS : un simulateur LoRa utilisant le langage Julia

2.1 Pourquoi Julia ?

De nombreux simulateurs de réseau LoRa sont disponibles tels que FLoRa [11] ou LoRaWAN ns3 [7] pour ne citer qu'eux. Cependant, ces simulateurs ne combinent pas rapidité de simulation et facilité de personnalisation. Le langage Julia [1] est un bon candidat pour pallier à ces problèmes. En effet, Julia est un langage compilé mais offre une syntaxe de haut niveau, permettant un bon compromis entre vitesse d'exécution et facilité de développement. L'une des caractéristiques de Julia facilitant la personnalisation est le *multiple dispatch*

Ces travaux sont effectués dans le cadre d'une thèse co-financée par l'EIT Digital

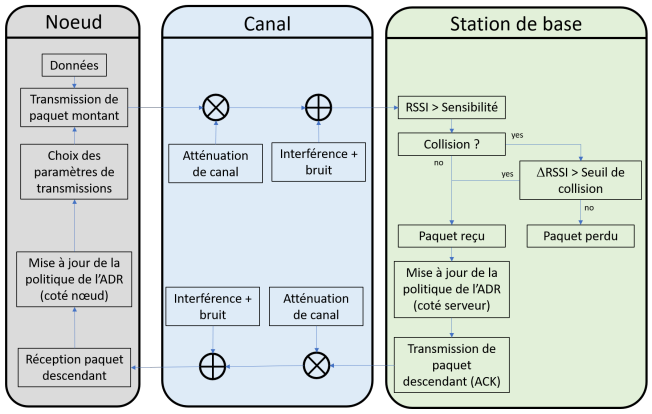


FIGURE 1 : Schéma bloc présentant l'interaction entre nœud et station de base simulée par J-LoRaNeS

permettant de redéfinir une fonction déjà existante en fonction du type des paramètres passés à la fonction.

2.2 Architecture du simulateur

La figure 1 présente l'interaction entre un nœud du réseau et une station de base (SB). Le nœud commence par envoyer un paquet avec les paramètres choisis. Pour chaque SB présente dans le réseau, l'atténuation du canal est calculée en fonction du modèle fourni par l'utilisateur, puis le RSSI du paquet est comparé à un seuil de sensibilité [10]. Si le paquet a un RSSI suffisant, on vérifie que le paquet peut être reçu malgré les collisions avec les transmissions concurrentes [4]. Si le paquet est reçu, alors la politique de choix des paramètres coté serveur est mise à jour (voir section 3), puis s'il est demandé par le paquet reçu, ou s'il est nécessaire pour communiquer les nouveaux paramètres de communication, un paquet descendant est programmé. Une SB disponible, c'est-à-dire respectant le DC, et ayant reçu le paquet montant est alors sélectionnée pour transmettre le paquet descendant. Le paquet descendant suit alors le même chemin de décision que le paquet montant pour savoir s'il a été reçu ou non. Puis, quel que soit le résultat de réception de ce paquet, la politique de l'ADR est mise à jour coté nœud, et de nouveaux paramètres de transmissions sont choisis avant de faire une nouvelle transmission montante.

3 Les différents mécanismes d'adaptation

3.1 ADR LoRaWAN

L'algorithme d'ADR proposé pour les réseaux LoRaWAN se compose de 2 parties. La première est décentralisée et a pour objectif d'augmenter la probabilité d'atteindre une SB du réseau. Pour ce faire, SF ou TP sont augmentés si un nombre N de transmissions montantes consécutives ne reçoivent pas d'accusé de réception. La seconde partie de cette mécanique est centralisée, le serveur du réseau va chercher à rendre la communication plus efficace énergétiquement. Pour cela, il enregistre le rapport signal à bruit de 20 paquets montants puis calcule une marge, permettant de déterminer de quelle façon TP et SF peuvent être modifiés afin de réduire la consommation de la transmission.

3.2 Les mécanismes utilisant les BMB

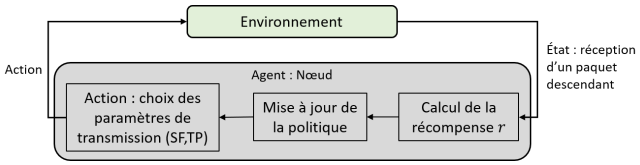


FIGURE 2 : Principe de fonctionnement des algorithmes BMB appliqué à LoRa

Ici, contrairement au cas LoRaWAN, le mécanisme se fait en une seule partie, de manière décentralisée. La figure 2, présente le fonctionnement de base des algorithmes BMB appliqués à LoRa. Dans un premier temps, l'agent choisit un bras, *i.e.* une combinaison des paramètres SF et TP, puis effectue une transmission avec les paramètres choisis. Cette action placera l'environnent dans un certain état, l'observation de cet état permet le calcul d'une récompense par l'agent. Ici l'observation de l'état se fait par la réception d'un accusé de réception, permettant le calcul de la récompense, à l'instant $t : r(t) = 1$ si l'ACK est reçu, $r(t) = 0$ sinon. Les étapes suivantes sont la mise à jour de la politique de l'algorithme puis le choix des paramètres de communication. Ces étapes sont effectuées différemment selon la nature de l'algorithme utilisé. Les algorithmes BMB utilisés dans cet article sont $\epsilon-$Greedy [12] et Thomson-Sampling (TS) [8].

3.2.1 $\epsilon-$Greedy

À chaque bras k, une valeur $\hat{\theta}_k \in [0; 1]$ est associée, qui correspond à la récompense reçue en moyenne par le bras lorsqu'il a été choisi. La mise à jour de la politique consiste donc à calculer $\hat{\theta}_k$ pour le bras utilisé une fois la récompense reçue. Pour la phase de sélection du bras, l'algorithme devra choisir entre exploiter ou explorer. S'il choisit d'exploiter alors le bras avec la valeur de $\hat{\theta}_k$ la plus importante sera choisi. En cas d'exploration, un bras est tiré au hasard. Le choix entre exploration ou exploitation est fait aléatoirement, l'algorithme explorera avec une probabilité $\epsilon \in]0; 1]$. Dans cet article nous avons choisi $\epsilon(t) = \frac{N_b}{N_b + \sum_k n_k(t)}$, avec N_b le nombre total de bras et $n_k(t)$, le nombre de transmissions effectuées avec les paramètres du bras k. Ce choix d'une valeur décroissante dans le temps pour ϵ permet une forte exploration au déploiement, puis une exploitation forte après un certain temps, assurant ainsi une utilisation des paramètres apportant une forte récompense.

3.2.2 Thomson-Sampling

À chaque bras k est associée une variable aléatoire suivant une loi bêta de paramètres (α_k, β_k), dont la valeur moyenne est $\hat{\theta}_k = \frac{\alpha_k}{\alpha_k + \beta_k}$. À l'initialisation les paramètres de la loi bêta ont les valeurs suivantes : $\alpha_k = \beta_k = 1$. La mise à jour de la politique consiste à changer les paramètres de la loi : $(\alpha_k(t+1), \beta_k(t+1)) = (\alpha_k(t), \beta_k(t)) + (r(t), 1-r(t))$, si le bras k a été choisi, sinon les paramètres restent inchangés. Pour choisir le bras qui sera utilisé pour la prochaine transmission, l'algorithme fait un tirage aléatoire pour chaque bras et choisit le bras qui a obtenu la plus grande valeur, ainsi le bras ayant le $\hat{\theta}_k$ le plus grand sera choisi en moyenne, mais les bras ayant une grande variance, c'est à dire ceux qui ont été explorés peu

de fois, auront tout de même une chance importante d'être choisis.

4 Simulations et résultats

Pour estimer l'apport réel des algorithmes présentés ci-dessus par rapport à l'ADR classique, nous utilisons le simulateur présenté dans la section 2. Dans cet article nous présentons les résultats pour un réseau avec une SB, 500 noeuds répartis uniformément dans un carré de coté 20 km, envoyant un paquet toutes les 10 minutes en moyenne. Le modèle de canal utilisé lors des simulations est le modèle d'Okumura-Hata [5] pour les petites et moyennes villes. Les mécanismes d'adaptation qui seront utilisés sont : l'ADR de LoRaWAN, un ADR utilisant l'algorithme BMB $\epsilon-$Greedy, et un autre utilisant l'algorithme TS. Dans le cas des ADR BMB, nous comparerons les cas "Oracle", c'est à dire que la SB envoie un accusé de réception pour chacun des paquets montants reçu nonobstant le rapport cyclique, avec les cas "Réels", c'est à dire le cas où la SB respecte le DC de 1 à 10% imposé par la réglementation, ce faisant, tout paquet montant reçu n'est pas nécessairement suivi par un accusé de réception, engendrant un mauvais calcul de récompense par les algorithmes. Les combinaisons de bras possibles sont identiques aux combinaisons possibles pour l'ADR LoRaWAN, *i.e.* pour SF 7 les TP possibles vont de 2 dBm à 14 dBm par pas de 3 dB, et pour les SF allant de 8 à 12, seule la TP de 14 dBm peut être sélectionné.

4.1 Évolution temporelle du Taux de Réception des Paquets

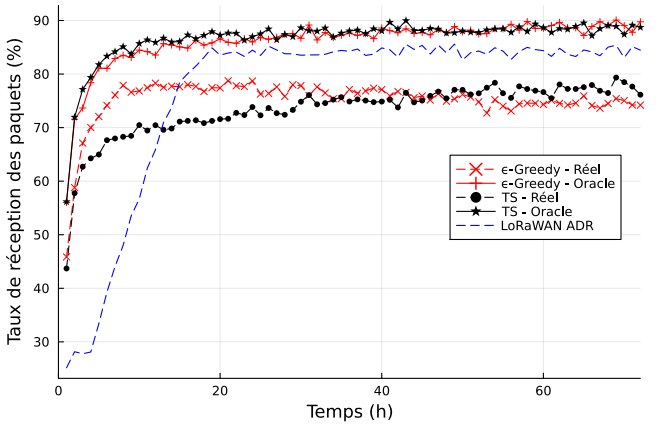


FIGURE 3 : Taux de réception des paquets pour les BMB $\epsilon-$Greedy et TS, dans les cas oracle et réel, ainsi que l'ADR LoRaWAN.

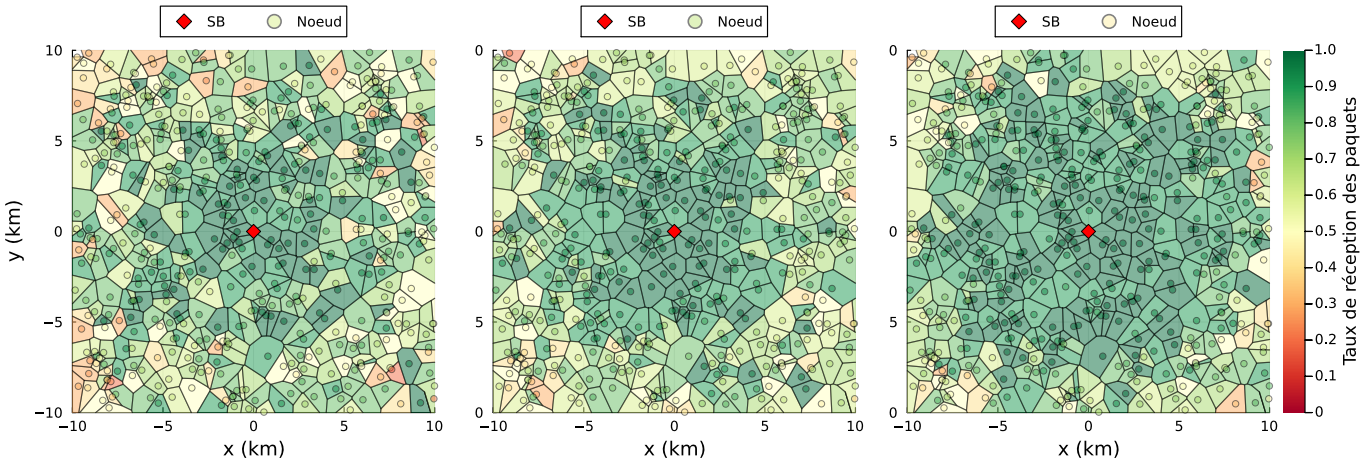
La figure 3 présente l'évolution du Taux de Réception des Paquets (TRP) pour les ADR : LoRaWAN en traits pointillés bleu, $\epsilon-$Greedy en rouge et TS en noir, les versions oracles sont en trait continu, alors que les versions réelles des algorithmes BMB sont les lignes pointillées. Le TRP est calculé sur une fenêtre glissante d'une heure. Sur cette figure, on constate que les algorithmes BMB ont un temps de convergence bien plus faible que l'ADR LoRaWAN. Par ailleurs, le TRP obtenu après convergence est meilleur que l'ADR de LoRaWAN pour les versions Oracle (avec des résultats quasi similaires pour les deux versions). Malheureusement, il s'avère que dans le cas réel les algorithmes BMB sont moins performants que l'ADR LoRaWAN. En effet, l'algorithme $\epsilon-$Greedy atteint son maximum en 18 heures avec un TRP à 78%, puis ses performances diminuent jusqu'à atteindre 75% en fin de simulation, alors que dans le cas LoRaWAN le TRP est en moyenne de 83% après convergence. Dans le cas TS, on constate une croissance rapide du TRP jusqu'à 8 heures de simulation, à ce moment-là le TRP est de 67%, puis on observe une croissance lente jusqu'à atteindre un TRP de 78% en fin de simulation, et un dépassement des performances de l'algorithme $\epsilon-$Greedy à partir de 48h de simulation. Cette dégradation de performance dans le cas réel est due au mauvais calcul de récompense causé par l'absence d'accusé de réception, malgré la réception du paquet montant, sous-évaluant ainsi les performances des bras ayant de bonnes performances, alors que les bras ayant de mauvaises performances sont moins impactés. On constate une diminution des performances du cas $\epsilon-$Greedy à cause du choix de la probabilité d'exploration ϵ. En effet, l'algorithme sera bloqué sur un choix de bras non-optimal, mais pourtant perçu comme tel, à cause de la faible valeur ϵ après 20h de simulation. Ce comportement dure jusqu'à ce que la valeur moyenne de ce bras passe en dessous de celle d'un autre bras. Ce problème pourrait être réglé en utilisant une valeur fixe pour ϵ, mais pour ne pas trop impacter les performances de l'algorithme, la valeur d'ϵ doit être petite, entraînant un long temps de convergence. Dans le cas TS l'apprentissage est beaucoup plus lent car la décroissance de la variance l'est aussi à cause du mauvais calcul de récompense, mais cela permet à l'algorithme de mieux résister aux mauvais calculs de récompenses par rapport à l'algorithme $\epsilon-$Greedy.

4.2 Comparaisons topologiques après convergence

Notre nouveau simulateur est capable de fournir une grande variété de résultats et nous avons travaillé sur les représentations possibles pour faciliter l'analyse du réseau. Par exemple, la figure 4 présente le TRP moyenné sur les deux dernières heures de simulation pour chaque noeud du réseau dans des conditions réelles pour les algorithmes $\epsilon-$Greedy, TS et l'ADR LoRaWAN. Afin d'estimer une carte des TRP moyens, un pavage de Voronoï permet d'observer les zones bien couvertes et celles pouvant créer des difficultés. Une carte pour chaque algorithme est disponible pour la comparaison. Sans surprise et pour les trois cas, on constate que les nœuds les plus performants sont ceux qui sont les plus proches de la SB. En complément des conclusions de la figure 3, on constate que les nœuds les plus proches de la SB ont des performances acceptables (TRP>80%) dans un rayon de 4 à 5 km pour l'algorithme TS, contre 8 à 9 km pour l'ADR LoRaWAN. L'ajustement des paramètres n'est pas possible pour les nœuds en bordure du réseau à cause de l'absence d'ACK. En effet, au bord du réseau, peu de configurations permettent d'avoir un bon TRP, contrairement au centre, or les performances de ces bras sont mal estimées à cause de l'absence d'ACK.

5 Conclusions et perspectives

Dans cet article nous comparons les performances d'algorithmes permettant l'adaptation des paramètres de communica-



(a) $\epsilon-$Greedy réel (b) TS réel (c) ADR LoRaWAN

FIGURE 4 : TRP moyen pour chaque nœud du réseau en fin de simulation

tions LoRa tels que présentés dans la littérature, sans respect du DC, avec une version réaliste de ces algorithmes qui respecte l'occupation du canal imposé par la réglementation. Pour ce faire, nous utilisons notre tout nouveau simulateur développé en Julia offrant une modularité de configuration et de nouvelles possibilités tant en termes de valeurs calculées qu'en visualisation de données. Les résultats montrent une forte dégradation des performances des algorithmes BMB, qui ne sont plus capables de dépasser les performances de l'algorithme prévu dans la norme LoRaWAN. Bien que les très bons résultats présentés dans la littérature pour les algorithmes BMB ne soient pas atteignables en pratique, cela ne veut pas dire qu'il ne faut pas les utiliser. En effet, ces résultats ont montré la faisabilité et le bon fonctionnement dans un cas favorable, qui est réalisable dans des conditions de faible densité de nœuds dans le réseau (DC est respecté). Il faut maintenant réfléchir à une solution qui pourrait fonctionner dans les cas à forte densité de nœuds dans le réseau, en particulier les plus éloignés de la SB. Un autre aspect important qui n'a pas été abordé dans cet article est l'optimisation de la consommation énergétique des nœuds. Traité par l'algorithme LoRaWAN, il pourrait aussi être mis en oeuvre par les algorithmes BMB en ajoutant une composante prenant en compte la consommation énergétique dans le calcul de la récompense. Le poids de cette composante serait plus important dans les cas à faible consommation énergétique, permettant ainsi de favoriser les bras à faible consommation énergétique en cas de TRP similaire entre plusieurs bras.

Références

[1] Julia language. `https://julialang.org/`.

[2] LoRa ALLIANCE : LoRaWAN 1.1 specifications. `https://lora-alliance.org/resource_hub/lorawan-specification-v1-1/`, 2017.

[3] J. COURJAULT, B. VRIGNEAU, O. BERDER et M. R. BHATNAGAR : A computable form for lora performance estimation : Application to Ricean and Nakagami fading. *IEEE Access*, 9:81601–81611, 2021.

[4] D. CROCE, M. GUCCIARDO, S. MANGIONE, G. SANTAROMITA et I. TINNIRELLO : Impact of LoRa imperfect orthogonality : Analysis of link-level performance. *IEEE Communications Letters*, 22(4):796–799, 2018.

[5] E. HARINDA, S. HOSSEINZADEH, H. LARIJANI et R.M. GIBSON : Comparative performance analysis of empirical propagation models for lorawan 868MHz in an urban scenario. *In 5th World Forum on Internet of Things (WF-IoT)*, pages 154–159. IEEE, 2019.

[6] S. LI, U. RAZA et A. KHAN : How agile is the adaptive data rate mechanism of LoRaWAN ? *In IEEE Global Communications Conference (GLOBECOM)*, pages 206–212, 2018.

[7] Davide MAGRIN : Network level performances of a LoRa system. 2016.

[8] D.J. RUSSO, B. Van ROY, A. KAZEROUNI, I. OSBAND, Z. WEN *et al.* : A tutorial on Thompson sampling. *Foundations and Trends® in Machine Learning*, 2018.

[9] O.B.A. SELLER et N. SORNIN : Low power long range transmitter, Aug. 2014. US Patent App. 14/170,170.

[10] SEMTECH : SX1276/77/78/79–860 MHz to 1020 MHz low power long range transceiver. Rapport technique Datasheet, Mar.

[11] M. SLABICKI, G. PREMSANKAR et M. Di FRANCESCO : Adaptive configuration of LoRa networks for dense IoT deployments. *In NOMS - IEEE/IFIP Network Operations and Management Symposium*, pages 1–9. IEEE, 2018.

[12] R. S SUTTON et A. G BARTO : *Reinforcement learning : An introduction*. MIT press, 2018.

[13] D.-T. TA, K. KHAWAM, S. LAHOUD, C. ADJIH et S. MARTIN : LoRa-MAB : A flexible simulator for decentralized learning resource allocation in IoT networks. *In 2019 12th IFIP Wireless and Mobile Networking Conference (WMNC)*, pages 55–62. IEEE, 2019.

[14] B. TEYMURI, R. SERATI, N.A. ANAGNOSTOPOULOS et M. RASTI : LP-MAB : Improving the Energy Efficiency of LoRaWAN Using a Reinforcement-Learning-Based Adaptive Configuration Algorithm. *Sensors*, 2023.